Chapter 6
Design Process for Applications on Blockchain

with Sin Kuang Lo

Software design is a creative process, which includes proposing and evaluating solutions to complex problems with many conflicting constraints. The final design of a software system is the result of many design choices about the selection, configuration, and integration of software, hardware, and communications components. This chapter presents a design process for architecting systems based on blockchains.

For a system that can potentially use blockchain, the first design choice is to decide whether to use a blockchain or conventional technologies. We discuss this choice in Section 6.1 and give four examples in Section 6.2. When using a blockchain, there are subsidiary design choices including whether to use a private blockchain or a public blockchain, what consensus protocol fits best, and what the block frequency should be. Chapter 3 identifies a variety of design choices, and in Section 6.3 we discuss how to address them. Often in a blockchain-based system, some data is stored on the blockchain, while other data is stored and communicated using conventional technologies, so another design choice is which data should be stored where.

6.1 Evaluation of Suitability

Due to their fundamental properties and limitations, blockchains do not fit all scenarios. Thus, before designing a system, the suitability of blockchain needs to be evaluated against the scenarios and requirements.

Figure 6.1 shows a process to evaluate the suitability of blockchain technology. There are seven main questions to be answered, shown as white diamonds. For some of them, subsidiary questions are shown as grey diamonds. The following subsections discuss these questions in detail.

X. Xu et al., *Architecture for Blockchain Applications*,
https://doi.org/10.1007/978-3-030-03035-3_6

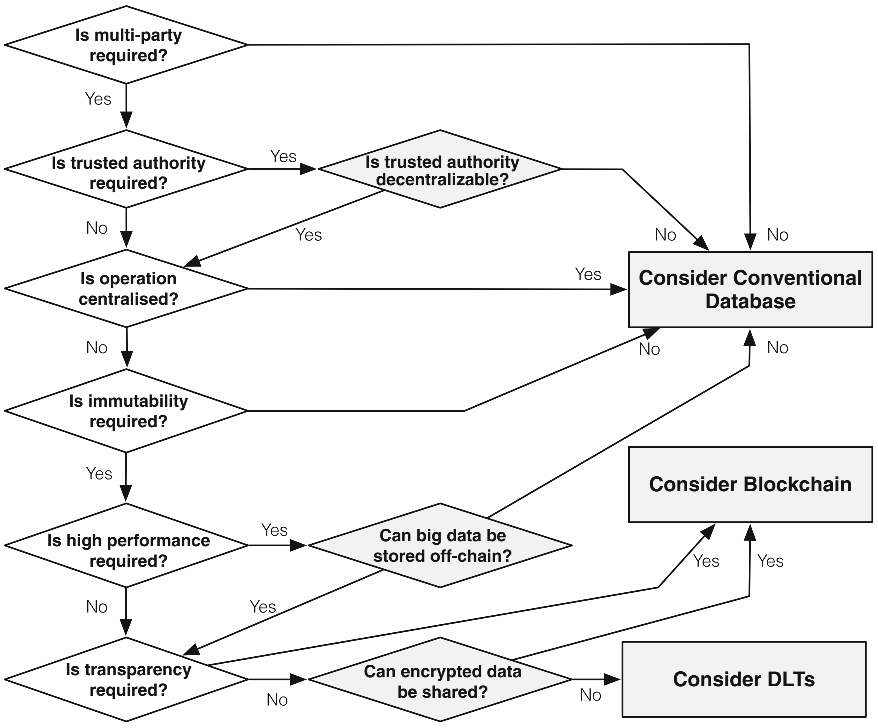
Is multi-party required?
Yes
Is trusted authority required?
Yes
Is trusted authority decentralizable?
No
Yes
No
No
Is operation centralised?
Yes
Consider Conventional Database
No
No
No
Is immutability required?
Yes
Consider Blockchain
Is high performance required?
Yes
Can big data be stored off-chain?
Yes
Yes
No
Yes
Is transparency required?
No
Can encrypted data be shared?
No
Consider DLTs

Fig. 6.1 Evaluation of suitability of blockchain and other DLTs. © 2017 IEEE. Reprinted, with permission, from Lo et al. (2017)

6.1.1 Multiparty

Does the system need to serve multiple different parties? A blockchain is not suitable for systems that only serve individual isolated users, because a conventional database will be simpler and more efficient. There are many different kinds of multiparty systems. Consider the supply chain domain, which has complex, dynamic, multiparty arrangements with regulatory and logistical constraints spanning jurisdictional boundaries. Information exchange in a supply chain can be as important and difficult as the physical exchange of goods. The multiple users here may be manufacturers, shipping companies, transport infrastructure organizations, financial services firms, or regulators. Another example domain might be inter-bank payments and reconciliation. Here the multiple parties are at least two different banks, but may also include the account holders performing payment transfers between the banks. So, parties might be organizations or individuals. In these examples, the different parties are legally distinct. However, even within one large enterprise (or government), there may be different functional or geographic divisions or departments. These informational or administrative ‘silos’ may need to be served as multiple parties. Blockchains can be suitable for supporting multiparty systems, because the blockchain is a physically distributed but logically centralized infrastructure, providing a single view of truth across those parties.

