A Decision Tree for Building IT Applications

Nour El Madhoun, Julien Hatin, Emmanuel Bertin

Noname manuscript No.
(will be inserted by the editor)

A Decision Tree for Building IT Applications

What to choose: Blockchain or Classical Systems ?

Nour El Madhoun · Julien Hatin · Emmanuel Bertin

Received: date / Accepted: date

Abstract Blockchain technology has gained an increasing attention from research and industry over the recent years. It allows to implement in its environment the smart-contracts technology which is used to automate and execute agreements between users. The blockchain is proposed today as a new technical infrastructure for several types of IT applications. This interest is mainly due to its core property that allows two users to perform transactions without going through a TTP, while offering a transparent and fully protected data storage. However, a blockchain comes along a number of other intrinsic properties, which may not be suitable or beneficial in all the envisaged application cases. Consequently, we propose in this paper to design a new tool which is "a decision tree" that allows to identify when a blockchain may be the appropriate technical infrastructure for a given IT application, and when another classical system (centralized or distributed peer-to-peer) is more adapted. The proposed decision tree allows also to identify whether it is necessary to use the smart-contracts technology or not.

Keywords · Blockchain · IT · permissioned · peer-to-peer · permissionless · security · smart-contracts · TTP.

1 Introduction

The Bitcoin application was the first implementation of the blockchain technology in 2009 in the field of cryptocurrencies [1]. Since its creation, the popularity of bitcoins has increased in a very remarkable way because customers have appreciated the security of this application which is guaranteed thanks to the blockchain technology. Indeed, this security comes firstly from the unique property of blockchain that allows two users to perform transactions without going through a Trusted Third Party (TTP), and secondly from the fact that the history of all these transactions is stored in a distributed and an immutable manner among blockchain users [2]. Consequently, blockchain technology has received an increasing attention in several other areas such as healthcare, insurance, data verification, etc. and it is proposed for implementation as a new security infrastructure for several types of Information Technology (IT) applications.

In order to develop an IT application, the implementation of a blockchain infrastructure is different from that of a traditional system: centralised or distributed peer-to-peer. In a centralised system, the application users can communicate together in a rapid way thanks to the powerful central authority TTP [3]. The distributed peer-to-peer system is generally used to share files between the application users (peers) without the need for a TTP and without any security layer [4]. The blockchain infrastructure is based on the same principle of the distributed peer-to-peer system where users can communicate together without the intervention of a central TTP. However, they are different in the execution environment and the security specifications, as well as the blockchain allows to implement in its environment the smart-contracts technology that are used to automate agreements between users [5].

Indeed, the *"non-intervention of a central TTP during communication between users"* represents *the intrinsic main property of a blockchain.* In addition, the blockchain comes along a number of other intrinsic properties which may not all be suitable or beneficial for all types of IT applications, while a classical "centralized or distributed peer-to-peer" system may be more appropriate [6]. Consequently, in this work, we are interested in proposing a decision tree identifying whether a blockchain is the best solution for a given IT application, or a classical system is more adapted. Our proposal allows also to identify whether it is necessary to use the smart-contracts technology or not and it relies on a comparative study between the blockchain and classical systems.

This paper is organized as follows. In section 2, we introduce an overview of classical systems, and in section 3, we present a background on the blockchain technology. In section 4, we compare between the three infrastructures: blockchain,

N. El Madhoun
LISITE Laboratory, ISEP, 10 Rue de Vanves 92130 Issy-les-Moulineaux, France
E-mail: nour.el-madhoun@isep.fr

J. Hatin
Orange Labs, 42 rue des Coutures BP 6243 14066 Caen, France
E-mail: julien.hatin@orange.com

E. Bertin
Orange Labs, 42 rue des Coutures BP 6243 14066 Caen, France
E-mail: emmanuel.bertin@orange.com

centralized and distributed peer-to-peer, and in section 5, we describe our proposed decision tree. In section 6, we review a selection of the related works and we compare them to our proposal. In the last section, we provide a brief conclusion and we discuss possible future work. We note that all our abbreviations are illustrated in Tab-1.

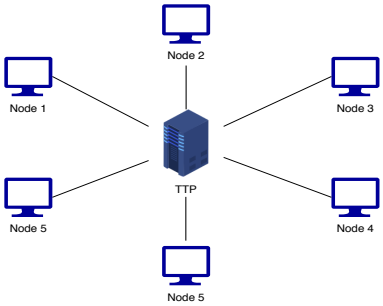

Fig. 1: Centralized system

2 Overview of Classical Systems

In order to develop an IT application, we typically have the choice between a centralized (client/server) system and a distributed peer-to-peer system. This choice is based on the needs of the IT application and the characteristics of each system.

2.1 Centralized System

It is called also client/server system because everyone depends on the same central authority which is the server TTP (see Fig-1). The nodes (clients or peers or users) of this system can communicate together in a rapid way thanks to the centralized TTP server. The characteristics of this system can be summarized as follows [3] [7] [8] [9] [10] [11]:

Table 1: Abbreviations

Abb.	Description
Ui	User (Peer or Node or Client) *i* (*i=1,2,..*)
B	Block
L	Ledger
T	Transaction
F	File
PK(Ui)	Public Key of a User *Ui*
SK(Ui)	Secret (Private) Key of a User *Ui*
H(T)	One way hashing function of *T*
Hpow(B)	One way hashing function of *B* generated with the PoW
SignUi(T)	Electronic Signature of *T*. It is generated thanks to *SK(Ui)*

1. *Type of Application*: by default, in a centralized system, any type of application can be implemented such as: websites, banks, cloud, etc. However, the implementation must be adapted according to the needs of the application such as adding a cryptographic layer if the transactions are sensitive (see in this section the property *4. Cryptographic Layer*).
2. *Intrinsic Properties*: these are actually the properties that come by default with a basic implementation for a centralized system.
   2.1 *Centralization*: the TTP represents the main center of management and trust for all nodes. It is a powerful authority which has a large calculation capacity and is able to process multiple requests at the same time. It is the primary database of the system and all requests go through the TTP as we illustrate in the example of Fig-2: Alice which wants to transfer 10€ to Bob, sends her request to the central bank (TTP) which in turn verifies Alice's account, validates the transaction and transfers to Bob the 10€. Consequently, the TTP stores the history of transactions in its database. The disadvantage of centralization is that the TTP represents a *Single Point of Failure* and therefore if it breaks down, the whole system breaks down.