6.1.2 Trusted Authority

A trusted authority is an entity that is relied upon to perform a function, like operating a system. If a single party can or must be relied upon as a trusted authority by all of the parties served by a system, then a blockchain may not be necessary. Instead, that trusted authority could implement a traditional centralized solution using conventional technologies. Most current complex systems are controlled by a trusted authority. Examples of these authorities include banks and government departments. The scope of the system being designed is important in deciding this question. For bank accounts, the bank will be a trusted authority. However, for inter-bank payments, each participating bank will not be a trusted authority; instead the conventional approach is for banks to collectively rely upon separate authorities to facilitate inter-bank payments. For example, within a country that trusted authority might be a central bank.

Relying on a trusted authority creates a single point of failure for the system. When a trusted authority experiences a problem, users accessing its services are affected. Technical single points of failure can be mitigated by using redundancy in conventional distributed systems architectures. However, those solutions do not address single points of organizational or business failure that remain present when relying on a trusted authority. These possible failures might include business failures, service interruptions, data loss, or fraud. For situations where the trusted authority is a monopoly or oligopoly service provider, there is also the possibility of what economists call 'rent-seeking' behaviour, which can unreasonably limit access to the service and can reduce efficiency through excessive charges.

Even when a natural trusted authority might in principle be available, in practice it might be difficult for everyone to accept reliance on that party. Consider a government with multiple different departments or agencies. Large enterprises or government could in principle define a central agency to provide services for coordinated operation across their whole organization. However, centralization of services can be perceived as a loss of control or power, and so in practice it may be difficult to achieve this kind of administrative centralization.

Blockchain can support systems where there is no single party that is acceptable or suitable for operating the system. That is because a blockchain is operated jointly by a collective of nodes. Using a blockchain does not remove trust, because users are still exposed to risk in their use of blockchain technology. In a blockchain, what is trusted (i.e. relied upon) is the blockchain software, the incentive or contractual mechanisms driving the behaviour of processing nodes that operate the blockchain system, and the trusted third-parties that act as 'oracles' which record information about the external world on the blockchain. Although a blockchain does not remove trust, it can remove the need to trust a single specific third-party to maintain a ledger and so is sometimes called a 'distributed trust' mechanism.

6.1.3 Operation

Given that a system supports multiple parties, and given no party is suitable as a trusted authority for administering the system, might centralized operation of the system still be possible? A common approach is that a group of parties might form a joint venture to operate a conventional centralized system. Credit card associations such as Visa and Mastercard are examples of this approach, formed as kinds of joint ventures between banks.

However in some cases, it is not possible or desirable to centralize operation of the system. The centralized operation of the system may lead to the administering party becoming a trusted authority, which will not always be acceptable to the parties using the system. Forming a new entity like a joint venture might be too costly for a given scenario. Also, the centralized administration of the system may still allow single points of business failure for the system. A distinctive benefit of blockchain-based systems is that there does not have to be a single authority or system operator. Eliminating single points of failure can increase system reliability or availability.

6.1.4 Data Immutability and Non-repudiation

Is data immutability required and acceptable? *Data immutability* means data cannot be changed or altered after its creation. Immutability supports *non-repudiation* which is the assurance that a party cannot deny the authenticity of their signature on a document or a message from them. Blockchains naturally support data immutability in the ledger, whereas conventional technologies naturally support mutable data. What is important as a requirement can vary from system to system.

Although the blockchain transaction history is immutable, the latest view of the current state in a blockchain can change. For example, a transaction may need to update the owner of an asset. What is recorded to the ledger in this case is the new owner for the asset, and so all that changes is our view of the latest owner. In a blockchain, the linking of blocks in a chain of cryptographic hashes supports immutability for historical transactions. In practice, past blocks in the blockchain data structure cannot be changed because it is continually replicated across many different locations and organizations; attempts to change it in one location will be interpreted as an attack on integrity by other participants and will be rejected. In economies where third-party service providers are not always trustworthy, a significant benefit of blockchain systems may be in the strong support that they can provide for immutability and non-repudiation. On a blockchain, the immutability of historical transactions which are cryptographically signed means that there is always strong evidence that those transactions were performed by someone with control over those cryptographic keys.

On the other hand, it is not possible to change the transaction history in most blockchains. This is normally a good thing in supporting data integrity. However, it can cause problems if blockchain contains illegal content, or if a court orders content to be removed from the blockchain. It will be easier to support these requirements using conventional technologies. Similarly, in blockchain systems, problems may arise such as disputed transactions, incorrect addresses, exposure or loss of private keys, data-entry errors, or unexpected changes to assets tokenized on blockchain. The immutability of blockchain ledgers may make them less adaptable than conventional technologies controlled by trusted third-party organizations that support rollback.

Using blockchain to achieve immutability and non-repudiation may be relatively expensive compared to other persistence mechanisms. There are existing mechanisms available to prove the originality of data, like hashing technology, and cryptographically signed data. In traditional database systems, the ACID properties (Atomicity, Consistency, Isolation, and Durability) are critical. However, for blockchains that use Nakamoto consensus (longest chain wins), the classic durability property does not hold because a transaction initially thought by a participant to be committed (i.e. on the longest chain) may later turn out to have been on a shorter chain, and so no longer be committed. Such blockchains only offer a long-run probabilistic durability property, and therefore are not immutable in a simple way. However, (a) switching to a longer chain is evident to participants, and (b) when a transaction has been committed to a blockchain for a sufficiently long time, it will in practice be immutable. Blockchains that use other consensus mechanisms (such as Practical Byzantine Fault Tolerance) can offer stronger, more conventional immutability properties. However, typically these consensus mechanisms can only be used where there is a small number of well-known nodes participating in the operation of the blockchain.

6.1.5 High Performance

Does the system need to support extremely short response times or process very large amounts of data? If so, conventional technologies may be more suitable than blockchain technology.

System performance usually relates to *latency* which is the system response time and *throughput* which is the aggregate system work rate. Blockchain systems such as Bitcoin and Ethereum cannot currently match the maximum throughput of conventional transaction processing systems such as the Visa payments network. This is a known and current limitation but is being addressed by the development of new mechanisms such as sharding, state channels, and reduced inter-block time. While blockchains are currently not highly scalable, this is not necessarily an inherent limitation, and may be overcome in the future. Consortium and private blockchains with careful design and performance tuning have much better performance compared to public blockchains. When data has previously been written to the blockchain, read latency is the response time for accessing historical data from a blockchain client. Read latency can be much faster on blockchain than with conventional technologies, because clients can keep a full local copy of the database, and so there are no network delays. The request to write data into a blockchain is done by sending a transaction to the network. The write latency is probabilistic, and there are several sources of uncertainty. All blockchains will have small network delays. For blockchains with Nakamoto consensus, a node should not be highly confident that the most recent block it saw will ultimately be included in the main chain. So, to increase the confidence that data has successfully been committed to the blockchain, we can wait for a number of confirmation blocks. Waiting for more confirmation blocks will increase write latency.

Blockchains are inherently not suitable for storing Big Data, i.e. large volumes of data or high-velocity data. This is because on a blockchain there is massive redundancy in the large number of processing nodes holding a full copy of the distributed ledger. Big Data is hard to physically move in a distributed system, and the large numbers of replicas make it infeasible to store it on a blockchain.

6.1.6 Transparency

The third question in the design process is whether data transparency is required or acceptable in the system. *Data transparency* is the property that data is available and accessible to by other parties in the system. Examples include Facebook public newsfeed posts or Twitter public tweets. Anyone can access and read these posts. Social media such as Facebook or Twitter support *confidentiality* by allowing users to choose what they publish to the public or to specific audiences. Consider also the supply chain domain. Logistics efficiency can be improved by providing greater transparency on the status of shipments and processes, which are currently often opaque. Using blockchain in trade finance to evidence trade-related documents can reduce lending risk, and smart contracts can control inter-organizational process execution, and transparently automate delayed or instalment payments. However, very often customer relationships, pricing, or even aggregate transaction volume are commercially sensitive information that parties do not want to share widely.

Blockchain provides a neutral platform where all participants can see and audit the published data. This is important to guard integrity, with validation by all processing nodes. In a public blockchain, nodes validate that cryptocurrency transfers are from addresses that have enough cryptocurrency and signed with an authorized private key. For smart contracts, nodes validate that the effects of the smart contract program execution are correctly recorded on the blockchain. If data transparency is required or acceptable, a blockchain may be suitable. However, if data transparency is not acceptable, it can be difficult to use a blockchain to manage that data. Confidentiality is harder to establish in blockchain-based systems, because information is visible to all participants.

Another confidentiality concern is the amount of interactions between parties. It is possible to create a new address for each transaction, but the flow of assets may still be used to infer relationships between addresses. Even if parties try to use pseudonyms, the contents of a transaction are publicly visible. Reuse of addresses and their connection via transfers of digital currency can provide opportunities to reidentify participants. Nonetheless, this limitation does not matter for all use cases. For example, public blockchains may be suitable as infrastructure for public advertising or fully open government registries, even in highly regulated industries. Consider that banks advertise on television, but television is not a highly regulated banking transaction system. Integrity in advertising may be required, but rather than privacy or confidentiality, publicity is important. Public blockchains can provide integrity and publicity. Other examples might include systems for secure software package management and IoT device configuration updates.

Sometimes, although raw data cannot be shared, it may be acceptable to share encrypted forms of that data, and in such cases a blockchain could be used. Information could be encrypted before being uploaded to the blockchain: asymmetrically with a particular party's public key, so that only this party can decrypt it, or symmetrically with a shared secret key, so that the group of parties with access to the secret key can decrypt it. The latter case requires a secure means of exchanging the secret key. Encrypting data before storing it on a blockchain may increase confidentiality, but will reduce performance and may harm independent auditability.