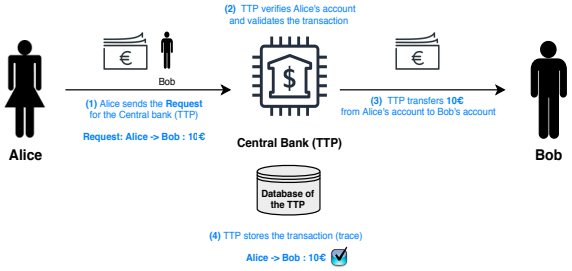

Fig. 2: Example of a bank transfer

   2.2 *Response Time Versus Loading Time*: in a centralized system, the response time to a request is very fast and the system adapts systematically to the loading time if there is a large volume of requests. This advantage is due to the powerful TTP that supports several management mechanisms. Consequently, we can guarantee with a centralized system that the response time always remains a fast option even if the TTP is overloaded.
3. *Data Replication*: it is a strategy used to achieve a high level of availability and fault tolerance, to protect the data and to ensure a better scalability. In a centralized system, the duplication of the TTP server depends on the needs of the application, so it may be used or not.
4. *Cryptographic Layer*: the default implementation of a centralized system does not provide any security layer. This means that all communications via the TTP may be observed and read by an attacker. So, for this reason, if the application transactions in a centralized system are sensitive and must be secured, then it is mandatory to add a cryptographic layer to this system in order to secure it and ensure security properties (such as integrity, confidentiality, authentication, etc.).
5. *Data Transparency*: in most centralized systems, the users cannot access to all the information stored on the centralized database. They cannot also consult the list of operations previously carried out in the system. Indeed, it is very complicated and difficult to implement the property of data transparency in a centralized system and for this reason, we will consider in this paper that it is not possible to do it in this type of system.
6. *Data Immutability*: it means not being able to change the information stored in the database. In most centralized systems, databases are generally editable where the administrator can change, add and update any information. Indeed, the idea of making this database non-modifiable forever is a very difficult objective to achieve and for this reason, we will consider in this paper that it is not possible to do it in this type of system.

2.2 Distributed Peer-to-Peer System

As illustrated in Fig-3, there is no centralized TTP server in a distributed peer-to-peer system. The nodes of this system can communicate together without the need for a TTP.

2.2.1 Characteristics

The characteristics of this system can be summarized as follows [4] [12] [13] [14] [15]:

1. *Type of Application*: the distributed peer-to-peer system is generally used for applications such as file sharing, streaming, updating software, etc.

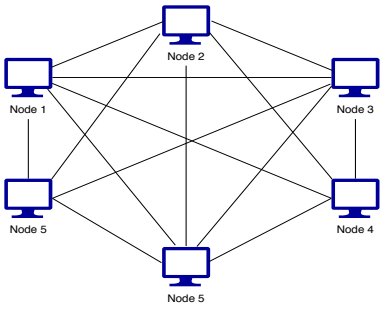

Fig. 3: Distributed peer-to-peer system

2. *Intrinsic Properties*: these are actually the properties that come by default with a basic implementation for a distributed peer-to-peer system.
   - 2.1 *Distribution (decentralization)*: in this system, there is no centralized TTP server and then no centralized database. However, each peer has its own database which stores a list of data to share (on demand) with other peers. The sharing is in fact in the form of distribution and reception of data between peers. In this type of system, comparable to the client/server system, each peer has the role of a client and a server as we illustrate in the following example of file sharing: a peer *U1* who has a file *F* can make it available to other peers *Ui* via a download platform "*U1* is therefore a server and the *Ui* are clients". In addition, *U1* can simultaneously download other files shared by other peers *Uj* "*U1* is therefore a client and the *Uj* are servers".
   - 2.2 *Data Replication*: in this type of system, the replication of data is needed to encourage the existence of the same data in several peers. However, it is not advantageous to implement a peer-to-peer system without data replication because the replication is considered an intrinsic property in this system.
   - 2.3 *Data Transparency*: each peer in this system can see the databases of other peers and the operations previously performed between the pairs. It is a native property if we want to implement a distributed peer-to-peer system.
3. *Cryptographic Layer*: as in the centralized system, the default implementation of a distributed peer-to-peer system does not provide any security layer. This means that all communications between peers may be observed and read by an attacker. So, if the application transactions are considered sensitive then a cryptographic layer is necessary for the implementation.
4. *Response Time Versus Loading Time*: indeed, the peers in the distributed system do not have a large calculation capacity and they may have overload situations leading to slower communications: time is more important. Consequently, this type of system does not guarantee the characteristic of the *"response time is always a fast option even in the case of network overload"*.
5. *Data Immutability*: each peer has its own database that wishes to share it or not in this type of network. Indeed, there is no mechanism in this system that prevents changing data and record all the information forever.

2.2.2 Types of distributed peer-to-peer system

We can find three types of a distributed peer-to-peer system [4] [16]: 1. a public peer-to-peer system where everyone can access to the application, 2. a private peer-to-peer system where a specific group of users can access to the application, 3. a trusted peer-to-peer system where a cryptographic layer is needed for implementation and only a specific group of users which can access to the secured application.

3 Blockchain Technology: Synthesis

The blockchain infrastructure is based on the same principle of a distributed peer-to-peer system where the peers can communicate together without going through a TTP. However, they are different from each other in some specifications and the execution environment.

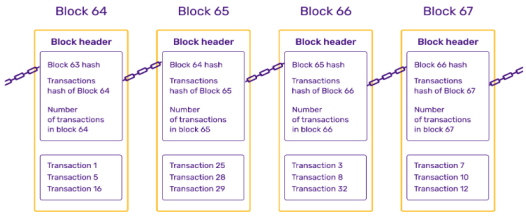

Fig. 4: Ledger of a blockchain [17]

3.1 Characteristics

In this section, we will describe the characteristics of the blockchain while illustrating the main differences with the distributed peer-to-peer system as well as the centralized system [18] [19]:

1. *Type of Application*: the classical peer-to-peer system is mainly designed to share files between the application users whereas the blockchain infrastructure is mainly designed to implement and execute trusted secure transactions (see the Definition) between the application users such as that based on cryptocurrencies and smart-contracts (see sections 3.4.1 and 3.3) [8] [11].

   Definition "Trusted Secure Transactions":
   They require sensitive computing in a secure and trusted environment. For more illustrations, we return to the example shown in Fig-2. The operation of transferring a sum of money from Alice's account to Bob's account is a very sensitive operation and it could be guaranteed through the secure TTP (central bank). Therefore, this type of operation is considered in the

blockchain technology a Trusted Secure Transaction (see section 3.4.1). In addition, any type of agreement between two parties such as contracts is also considered in the blockchain technology a Trusted Secure Transaction (see section 3.3).