Encrypting data will make it difficult or impossible to use smart contracts with that data. If information needs to be processed by smart contracts, the information typically has to be decrypted. This is because smart contract code runs on all nodes of the network, and thus any of them needs to be able to process the input data. This is required to achieve consensus on the outcomes of smart contract execution. Embedding keys within a smart contract would reveal the keys to all participants of the blockchain network.

Sometimes encryption is not acceptable because there may be concerns about successful encryption key management or future technological developments in decryption (such as through quantum computing). Encrypted data may still reveal information as metadata, such as aggregate transaction volume.

Greater transparency is in tension with confidentiality, even if pseudonyms and encryption are used. Consortium and private blockchains can provide read access controls, but this will not provide commercial confidentiality between competitors on a consortium blockchain. The main trade-off is between the benefits of sharing data within the group of collaborators (visibility) and retaining confidentiality towards competitors where needed. In situations where full data transparency between all participants may not be acceptable, and where encrypting data is not acceptable or workable, a more-controlled data sharing can be enabled by distributed ledger technology platforms that are not full blockchains. Platforms such as R3's Corda or Hyperledger Fabric provide small ledgers shared between parties of interest to each transaction. These platforms may be suitable where greater control is required over confidentiality.

6.2 Example Use Cases for Suitability Evaluation

This section uses the above evaluation framework to assess the suitability of using blockchain for four use cases. The first use case, supply chain, is aligned with the one described in Section 4.1. To illustrate other outcomes, we introduce three additional use cases in brief. Table 6.1 gives the summary of the evaluation results based on the seven questions. *Note that these results are illustrative only and should not be taken as valid guidance for real-world systems.*

6.2.1 Use Case 1: Supply Chain

A supply chain is the collection of processes involved in creating and distributing goods, from raw materials to completed products, through to consumers. According to a Deloitte survey, 42% of the companies in consumer goods and manufacturing planned to spend at least $5 million on blockchain technology in 2017.[1] Walmart has tested blockchain technology for their supply chain management in a pilot project that started on the first quarter of 2017 on tracking pork in the USA and China. The use of blockchain for supply chain is an extremely active area of innovation and technology development.

Supply chains are highly complex multiparty systems that span participants such as farmers, factories, transport providers, and retailers. Operations are distributed and often loosely coupled between participants. Data transparency is desired by participants to support logistics planning and to identify and respond to problems. Controlled confidentiality is required for open supply chain infrastructure, and this could be supported by the use of related-party ledgers in distributed ledger systems or by combining conventional information exchange technologies with hashed information on blockchains to ensure integrity and authorization. However, in vertically controlled supply chains, confidentiality can be managed by the use of a private blockchain. Transaction history and data immutability are desired to enable traceability back to the origin of goods and to control fraud and substitution. Current supply chain systems are often still paper-based, and thus cannot easily share information in real time. Digital solutions often only apply within vertically controlled parts of the supply chain, and information gaps can be created when subcontractors are used or when goods leave the scope of control. The time taken in

[1] https://www.bloomberg.com/news/articles/2016-11-18/wal-mart-tackles-food-safety-with-test-of-blockchain-technology.

Table 6.1 Results of the suitability evaluation of four example use cases

	Supply chain	Electronic health records	Identity	Stock market
Multiparty	Required	Required	Required	Required
Trusted authority	Not required	Decentralized	Not required	Not required
Centralized operation	Not required	Not required	Not required	Not required
Data immutability and non-repudiation	Required	Required	Required	Required
High performance	Not required	Not required	Not required	Required
Data transparency and confidentiality	Transparent (but not fully public)	Confidential	Transparent	Confidential
Sample result	DLT	Conventional system	Blockchain	Conventional system

Results are illustrative only, and should not be taken as ultimate guidance. © 2017 IEEE. Reprinted, with permission, from Lo et al. (2017)

a supply chain is dominated by physical transportation and storage, which moderates demand for performance. Reasonably short latency is required at key points of handover of goods, but there is no requirement for extreme throughput or latency.

Supply chains are a promising area for blockchain-based applications. The complex, dynamic structure of business relationships and operations in a supply chain can be accommodated by the flexible structure of blockchain node networks, and the logically centralized view of information provided by a blockchain supports many of the demands for transparency in a supply chain.

6.2.2 Use Case 2: Electronic Health Records (EHRs)

Electronic health records (EHRs) are collections of patient medical records. They contain clinical data such as blood type, vital signs, past medical records, medications, and radiology reports for patients.[2] Currently, these records are often maintained by specific healthcare providers over time, in siloed systems not connected to other EHRs.

Multiple parties including patients, professionals, and organizations from different medical jurisdictions are involved in data exchange to allow more efficient healthcare and research. Healthcare service providers are decentralized trusted authorities. Each has access to patient data and has the authority to make the changes to that data. The operation of EHR systems is often distributed across healthcare service providers. Data transparency remains one of the main issues in existing EHRs. Patient privacy is critical, and normally information should only be shared with patient consent. Sometimes exceptions are made, for example, to access medical records in emergency situations, or to allow access to anonymized data for approved medical research. Accesses made to EHRs are often required to be logged for audit purposes. In addition to tight controls on read access, it is also important that health records cannot be inappropriately created or updated. EHRs do not typically need very low latency updates, and most patients' records do not change often. However, sometimes large diagnostic image information needs to be managed for an EHR.

Because of privacy constraints, blockchains are not normally used to store patient records directly, even in encrypted form. Instead, conventional systems are used to manage EHR source data, with blockchains providing auxiliary services. One example is the use of blockchains to keep audit logs of accesses made to EHRs. Records in these audit logs are typically encrypted or hashed to maintain patient privacy. MedRec[3] is an initiative to explore on blockchain architecture in contributing to secure and interoperable EHR systems. MedRec stores a pointer to

[2]https://www.cms.gov/Medicare/E-Health/EHealthRecords/index.html.

[3]https://medrec.media.mit.edu/.

patients' data in the blockchain and allows patients to choose when and with whom to share their data.

6.2.3 Use Case 3: Identity Management

Identity management underlies most business and social interactions. Individuals, organizations, devices, and assets can be identified by many schemes such as passports, wedding certificates, serial numbers, and registration certificates. An identity management system (IDM) manages user identities within an enterprise system. Conventionally, the operations of such systems are centralized and managed by a trusted authority. The authority sets permissions and roles for users to ensure they only access parts of the system relevant to them. Integrity is critical for IDM, to allow only authorized updates to users and their authorizations. Authorization can be complicated by requirements for delegated authorization and by requirements to enable dynamic revocation of authorizations. Logs of system accesses are often required, to be able to audit and investigate proper use of the system. Read accesses to an IDM can be frequent, to confirm authorized access, but updates to information in an IDM are normally much less frequent. It is often acceptable for there to be some delay in propagating updates to information about user identities and their authorizations.

Blockchain has been trialled for the management of individuals' identity for authorization, authentication, user role, and privileges within enterprise systems.[4,5] Blockchain allows the roles, permissions, and privileges of users to be verified by the distributed peers connected to the blockchain network. This removes the need for a centralized administrator and centralized database. Data on blockchain is transparent to everyone on the network by default. The immutable transaction history is duplicated to all connected peers. IDMs on a blockchain ensure that user identities, roles, and authorizations will not be altered improperly. Despite the fact that most current blockchains' performance does not match that of existing systems, it can still be viable to implement IDMs on blockchain because most operations require read access, which can have low latency for blockchains. Privacy is a critical requirement for IDMs, and so plaintext identity information for users is not normally stored directly on a blockchain. Instead, that is either kept off-chain or perhaps encrypted on-chain. For any solution, a significant privacy concern for system designers must be the possibility of reidentification attacks that may allow identities to be inferred from metadata or relationships stored on the blockchain.

[4]https://www.ibm.com/blogs/blockchain/2017/05/its-all-about-trust-blockchain-for-identity-management/.

[5]https://letstalkpayments.com/22-companies-leveraging-blockchain-for-identity-management-and-authentication/.

6.2.4 Use Case 4: Stock Market

A stock market is a place where stocks, bonds, and securities are traded. A stock market system inherently involves multiple entities to issue and trade stocks and conventionally is implemented by a centrally controlled and maintained register of stock ownership. In most jurisdictions, regulatory approval is required for the operation of stock market infrastructure, and regulatory approval may be required for the trading of specific stocks. In those contexts, the stock market is a natural trusted authority. Integrity, immutability, and non-repudiation are critical to ensure that high-value trades cannot be undone by either party. Transaction history is important in providing evidence for trades and current stock holdings. Stock markets typically have a high-volume, extremely low-latency price-setting mechanism to match buyers and sellers. However, stock markets typically settle trades (i.e. exchange the stocks and payment) at a later time. Settlement can have high throughput requirements but typically does not have extreme latency requirements.

Blockchain technology allows trades to be settled by the blockchain infrastructure using peer confirmation, removing the need for centralized operation and centralized authority to verify trades. Data transparency, however, is an issue for blockchains in the context of the stock market. All investors and market participants are exposed to blockchain participants. Even in a consortium blockchain between brokers, this creates a disadvantage to the investor and may be prohibited by a regulator. Transaction history is important because it keeps track of the ownerships of shares and also any changes that happen. Data immutability is also crucial as it ensures that no successful transactions can be tampered with by anyone. Looking at the scalability of existing stock exchanges, blockchain technology might not be suitable for this use case until the performance of blockchain can match up with current conventional technologies. Overall, blockchain is not highly suitable for the operation of conventional regulated stock markets. However, some blockchain solutions are being explored. NASDAQ offers its Linq blockchain ledger for registration and settlement of private securities,[6] and the Australian Stock Exchange (ASX) is also exploring distributed ledger technology to replace their current Clearing House Electronic Subregister System, for core modules such as trade registration and settlement.[7]

6.3 Design Process for Blockchain-Based Systems

In this section, we discuss an indicative model for the design of systems that might use blockchain technology. The process is shown in Fig. 6.2. Every step in the process is a procedure to decide between alternative options. The available

[6]http://ir.nasdaq.com/releasedetail.cfm?releaseid=948326.