2. *Intrinsic Properties*: these are actually the properties that come by default with a basic implementation for a blockchain.
   2.1 *Distribution (decentralization)*: it represents the main property of the blockchain technology and it means that the blockchain communications do not rely on a TTP. The data are stored in a distributed manner and the blockchain users communicate together without the need for a TTP and without the need they know or trust each other before. In fact, in the blockchain, there is no centralized TTP server, there is no centralized database and each peer does not have its own database. However, the database of the blockchain is represented in the form of a ledger L which does not store, as in the classical peer-to-peer system, a list of data to share (on demand) with other peers, but it stores in a tamper-proof and secure way the history of all the exchanges made between the users (peers).
      – The ledger L constitutes a chain of several blocks interconnected with each other and with the addition of new blocks, this chain increases more and more. Each block contains the history of one or more exchanges as well as other important information [16]. In addition, each user has a copy of L and there is a consensus algorithm ensuring that each user owns the same copy of L as the other users (see section 3.2).
   2.2 *Data Replication*: all the blockchain users have the same copy of the ledger and then the data are duplicated throughout the system.
   2.3 *Cryptographic Layer*: the security layer is indeed one of the intrinsic properties of blockchain technology which it is not the same case in conventional systems. For authentication purposes, each user in the blockchain has its own key pair (public/private) which is generated thanks to the Elliptic Curve Digital Signature Algorithm (ECDSA). For the integrity of transactions and blocks, hashing functions are used [20].
   • We show a simple example of how a blockchain works: when a user *U1* performs a transaction T with another user *U2*, the processes that will be executed are as follows [21]:
      – *U1* generates *SignU1(T)* by signing the hash *H(T)* using its *SK(U1)*. This signature guarantees the authentication of *U1* and the integrity of T.
      – Afterwards, *U1* broadcasts T and *SignU1(T)* to all the other users.
      – Each receiver verifies the authenticity of *U1* and the validity (integrity) of T using the *PK(U1)*.
      – All users participate together in the execution of a consensus algorithm in order to obtain the eligible hash of the block (see section 3.2).
      – Indeed, the user who first finds the eligible hash, creates the block B and broadcasts it to all the other users. B mainly contains the following elements [22]:
         · Data of T: depend on the type of the application (exchange money, execiution of contracts, exchange documents, etc.). Ideed, the *SignU1(T)* is included also in these data.
         · *Hpow(B)*: it is the eligible hash identifying the current B and ensuring its integrity.
         · *Hpow(previous B)*: it links the current B to the previous B. This link creates a chain of blocks (see in this section the property *2.5 Data Immutability*).
      – Finally, each user adds B to its copy of L by linking it to the previous block thanks to the hash *Hpow(previous B)*.
   • In Fig-4, we illustrate an example of a ledger of a blockchain. We conclude the following security properties ensured thanks to the blockchain:
      – *Data Integrity & Authentication of the Origin*: the eligible hash of the block allows to guarantee its integrity. The electronic signature of the transaction generated by the user allows to ensure the integrity of this transaction and to ensure the authenticity of the user.
   2.4 *Data Transparency*: each user of the blockchain can observe how blocks have been added over time: everything (transactions, messages, etc.) is transparent.
   2.5 *Data Immutability*: it is ensured thanks to the three following elements: the hash calculation for each block, the links between the blocks, the duplication of the ledger L for all users. Indeed, if we assume that there is an attacker which wants to try to change the contents of the block *Bi* for a user *Uj*, then he needs to change: the hash of *Bi*, the link with the next block *Bi+1*, the hash of *Bi+1*, the link with the next block *Bi+2*, the hash of *Bi+2*, ... until the last block of the copy of L of *Uj* . In addition, the attacker needs to repeat all these changing operations for all the other users. So, the information stored in the blocks are indeed reserved forever and cannot be changed unless an attacker can gather of more than 51% of the computational power network [16].
3. *Response Time Versus Loading Time*: in a blockchain infrastructure, we cannot guarantee the property of the *"response time is always a fast option even in the case of network overload"* because, as in a distributed peer-to-peer system, the peers do not have a large calculation capacity, they need to send data to all the other peers and especially with the blockchain infrastructure an additional complexity is added through the consensus algorithm (see section 3.2). For example, Bitcoin (see section 3.4.1) can only execute a seven transactions per second, while the Visa centralized system can execute more than fifty thousand transactions per second [23].

3.2 Consensus Algorithm

A consensus algorithm is a crucial element in a blockchain infrastructure because it is responsible for maintaining the security of the blockchain. This algorithm can be defined as the mechanism by which a blockchain network reaches a consensus. Indeed, since the blockchain does not depend on a central authority, the distributed peers must agree on the validity of the transactions and therefore this is where the consensus algorithms come in. By default, a consensus algorithm requires execution time in order to allow all the blockchain users to agree on the same block, ensure that the last block has been correctly added to the chain and protect the blockchain against malicious attacks. There are several types of consensus algorithms. The most common implementations are Proof of Work (PoW) and Proof of Stake (PoS). Each solution has its advantages and disadvantages when it tries to balance security, functionality and scalability [10] [24] [25].

3.2.1 Proof of Work (PoW)

It is also known as mining. It was the first consensus algorithm to be created. It is used by Bitcoin and many other crypto-currencies. The PoW algorithm is an essential part of the mining process. The Mining via PoW involves many hashing attempts, so more computing power, which means more testing per second. The PoW is in fact a data that is difficult to produce because it requires a time for consumption (ten minutes in average). The PoW is easy to check by the other users of the blockchain. The production of a valid PoW is an eligible hash which depends on a random process with a low probability, so that a lot of trial and error is required on average before a valid PoW is generated.

3.2.2 Proof of Stake (PoS)

The PoS consensus algorithm was developed in 2011 as an alternative to PoW. Although PoW and POS share similar goals, they also have some fundamental differences and features. Especially when validating new blocks. So, Whereas the PoW requires users to run the hash algorithms several times by adding a random alphanumeric string to the data in the block until the set's footprint is below a given threshold, the PoS asks the user to prove possession of a certain amount of cryptocurrency (their participation) to claim to validate additional blocks in the blockchain and to receive the reward.

3.3 DApps and Smart-contracts

In a Blockchain environment, the Decentralised Applications (DApps) are used to automate exchanges between users. A DApp is a program (written in a programming language) that facilitates, executes and enforces the negotiation or execution of an agreement using Blockchain technology. The main goal of these DApps is to allow two anonymous parties to do business with each other without the need for an intermediary. In addition, the instructions of a DApp work exactly as they were programmed, without any possibility of immobilisation, censorship, fraud or interference of third parties. A smart-contract is a DApp that runs on the Ethereum blockchain (see section 3.4.2). However, the term "smart contract" today replaced the term "DApp" for all blockchain applications because of the celebrity of the blockchain Ethereum. And so the term "smart-contracts" is equivalent to "DApps" [26] [27]. For these reasons, in this paper, we use the term "smart-contract" or "DApp" for the same meaning. Indeed, the implementation of DApps is considered a Trusted Secure Transaction (see the Definition in section 3.1) [28].