[7]http://www.asx.com.au/services/chess-replacement.htm.

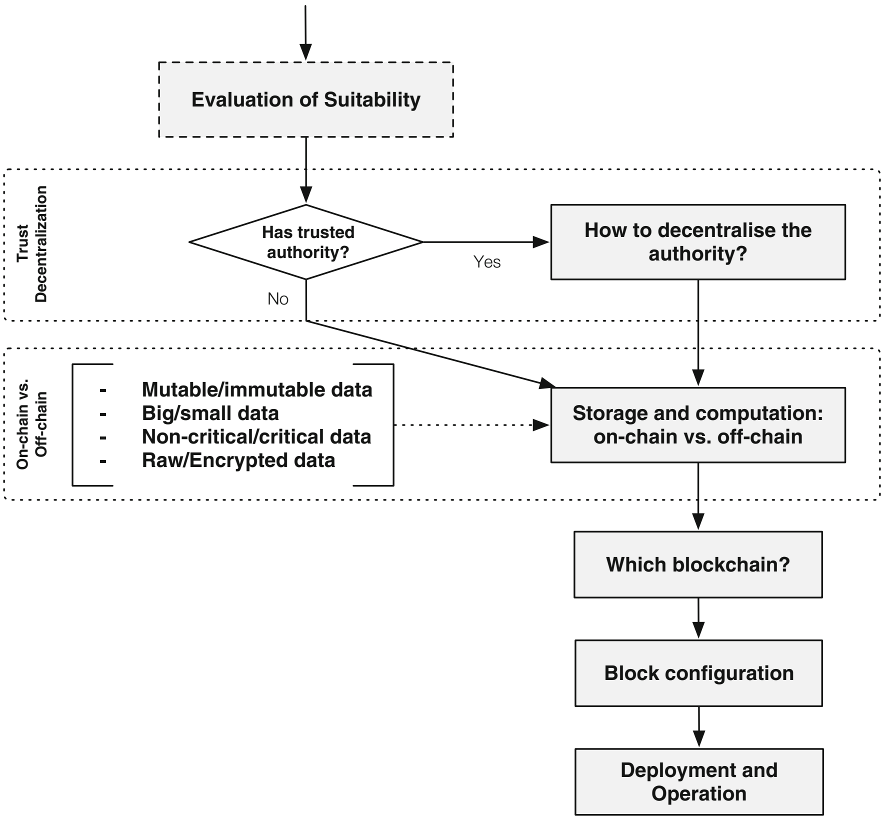

Fig. 6.2 Design process for blockchain-based systems. © 2017 IEEE. Reprinted, with permission, from Xu et al. (2017)

options discussed in Chapter 3 are used to assist decision-making and to guide the system design at different stages of the design process. This enables a systematic comparison of the capabilities of different design options. Chapter 3 describes the impact of the design options on quality attributes. Trade-off analysis between affected quality attributes is the foundation for the comparison of design options. The design process starts after the initial evaluation of blockchain suitability. The arrows illustrate one possible sequence of design decisions.

6.3.1 Trade-Off Analysis

As with any software system, there are trade-offs between quality attributes in the design of blockchain-based systems. Some decisions mainly affect scalability (like block size and frequency), security (like consensus protocol), cost efficiency (like type of blockchain), or performance (like data structure). Design decisions that improve the performance of one quality attribute for a system may harm the performance of other quality attributes. Some simple examples of this include:

- Encrypting data before storing it on a blockchain may increase confidentiality, but will reduce performance, and may harm transparency or independent auditability.
- Storing only a hash of data on-chain and keeping the contents off-chain will improve confidentiality and may improve performance but partly undermines the distinctive benefit of blockchains in providing distributed trust. This may create a single point of failure, reducing system availability and reliability.
- Using a private blockchain instead of a public blockchain may allow greater control over the admittance of processing nodes and transactions into the system but will also increase barriers to entry for participation and thus partly reduce some of the benefit of using a blockchain.
- For blockchains that use Nakamoto consensus such as Bitcoin or Ethereum, waiting for a higher number of confirmation blocks may increase confidence in integrity and durability of transactions but will harm latency and thus may impact service availability.

6.3.2 Decentralization

According to the discussion in Section 6.1, a blockchain is used in scenarios where no single trusted authority is required or acceptable and where the trusted authorities can be decentralized or partially decentralized. For the deployment and operation of systems, there is a spectrum of options ranging from centralized monopolies to central parties with a competition between parties, to services provided jointly by a consortia, through to fully open service provision in a public peer-to-peer system. It is possible that some components or functions are decentralized while others are centralized. Design decisions regarding trust decentralization are discussed in Section 3.2.

6.3.3 On-Chain vs. Off-Chain

Blockchains are usually combined with other components in a broader system. Functionality such as user interfaces, cryptographic key management, IoT integration, and communication with other external systems is inherently off-chain. Many kinds of data are also better stored off-chain, for scalability reasons (big data), for confidentiality reasons (private data), or for dealing with legacy databases. Although we say ‘big data’ is not suitable for storing on a blockchain, even ‘not tiny’ data may be too large to feasibly store on a blockchain. Cost calculations can help to determine the resolution of design decisions for this issue (see also Chapter 9).

While blockchains provide some unique properties, the amount of computational power and data storage space available on a blockchain network remains limited. In addition, the monetary cost of using public blockchains follows a different cost model than conventional software systems. In regard to cost efficiency, performance, and flexibility, major design decisions in using a blockchain include choosing what data and computation should be placed on-chain and what should be kept off-chain. Table 6.2 captures some of these options, which are described in more detail below.

Data

A common practice for data management in blockchain-based systems is to store raw data off-chain and to store on-chain just metadata, small critical data, and hashes of the raw data. However, the applications of storing item data on blockchain are not just for integration with external data. There are various uses for wholly on-chain auxiliary data, including 'colored coins' which are a class of overlays on Bitcoin to represent and manage real-world assets.

A detailed discussion of on-chain data storage cost can be found in the respective cost chapter, in Section 9.1. Here we focus on a higher-level consideration as part of the design process.

In the Bitcoin blockchain, there are different ways to store data in transactions. This was not a core feature in the original design of Bitcoin but has now been incorporated with a specific command, called *OP_RETURN*. Table 6.2 compares this mechanism with alternatives. While it offers some level of flexibility, storing data on the Bitcoin blockchain is slow and costly and limited to 40 bytes.

Ethereum, on the other hand, theoretically allows storing arbitrary structured data of any size in a transaction directly. However, the size of a transaction is limited by the maximum size of a block, and in practice transactions typically need to be smaller to be accepted due to the transaction load from other users. In addition, Ethereum provides two other ways to store arbitrary data, using smart contracts. The first option is to store the data as a variable in a smart contract. The second option is to store arbitrary data as a log event of a smart contract. Storing data as a variable in a smart contract is more efficient to manipulate, but less flexible due to the constraints of the Solidity language on the value types and length. The flexibility and performance of using smart contract log events is intermediate because log events allow up to three parameters to be queried.

Finally, we reiterate that data storage on blockchain follows a different cost model than conventional data storage. Although it may seem more expensive, storing data on blockchain is a one-time cost for permanent storage. (However, note that Ethereum allows a partial refund on reclaimed smart contract variable storage.)

Selection of off-chain data storage concerns the interaction between the blockchain and the conventional data storage facilities. Off-chain data storage can be through conventional enterprise IT systems, a private cloud on the client's infrastructure, or a public storage provided by a third-party. The flexibility of using cloud to store data depends on the implementation. Some peer-to-peer data storage

Table 6.2 Design decisions regarding storage and computation with an indication of their relative impact on quality properties (⊕, Least favourable; ⊕⊕, Less favourable; ⊕⊕⊕, More favourable; ⊕⊕⊕⊕, Most favourable)

			Impact			
Design decision		Option	Fundamental properties	Cost efficiency	Performance	Flexibility
Data	On-chain	Embedded in transaction (Bitcoin)	⊕⊕⊕⊕	⊕	⊕	⊕⊕
		Embedded in transaction (Public Ethereum)		⊕⊕⊕⊕	⊕	⊕⊕⊕
		Smart contract variable (Public Ethereum)		⊕⊕	⊕⊕⊕	⊕
		Smart contract log event (Public Ethereum)		⊕⊕⊕	⊕⊕	⊕⊕
	Off-chain	Private/third-party cloud	⊕	~KB negligible	⊕⊕⊕⊕	⊕⊕⊕⊕
		Peer-to-peer system		⊕⊕⊕⊕	⊕⊕⊕	⊕⊕⊕
Computation	On-chain	Transaction constraints	⊕⊕⊕⊕	⊕	⊕	⊕
		Smart contract				
	Off-chain	Private/third-party cloud	⊕	⊕⊕⊕⊕	⊕⊕⊕⊕	⊕⊕⊕⊕

© 2017 IEEE. Reprinted, with permission, from Xu et al. (2017)

facilities are designed to be friendly to blockchain, such as IPFS[8] and Storj.[9] IPFS is free, but ensuring availability requires providing an IPFS server that hosts the data. The cost of Storj is US$0.015/GB/month. In a peer-to-peer data storage, the data is replicated automatically by the peer-to-peer network or based on the behaviour of users, e.g. data is replicated once a user accesses it. In a cloud environment, data replication needs to be managed by the system or consumer.

Computation

Computation in a blockchain-based system can be performed on-chain (e.g. through smart contracts) or off-chain. Different blockchains offer different levels of expressiveness for on-chain computation. For example, Bitcoin only allows simple scripts and conditions that must be satisfied to transfer Bitcoin payments. Ethereum allows more general (Turing complete) programs, and these programs can not only perform conditional payments but also make modifications to the working data in smart contract variables. There are other smart contract languages which are more expressive than Bitcoin's simple scripts, but which are purposefully not Turing complete, in order to facilitate static analysis. An example is the Digital Asset Modelling Language (DAML),[10] which is designed to codify financial rights and obligations.