3.4 Examples of Existing Blockchains

3.4.1 First Blockchain Application (Bitcoin)

The idea of blockchain technology was introduced in 1991 by a group of researchers to time-stamp digital documents that could not be backdated or change their contents [29]. Then, it was not really used until Satoshi Nakamoto used this concept in 2009 to create the Bitcoin payment system [1] [30]. The latter was therefore the first application using the blockchain infrastructure. Bitcoin system is an application of cryptocurrencies allowing two persons to perform financial trusted transactions without passing through a TTP, and then without passing through a banking network [31] [32]. "bitcoin" with small letter is the name of the cryptocurrency of the Bitcoin (big letter) blockchain application. So, for a transaction T of exchanging *1 bitcoin* from the user *U1* to the user *U2*, the contents of the B are: Data of T (the sender *U1*, the receiver *U2* and the amount *1 bitcoin*, the signature *SignU1(T)*), the eligible hash *Hpow(B)*, the link to the previous block with its eligible hash *Hpow(previous B)*.

3.4.2 Ethereum

Ethereum is a distributed, open source, public IT platform based on blockchain technology. Ethereum's cryptocurrency is called 'Ether (ETH)'. In fact, since the introduction of blockchain technology and the Bitcoin system, one of the most remarkable innovations has been the introduction of smart contracts on Ethereum. Smart-contracts on Ethereum are written in Solidity (a Scripting language specially designed for Ethereum). Ethereum allows developers to model, secure and exchange whatever they can mathematically represent thanks to turing-completeness (turing-completeness) [11] [32].

3.4.3 EOS (Scalable Decentralized Apps Network)

EOS is a public blockchain using the functionality of smart-contracts and which plans to offer decentralized applications on a commercial scale. The EOS project aims to solve the problem of scalability of blockchains and their maintenance costs before Ethereum. Indeed, it promises to be able to process millions of transactions per second (millions of users simultaneously), provide zero transaction fees (free use), a better user experience for developers and better governance compared to the old blockchains, to optimize the load in computation so as to sequence and to parallelize the tasks. The EOS blockchain is under development and we cannot create or deploy applications on it [33].

3.5 Types of Blockchains

A blockchain infrastructure may be permissionless or permissioned. With a permissionless blockchain, any user can read or write at any time. With a permissioned blockchain, only a set of users which is allowed to write and read [34]. In addition, a blockchain may be public or private. In a public blockchain, each user is allowed to contribute in the validation of a block. In a private blockchain, all users are known and the validation of a block is done by a selected set of users [35]. Consequently, we can conclude the three main types of blockchain implementations: 1. Permissionless blockchain, 2. Public permissioned blockchain, 3. Private permissioned blockchain.

4 Comparison between the Blockchain & Classical Systems

In this section, we provide a comparative study between the three infrastructures: blockchain, centralized and distributed

Table 2: Summary on the characteristics of systems

System / Characteristic	Centralized	Distributed P2P	Blockchain
Types of application	Any type	Generally 'File sharing'	Trusted Secure Transactions
Centralization	√ (Intrinsic)	-	-
Distribution	-	√ (Intrinsic)	√ (Intrinsic)
Data Replication	√ (Depending on the application)	√ (Intrinsic)	√ (Intrinsic)
Cryptographic Layer	√ (Depending on the application)	√ (Depending on the application)	√ (Intrinsic)
Data Transparency	-	√ (Intrinsic)	√ (Intrinsic)
Data Immutability	-	-	√ (Intrinsic)
Response Time Versus Loading Time	Fast (Intrinsic)	More Important (Not Fast)	More Important (Not Fast)

peer-to-peer. This study is based on the characteristics of each infrastructure. So, as illustrated in Table 2 and in sections 2.1, 2.2.1 and 3.1:

- *Types of applications*: it may be any type for a centralized system, generally a file sharing application for a peer-to-peer system and trusted secure transactions for a blockchain technology.
- *Centralization*: it is an intrinsic property in the centralized system.
- *Distribution*: it is an intrinsic property in a distributed peer-to-peer system and the blockchain technology.
- *Data Replication*: it may be used or not in a centralized system whereas it is an intrinsic property in the peer-to-peer system and the blockchain technology.
- *Cryptographic Layer*: it may be used or not in a centralized system and a peer-to-peer system. However, it is an intrinsic property in the blockchain technology.
- *Data Transparency*: it cannot be used in a centralized system. However, it is an intrinsic property in the distributed peer-to-peer system and the blockchain technology.
- *Data Immutability*: it cannot be used in a centralized system and in a distributed peer-to-peer system. However, it is an intrinsic property the blockchain technology.
- *Response Time Versus Loading Time*: in a centralized system, the response time to a request is very fast (intrinsic property). However, we cannot guarantee this property in the distributed peer-to-peer system and the blockchain technology.

5 Proposed Decision Tree

In this section, we describe our proposed decision tree illustrated in Fig-5. According to the needs of the IT application, our proposal allows to identify which infrastructure is the best solution for the implementation. We also clarify that our proposal is based on the fact that all the needs of the IT application are known in advance. This means that a preliminary study of these needs is necessary.

5.1 Proposal Description

The design of our decision tree is based on the needs of the IT application that are directly related to the characteristics of classical systems (see sections 2.1 and 2.2.1) and blockchain technology (see section 3.1). As illustrated in Fig-5:

- Each *blue box* represents *a need (requirement)* for the IT application. We number the boxes from **(1)** to **(21)**. This numbering does not mean that the execution of the boxes is sequential but it is intended to facilitate the explication of our proposal.
- Each *green leaf* represents a *decision* for a type of infrastructure.
- Each *red leaf* represents an *impossible decision*. This means that the sequencing of the needs of the application cannot be carried out. We therefore recommend readers to conduct a new study of the application requirements.
- The *right side* represents the *true case "1"*.
- The *left side* represents the *false case "0"*.

- We note that the "Presence of TTP" is considered in our work as a decision result in a centralized system and not as a requirement for the IT application. We have aggregated the properties of a centralized system in "data replication" and "Response Time VS Loading Time". As presented in section 4, if the application needs or not for the data replication than a centralized system may be used, and if the application needs to ensure the property of the response time is a fast option than we absolutely need to choose a centralized system.
- We note that the needs given in (see Fig-5): box (2), box (5) and boxes (3), (6), (9) are considered as essential elements for making the result decisions in the best conditions according to our comparative study in section 4.