Smart contracts are not processed until their invoking transactions are included in a new block. Blocks impose an order on transactions, thus resolving nondeterminism which might otherwise affect their execution results. One benefit of using on-chain computation, rather than using blockchain as a data layer only, is the inherent interoperability among the systems built on the same blockchain network. Other benefits are the neutrality of the execution environment and immutability of the program code once deployed. This facilitates building trust in the shared code among untrusting parties.

Other Considerations

Deciding between on-chain and off-chain not only depends on trade-offs among quality attributes, but also on how information and computation are used by other components in the broader system. Take identity information (Section 6.2.3) as an example. Identity supports systems where there is a requirement to know the individual human or system involved in transactions. Services such as international payments have regulatory requirements to establish the identity of participants, as part of Anti-Money Laundering (AML) and Counter-Terrorism Financing (CTF)

[8]https://ipfs.io/.

[9]https://storj.io/.

[10]https://digitalasset.com/press/introducing-daml.html.

policies. From a purely technical perspective, real-world identities are not necessarily required. For example on Bitcoin, transacting agents (which are not necessarily persons) are only cryptographically identified, pseudonymously. So international exchange of the Bitcoin digital currency can be performed without establishing real-world identity. Nonetheless, AML/CTF requirements are not obviated by the use of a blockchain. Identity is critical here, and identity on blockchain is sometimes considered to be a key enabler for many financial services on blockchain. However identity information does not necessarily need to be stored on-chain, off-chain protocols might be used instead. Privacy and confidentiality can be a challenge when integrating identity information into a blockchain-based system.

6.3.4 Blockchain Selection and Configuration

At this stage, a blockchain platform is selected according to the requirement of the use case and characteristics of blockchain platforms and trade-off analysis discussed in Chapter 3. Normally, the consensus protocol and some other decisions are fixed once a particular blockchain is selected. Hyperledger Fabric is an exception, where a modular architecture is used to support pluggable implementations of various consensus protocols. For some blockchain platforms, for example, those using a proof-of-work protocol, the inter-block time can be configured through adjustments to the difficulty of mining.

6.3.5 Deployment and Operation

Finally, the choice of where to deploy the modules of the blockchain-based system is also important for the quality attributes of blockchain-based systems. For example, deploying a blockchain on a cloud provided by a third-party, or using a blockchain-as-a-service model directly, introduces the uncertainty of cloud infrastructure into the system. Here the cloud provider becomes a trusted third-party and a potential single point of failure for the system. Deploying a public blockchain system on a virtual private network can make it a private blockchain, with permissioned access controls provided at the network level. However the virtual private network will introduce its own additional latency overhead.

There are specific design challenges related to the operation of blockchain-based systems, which architects should be aware of when deciding to use a blockchain. Blockchain-based systems can be harder to modify than conventional systems. The blockchain platform software runs on multiple independently operating nodes, and updating that software can be physically and administratively difficult to coordinate. The blockchain ledger is also immutable by design and so cannot be retrospectively updated to facilitate system modification. Similarly, in blockchain-based systems that use smart contracts to regulate interactions between mutually untrusting parties, trust is derived partly from the fact that the code cannot be changed easily.

This inherently creates challenges for governance: the management of the evolution of blockchain-based systems. Changes may be made to correct defects, add features, or migrate to new IT contexts. However, in a multiparty system with no single owner, managing these changes is more like diplomacy than traditional risk management or conventional product management. Hence, the current configuration of blockchain is not suitable to implement on a system that may need to change or be modified frequently. Lessons may be drawn from governance in open-source software, which faces similar development challenges. However, the governance of a blockchain is not just a software development problem—it is also a deployment and operations problem. For both public and private blockchain systems, key stakeholders include the users of the blockchain, software developers with moral or contractual authority over the code base, miners or processing nodes in the blockchain ecosystem, and government regulators in related industries. However, blockchain immutability may also simplify governance oversight to some degree. For instance, smart contracts deployed on a blockchain will be resistant to tampering and will continue to be individually available for execution while the whole blockchain operates normally. These factors should be taken into consideration when deciding to use blockchain as a component.

6.4 Summary

Due to their fundamental properties and limitations, blockchains do not fit to all scenarios. Thus, before designing a system, the suitability of a blockchain needs to be evaluated against the system requirements. This chapter started with a suitability framework for assessing the suitability of using blockchain in a various contexts, based on the characteristics of the use case. After the suitability framework, a general process for designing blockchain-based applications was discussed. Throughout this design process, the available options discussed in Chapter 3 are used to assist decision-making and to guide the system design at different stages of the design process, by enabling a systematic comparison among the capabilities of different design options.

6.5 Further Reading

This chapter is partly based on our earlier works (Xu et al. 2017; Lo et al. 2017).

MedRec is an initiative to explore how a blockchain-based architecture can contribute to secure and interoperable EHR systems. More details of MedRec can be found in Azaria et al. (2016).