We now describe how our tree makes a decision of an infrastructure for a type of an IT application:

(1) We start by asking if the IT application needs the "Replication of Data":
   - If it does not need, then we decide to implement a centralized system. As presented in sections 2.2.1 and 4, it is not possible to implement a distributed peer-to-peer system or a blockchain without replicating the data. Consequently, the non-replication of data is better supported in the case of a centralized system. We also specify that whatever the case of the necessity of the other properties we will always arrive at a decision of a centralized system.
   - If it needs, then we go to the box (2).

(2) If the IT application does not need for a "Cryptographic Layer", then the implementation of a blockchain infrastructure is not necessary because the "Cryptographic Layer" is an intrinsic property in this infrastructure (see section 3.1). Therefore, we will have two choices: either a distributed peer-to-peer system or a centralized system. Otherwise, we go to the box (7).

(3) If the IT application does not need for the "Data Transparency", then the implementation of a distributed peer-to-peer system is not necessary because the "Data Transparency" is an intrinsic property in this system (see section 2.2.1). Therefore, we will have only the choice of a centralized system. Otherwise, we go to the box (5).

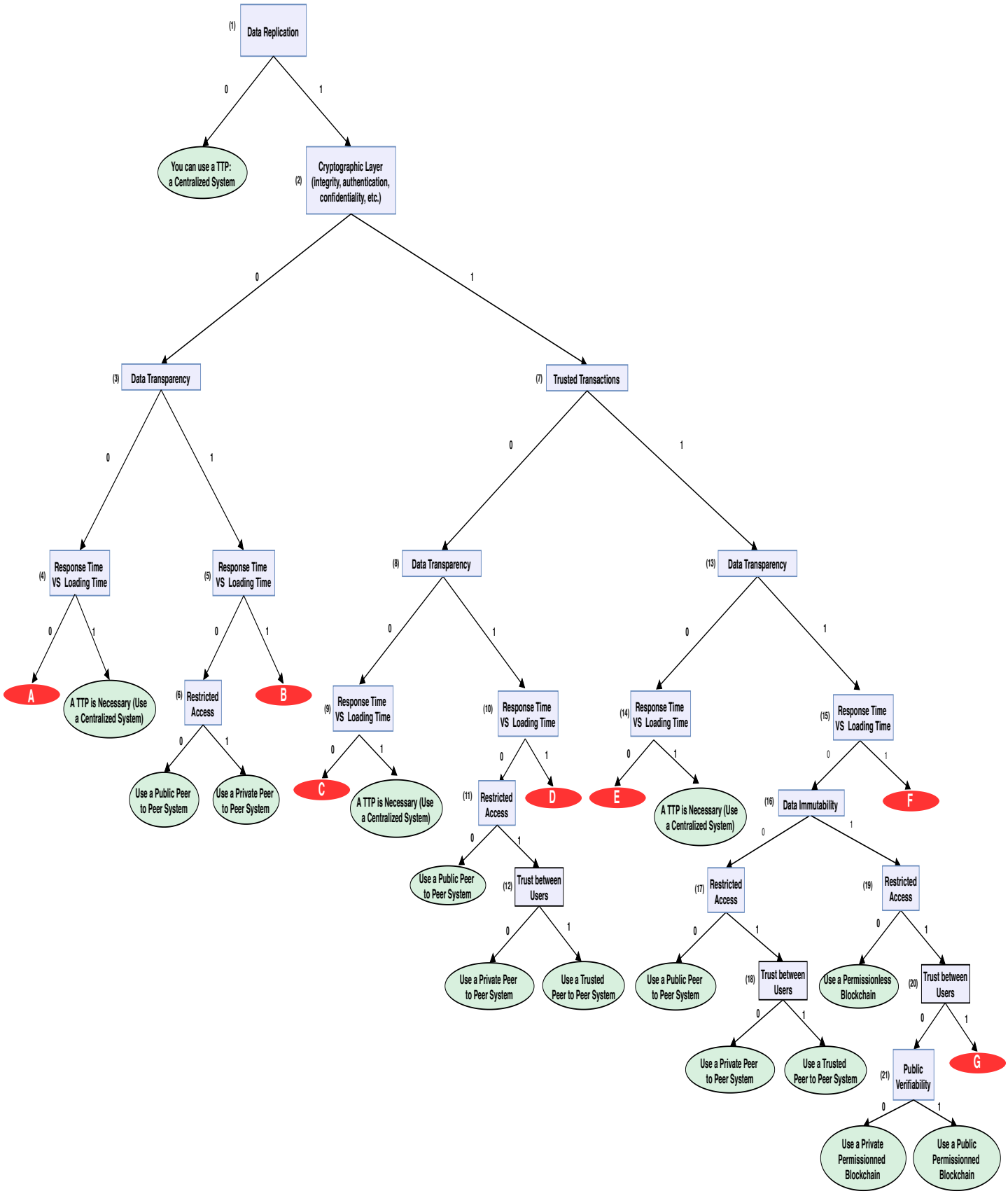

Fig. 5: Proposed decision tree

(4) As we presented in sections 2.1 and 4, the response time to a request is a fast option in a centralized system. Consequently, if the IT application needs to ensure this property, then we will choose to implement a centralized system. Otherwise, we don't have another solution and so an impossible decision.
(5) At this stage, if the IT application needs to ensure the property of *"response time is always a fast option even in the case of network overload"*, then this cannot be feasible because this property cannot be guaranteed with the choice of setting up the data transparency (see sections 2.1 and 4). Otherwise, we go to the box (6).
(6) If the application access is restricted then we can use a private peer-to-peer system, if not a public peer-to-peer can be used (see section 2.2.2).
(7) If the IT application does not need for the "trusted transactions", then the implementation of a blockchain infrastructure is not necessary because, as presented in section 3.1, the blockchain is principally designed to implement trusted secure transactions. In addition, if the IT application needs or not to execute "trusted transactions", we will ask the need of "Data Transparency" respectively in boxes (8) and (13).
(8) If the IT application does not need for the "Data Transparency", then the implementation of a distributed peer-to-peer system is not necessary (see boxes (3) and (4)). Therefore, we will have only the choice of a centralized system. Otherwise, we go to the box (10).
(9) If the IT application needs to ensure the property of *"response time is always a fast option even in the case of network overload"*, then we will choose to implement a centralized system (see box (4)). Otherwise, we don't have another solution and so an impossible decision.
(10) As in box (5), if the IT application needs to ensure the property of *"response time is always a fast option even in the case of network overload"*, then this cannot be feasible with the choice of setting up the data transparency. Otherwise, we go to the box (11).
(11) If the access to the application is not restricted, then we can use public peer-to-peer system. Otherwise, we go to the box (12).
(12) When the IT application needs to guarantee the requirement of box (2) and the application access is restricted, we need to ask the question about the trust between users (i.e: a group of friend). If the application needs that users trust each others, then we can use a trusted peer-to-peer system (see section 2.2.2). This system can be both secured and distributed but remains less expensive than a blockchain. Otherwise, we can use a private peer-to-peer system.
(13) As in boxes (3) and (8), if the IT application does not need for the "Data Transparency", then we have only the choice of a centralized system. Otherwise, we go to the box (15).
(14) As in boxes (4) and (9), the centralized system will be chosen if the IT application needs to ensure the property of *"response time is always a fast option even in the case of network overload"*. Otherwise, we don't have another solution and so an impossible decision.
(15) As in boxes (5) and (10), we don't have a solution if the IT application needs to apply the data transparency with the choice of the property of *"response time is always a fast option even in the case of network overload"*. Otherwise, we go to the box (16).
(16) At this stage, if the IT application does not need for the "Data Immutability", then the implementation of a blockchain infrastructure is not necessary because the "Data Immutability" is an intrinsic property in this infrastructure (see section 3.1). Therefore, we will have only the choice to implement a distributed peer-to-peer system. Otherwise, we go to the box (19).
(17) If the access to the application is not restricted, then we can use public peer-to-peer system. Otherwise, we go to the box (18).
(18) As in box (12), if the application needs that users trust each others, then we can use a trusted peer-to-peer system. Otherwise, we can use a private peer-to-peer system.
(19) If the application access is not restricted, then we can use a permissionless blockchain (see section 3.5). Otherwise, we go to the box (20).
(20) If the application needs the trust between users, then we don't have a possible decision because if we want to implement a blockchain then we need to be sure that the users do not know each other before and do not trust each other. Otherwise, we go to the box (21).
(21) As presented in section 3.5, the public verifiability allows any user to verify the correctness of the blockchain system. In the private verifiability, a set of specific users that can verify the state of the blockchain. Therefore, if the IT application needs the public verifiability, then we can use a public permissioned blockchain. Otherwise, a private permissioned blockchain is necessary. The permissioned is because the restricted access is true.

5.2 Cases of Non-Decision

In this section, we provide the explanation of *the cases of non-decision* presented in the *red leaves* in the tree (see section 5.1 and Fig-5):

- Case A: (1) true, (2) false, (3) false, (4) false: it an impossible decision because we cannot choose neither a centralized system without the need for a fast response time, nor a peer-to-peer system without the data transparency, nor a blockchain technology without a cryptographic layer and without the data transparency.
- Case B: (1) true, (2) false, (3) true, (5) true: it an impossible decision because we cannot choose neither a centralized system with data transparency, nor a peer-to-peer system with the need for a fast response time, nor a blockchain technology without a cryptographic layer and with the need for a fast response time.
- Case C: (1) true, (2) true, (7) false, (8) false, (9) false: it an impossible decision because we cannot choose neither a centralized system without the need for a fast response time, nor a peer-to-peer system without the data transparency, nor a blockchain technology without trusted secure transactions and without data transparency.
- Case D: (1) true, (2) true, (7) false, (8) true, (10) true: it an impossible decision because we cannot choose neither a centralized system with the data transparency, nor a peer-to-peer system with the need for a fast response time, nor a blockchain technology without trusted secure transactions.
- Case E: (1) true, (2) true, (7) true, (13) false, (14) false: it an impossible decision because we cannot choose neither a centralized system without the need for a fast response time, nor a peer-to-peer system without the data transparency, nor a blockchain technology without the data transparency.
- Case F: (1) true, (2) true, (7) true, (13) true, (15) true: it an impossible decision because we cannot choose neither a centralized system with data transparency, nor a peer-to-peer system with the need for a fast response time, nor

a blockchain technology with the need for a fast response time.
- Case G: (1) true, (2) true, (7) true, (13) true, (15) false, (16) true, (19) true, (20) true: it an impossible decision because we cannot choose neither a centralized system without the need for a fast response time, nor a peer-to-peer system with the need for the data immutability, nor a blockchain technology with the need for prior trust between users.

5.3 Discussions (Examples)

In this section, we provide some examples of IT applications aiming to discuss and validate our decision tree. Indeed, we consider in our work that these examples need to be executed in a new environment and therefore we do not consider their current implementation in existing environments. In addition, we point out that our tree helps not only to decide which type of infrastructure is the best suited, but also to eliminate cases where the blockchain is not necessary. This means that if the decision result is not a blockchain by executing our tree on an example of an IT application, so surely this IT application does not require using a blockchain. However, if the decision result is a blockchain, this does not mean that the IT application should use a blockchain but means that it can use a blockchain if there are no other external factors that can influence this decision. In this paper, we did not take into consideration any external factor such as state, legal, regulation, business model, etc., but we only took into account technical and scientific needs. Consequently, it is necessary for the engineer to address these factors before executing our decision tree. For example, if we are in a sector or in a state which prohibits blockchain, in this case the engineer will not be able to take into consideration the results of our tree.

* **A notarial IT application:** it is an application for writing notarial contracts. By running our tree, we obtain:
  - The application needs: Data Replication in (1), Cryptographic Layer in (2), Trusted Transactions in (7), Data Transparency in (13).
  - The application does not need to respect that the response time is a fast option in (15).
  - The application needs: Data Immutability in (16).
  - The application access is not restricted in (19).
  - (1) true, (2) true, (7) true, (13) true, (15) false, (16) true, (19) false –> *Use a Permissionless Blockchain.* Currently, notarial applications go through a complicated process and several centralized systems to provide a home buying service for example. Blockchain is an interesting technology to decentralize communications and guarantee the fluidity of services if there are no other factors that can prevent this decision.
* **Rental application "owner-tenant":** it is an application to rent housing between owners and tenants. By running our tree, we obtain:
  - The application needs: Data Replication in (1), Cryptographic Layer in (2), Trusted Transactions in (7), Data Transparency in (13).
  - The application does not need to respect that the response time is a fast option in (15).
  - The application needs: Data Immutability in (16).
  - The application access is restricted (a set of owners) in (19).
  - The trust between users is not needed in (20).
  - The public verifiability is not needed in (21) (only the set of owners which can validate the blocks).
  - (1) true, (2) true, (7) true, (13) true, (15) false, (16) true, (19) true, (20) false, (21) false –> *Use a Private Permissionned Blockchain.* Currently, rental applications go through a centralized system where we cannot rent an apartment or house without trusting a trusted third party. The use of blockchain can be very useful in attributing trust between people in a direct way if there are no other factors that can prevent this decision.
* **Family file sharing application:** it is a simple application to share files between family members or friends. By running our tree, we obtain:
  - The application needs: Data Replication in (1), Cryptographic Layer in (2).
  - The application does not need: to perform Trusted Transactions in (7).
  - The application needs for the Data Transparency in (8).
  - The application does not need: to respect that the response time is a fast option in (10).
  - The application access is restricted in (11). The trust between users is needed in (12).
  - (1) true, (2) true, (7) false, (8) true, (10) false, (11) true, (12) true –> *Use a Trusted peer-to-peer System.* It is a good idea if there are no technical difficulties such as firewalls and costs in terms of storage on the peers.
* **Navigation application "Waze":** it is an application of mobile navigation. By running our tree, we obtain:
  - The application needs: Data Replication in (1), Cryptographic Layer in (2), Trusted Transactions in (7).
  - The application does not need for the Data Transparency in (13).
  - The application needs: that the response time is a fast option in (14).
  - (1) true, (2) true, (7) true, (13) false, (14) true –> *A TTP is Necessary (Use a Centralized System).* This application needs a very fast response time and real-time monitoring. For this reason, a centralized system is necessary.
* **B2B traceable supply chain:** in this application the different actors of the supply chain do not trust each other. By running our tree, we obtain:
  - The application needs: Data Replication in (1), Cryptographic Layer in (2), Trusted Transactions in (7), Data Transparency in (13).
  - The application does not need to respect that the response time is a fast option in (15).
  - The application needs: Data Immutability in (16).
  - The application access is restricted (the different actors of the supply chain) in (19).
  - The trust between users is not needed in (20).
  - The public verifiability is needed in (21) (all actors participate to validate a blocks).
  - (1) true, (2) true, (7) true, (13) true, (15) false, (16) true, (19) true, (20) false, (21) true –> *Use a Public Permissionned Blockchain.* This application needs a decentralized secured environment and the blockchain technology is the best solution in order to track all the events.
* **Simple Web application:** by running our tree, we obtain:
  - The application does not need: Data Replication in (1).
  - (1) false –> *A TTP is Necessary (Use a Centralized System).* It is very type of web application which needs a centralized classical system.

* **Application for diplomas:** it is an application which allows you to certify diplomas forever. The application also makes it possible to verify the certification of these diplomas. By running our tree, we obtain:
  - The application needs: Data Replication in (1), Cryptographic Layer in (2), Trusted Transactions in (7), Data Transparency in (13).
  - The application does not need to respect that the response time is a fast option in (15).
  - The application needs: Data Immutability in (16).
  - The application access is restricted (a set of universities, organisms, students, administrators, etc.) in (19).
  - The trust between users is not needed in (20).
  - The public verifiability is not needed in (21) (only the set of administrators which can validate the blocks).
  - (1) true, (2) true, (7) true, (13) true, (15) false, (16) true, (19) true, (20) false, (21) false –> *Use a Private Permissionned Blockchain.* This type of application needs forever certification and also in a decentralized way where all stakeholders can check the validity of documents at any time. A blockchain is an ideal solution for this type of application if there are no other factors that can prevent this decision.
* **Autonomous car that wants to access an electric charging network:** it is a vehicular and IoT application. It allows anyone to offer its charging station for autonomous vehicles. By running our tree, we obtain:
  - The application needs: Data Replication in (1), Cryptographic Layer in (2), Trusted Transactions in (7), Data Transparency in (13).
  - The application does not need to respect that the response time is a fast option in (15).
  - The application needs: Data Immutability in (16).
  - The application access is not restricted in (19).
  - (1) true, (2) true, (7) true, (13) true, (15) false, (16) true, (19) false –> *Use a Permissionless Blockchain.* If there are no other factors that can prevent this decision, the blockchain is an interesting technology for this type of applications.

6 Related Work

In literature, several decision models have been proposed aiming to identify whether a blockchain is needed or not for a given IT application. We review in this section a selection of these models and we compare them to our proposed decision tree. In fact, our decision tree is based on the needs of the IT application that are directly related to the comparison characteristics discussed in section 4. Our proposed tree helps an IT application to identify exactly which infrastructure is the best solution for the implementation: blockchain or centralized or distributed peer-to-peer. In addition, it specifies the types of infrastructures such as private peer-to-peer system, permissionless blockchain, public permissioned blockchain, etc. The strong point of our decision tree (see Fig-5) is that it addresses the needs of: Cryptographic Layer, Data Transparency, Data Immutability, Trusted Transactions, Response Time versus Loading Time. These needs make the result decisions in the best conditions (see sections 4 and 5.1).

In the research work [36], a simple decision model has been proposed which allows to identify which type of Distributed Ledger Technology (DLT) is the appropriate solution for an IT application. This model takes into consideration only the characteristics: restricted access and the data integrity (only one security property) and it does not treat the decisions for the infrastructures: blockchain, centralized and distributed peer-to-peer. Authors in [34] present a decision model that aims to identify the best suited solution among: do not use a blockchain, permissionless blockchain, public permissioned blockchain and private permissioned blockchain. Another decision model is proposed in [37] which allows to choose the appropriate solution among: public blockchain, private blockchain, do not use a blockchain. Thus, in the research work [38], a decision model is introduced which identifies the most adapted solution among: do not use a blockchain, permissionless blockchain and permissioned blockchain, .

In fact, the three models [34], [37] and [38] are based only on the following characteristics/needs: restricted access, trust between users, public verifiability and the presence of a TTP. They lack addressing the following needs: a fast execution (as in boxes (4), (5), (9), (10), (14), (15) in our proposal), a cryptographic layer and trusted transactions (as in boxes (2), (7) in our proposal). In addition, the model introduced in [34] does not address the decisions: public/ private/trusted peer-to-peer system, centralized system. The model presented in [37] does not take into consideration the decisions: permissionless blockchain, public/private permissionned blockchain, public/private/trusted peer-to-peer system, centralized system. The model proposed in [38] lacks to identify the solutions: public/private/trusted peer-to-peer system, centralized system. We note that the three models [34], [37] and [38] only indicate the decision of "do not use a blockchain" without specifying which conventional system is the best suited. To the best of our knowledge, our proposal was not presented with the same ideas/needs in the literature. This make us the first to give better result decisions than the related work.

7 Conclusion and Future Work

In this paper, we proposed a decision tree identifying whether a blockchain is the appropriate solution for a given IT application, or a classical "centralized or distributed peer-to-peer" system is more adapted. The design of our decision tree is based on the needs of the IT application and the characteristics of the three infrastructures. To the best of our knowledge, our decision has not been previously proposed.

Regarding our future work, we plan firstly to built a website which allows a user to enter the needs of his application and get the decision result. Secondly, we plan to present this website to IT engineers to gather their feedback on our tool.

References

1. S. Nakamoto, "Bitcoin a peer-to-peer electronic cash system," *Working Paper*, 2008.
2. I.-C. Lin and T.-C. Liao, "A survey of blockchain security issues and challenges." *IJ Network Security*, 2017.
3. Z. Wan, R. H. Deng, and D. Lee, "Electronic contract signing without using trusted third party," *International Conference on Network and System Security*, 2015.
4. S. Zaid, G. Linscott, A. Becevello, T. Zaid, and P. Lem, "System and method for anonymous addressing of content on network peers and for private peer-to-peer file sharing," Aug. 18 2015, uS Patent 9,112,875.
5. X. Li, P. Jiang, T. Chen, X. Luo, and Q. Wen, "A survey on the security of blockchain systems," *Future Generation Computer Systems*, 2017.

6. Y. Caseau and S. Soudoplatoff, "La blockchain, ou la confiance distribuée," *Fondation pour l'innovation politique*, 2016.
7. F. L. Haddi and M. Benchaïba, "A survey of incentive mechanisms in static and mobile p2p systems," *Journal of Network and Computer Applications*, 2015.
8. L. Luu, D.-H. Chu, H. Olickel, P. Saxena, and A. Hobor, "Making smart contracts smarter," *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, 2016.
9. J. A. Miller, C. Bowman, V. G. Harish, and S. Quinn, "Open source big data analytics frameworks written in scala," 2016.
10. A. Dorri, S. S. Kanhere, and R. Jurdak, "Blockchain in internet of things: challenges and solutions," *arXiv preprint arXiv:1608.05187*, 2016.
11. E. Hildenbrandt, M. Saxena, X. Zhu, N. Rodrigues, P. Daian, D. Guth, and G. Rosu, "Kevm: A complete semantics of the ethereum virtual machine," *Technical Report*, 2017.
12. K. Khan and W. Goodridge, "Dynamic Adaptive Streaming over HTTP (DASH) within P2P systems: a survey," *International Journal of Advanced Networking and Applications*, 2019.
13. A. Esnault, "Systèmes pair-à-pair pour l'informatique opportuniste," *Ph.D. Thesis, Université de Bretagne Sud*, 2019.
14. A. Bachmann, A. Becker, D. Buerckner, M. Hilker, F. Kock, M. Lehmann, P. Tiburtius, and B. Funk, "Online peer-to-peer lending-a literature review," *Journal of Internet Banking and Commerce*, vol. 16, no. 2, p. 1, 2011.
15. Q. Zhang, W. Zhu, X. Zhang, and Y. Xiong, "Peer-to-peer method of quality of service (qos) probing and analysis and infrastructure employing same," Apr. 13 2010, uS Patent 7,698,460.
16. N. El Madhoun, J. Hatin, and E. Bertin, "Going beyond the blockchain hype: In which cases are blockchains useful for it applications?" in *The 3rd IEEE Cyber Security in Networking International Conference (CSNet 2019)*, 2019.
17. "What Is Ether Mining ?" [Online]. Available: `https://blockspoint.com/guides/ethereum/what-is-ether-mining`
18. M. Pilkington, "11 blockchain technology: principles and applications," *Research handbook on digital transformations*, 2016.
19. M. Belotti, N. Bozic, G. Pujolle, and S. Secci, "A vademecum on blockchain technologies: When, which and how," 2018.
20. J.-P. Delahaye, "Le bitcoin une monnaie révolutionnaire?" *UMR*, 2014.
21. S. Norton, "Cio explainer: What is blockchain?" *The Wall Street Journal*, 2016.
22. A. M. Antonopoulos, "Mastering bitcoin: unlocking digital cryptocurrencies," *" O'Reilly Media, Inc."*, 2014.
23. A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the security and performance of proof of work blockchains," *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, 2016.
24. D. Mingxiao, M. Xiaofeng, Z. Zhe, W. Xiangwei, and C. Qijun, "A review on consensus algorithm of blockchain," *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 2567–2572, 2017.
25. L. Bach, B. Mihaljevic, and M. Zagar, "Comparative analysis of blockchain consensus algorithms," *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pp. 1545–1550, 2018.
26. A. Scott, "Vitalik buterin: I quite regret adopting the term smart contracts for ethereum," 2018. [Online]. Available: `https://bitcoinist.com/vitalik-buterin-ethereum-regret-smart-contracts`
27. V. Buterin, "The Ethereum White Paper: A Next-Generation Smart Contract and Decentralized Application Platform," 2013. [Online]. Available: `https://github.com/ethereum/wiki/wiki/White-Paper`
28. K. Delmolino, M. Arnett, A. Kosba, A. Miller, and E. Shi, "Step by step towards creating a safe smart contract: Lessons and insights from a cryptocurrency lab," in *International conference on financial cryptography and data security*. Springer, 2016, pp. 79–94.
29. D. Yermack, "Corporate governance and blockchains," *Review of Finance*, 2017.
30. M. C. K. Khalilov and A. Levi, "A survey on anonymity and privacy in bitcoin-like digital cash systems," *IEEE Communications Surveys & Tutorials*, 2018.
31. D. Ron and A. Shamir, "Quantitative analysis of the full bitcoin transaction graph," 2013.
32. J. Pons, "La mise en œuvre de la blockchain et des smart contracts par les industries culturelles," *Annales des Mines-Réalités industrielles*, 2017.
33. I. Grigg, "Eos-an introduction," *White paper. https://whitepaperdatabase. com/eos-whitepaper*, 2017.
34. K. Wüst and A. Gervais, "Do you need a blockchain?" *IACR Cryptology ePrint Archive*, 2017.
35. A. Ellervee, R. Matulevicius, and N. Mayer, "A comprehensive reference model for blockchain-based distributed ledger technology," *ER Forum*, 2017.
36. D. Birch, R. G. Brown, and S. Parulava, "Towards ambient accountability in financial services: Shared ledgers, translucent transactions and the technological legacy of the great financial crisis," *Journal of Payments Strategy & Systems*, 2016.
37. B. Suichies, "Why blockchain must die in 2016," 2015. [Online]. Available: `https://medium.com/block-chain/why-blockchain-must-die-in-2016-e992774c03b4`
38. M. E. Peck, "Do you need a blockchain?" 2017. [Online]. Available: `https://spectrum.ieee.org/computing/networks/do-you-need-a-blockchain`