On the Categorisation of Digital Contracts

Carlos Molina–Jimenez[a], Hazem Danny Nakib[b], Sandra Felizia[c] and Bob [d]

[a] carlos.molina@cl.cam.ac.uk, Dept. of Computer Science and Technology, University of Cambridge;
[b] hazemdanny.nakib@gmail.com, University College London;
[c] Faculty of Law, National University of Rosario;
[d] bob@abc.ab, Bob's address

ARTICLE HISTORY
Compiled August 4, 2021

ABSTRACT

The use of computer technology in online business has led to business contracts that, to a lesser or greater extend, are performed automatically. Unfortunately, the concept of contracts is familiar to lawyers but unfamiliar to the technical people that are responsible for producing the technology to automate contracts. On the other side, currently, most lawyers are unfamiliar with the computer technology (e.g. computer code) that underpins automatic contracts. Consequently, they struggle to categorically determine if they are legal contracts as understood within traditional legal systems, for example, the Common and Civil Laws. The misunderstanding between layers and technicians start with the specialised languages that the two communities use. The intention of this paper is to shed some light on the confusion: it clarifies the terminologies used by both communities and precisely defines the concept of contract as understood by technicians.

KEYWORDS
contracts; digital contracts; automatic contracts; smart contracts; blockchains

Contents

Contact: carlos at email: carlos.molina@cl.cam.ac.uk

1. Introduction

Contracts have been used to stipulate the rules of business interactions since ancient times. Contracts formed and carried out without the assistance of computer technology are widely documented and understood, at least by the Law community. For instance, they can be traced back as far as Book 11 of The Laws by Plato [1] and verse 282 of the The Holy Quran [2]. The situation has been disrupted by the inclusion of computer technology in the legal systems aimed at the creation of automated contracts. Technology can be used to automate different contract phases. However, the focus of this paper is on the automation of the actual performance of the contract by means of computer code. We do not discuss database systems that are used for automating the storage and retrieval of documents used for drawing up contracts that are performed manually [3, 4, 5]. Academic research aimed at contract automation can be trace back at least to the mid1980's [6, 7]. Since them, several ideas on contract automation have been suggested by the Computer Science community. In these efforts, automated contracts have been referred to as electronic contracts, executable contracts, digital contracts, programmable contracts and other names. In 1997 Nick Szabo [8] called them smart contracts, an unfortunate term that became popular in 2008 when the Bitcoin platform demonstrated their potential in the implementation of decentralised systems [9]. We say unfortunate because in our opinion, the term generates unnecessary confusion outside the Computer Science community.

Firstly, there is nothing smart in the contracts implemented in current blockchain platforms like Bitcoin [10], Ethereum [11] and Hyperledger [12] or in their predecessors [13, 14, 15, 16, 17]. All of them are realised as conventional computer code written by programmers to execute some actions (for example, transfer N dollars from Alice to Bob) when certain conditions (for example, expiration of a payment deadline) hold.

Secondly, there are arguments risen by lawyers that the word contract is inadequate: a thorough examination of these pieces of code their uses will reveal that some of them are not contracts as conceived by lawyers [18]. In fact, has has been observed by some authors (see for example [19]) there is a mismatch between the terminology used by layers and the one used by computer science people. Incidentally, in law language to execute a contract means to sing a contract whereas in technical language it means to run the computer code that implements the contract. In this article, we will use the term **digital contract**.

The main aim of this paper is to help clarify the current confusion about what automatic contracts are with the intention of helping both the law and computer science community understand each other. In pursuit of this aim we provide a a categorisation of existing contracts in accordance with the degree of automation that they use in the performance stage.

We have developed a number of vectors by which to classify contracts:

In addition to the degree of automation contracts can be categorised from other perspectives, accordingly, have developed a number of vectors by which to classify contracts. We briefly mention some that are directly related to the issue of automation. We believe that this additional discussion will place this article in a global context. Incidentally, there are bilateral and multilateral contracts; a contract can be deployed in a centralised manner (on a single trusted third party) or fully decentralised (on a blockchain); we can also categorise contract in accordance with the type of signatures that they use (hand–written or electronic) and from the dregree of degree of intrusiveness (monitoring or enforcement).

A salient feature of our work is that we explain our definitions and arguments with the assistance of examples and flowcharts that show graphically the life cycle of contracts. We believe that they will help readers unfamiliar with computer technology appreciate the technology that underpins automated contracts and appreciate their legal implications, for instance, to informatively determine if a given automated contract meet the requirements to be declared a legal contract.

In pursuit of this aim, in section 2 we describe the contract terminology used by law and computer community and relate terms to each other. In Section ... In Section 8 we discuss work that has motivated and influenced our. We use section 7 to open questions that in our opinion need further attention. Conclusion remarks are presented in Section 9.

2. Contract automation and terminology

The advantages of conducting business online and recent progress in computer technology have motivated computer engineers to implement contracts that to a lesser or greater extent are enforceable automatically. As a result we have currently a large variety of contracts with different properties that from the degree of automation can be divided into two large categories: **non–automatic** and **automatic contracts**.

Within the category of non–automatic contracts we group the traditional contracts normally recorded on paper in a natural languages (for example, in English). They drawn up by lawyers and performed and enforced with no signs of automation. Within the category of automatic contracts we group contracts that are performed partially or entirely automatically; therefore the performance of at least some of their actions involve the execution of computer code. These contracts are the result of the convergence of law and computer science which is the main topic of interest of this article. They are written by computer programmers in programming languages (for example Solidity) and are poorly documented and understood. As mentioned in Section 1, they are an emerging technology and consequently are known under different terms; in this article we will use the term **digital contracts** to refer to them. We will justify our choice in Section 7. The categories of non–automatic and automatic contracts can be further divided into subcategories. To get the feeling it might be helpful to look at Fig. 1 which shows an overview of contract categorisation.

We will discuss each category thoroughly in subsequent sections; for the time being, enough is to understand that non–automatic contracts are drawn up and enforced in court whereas automatic contracts are implemented by computer engineers and enforced by computer code.

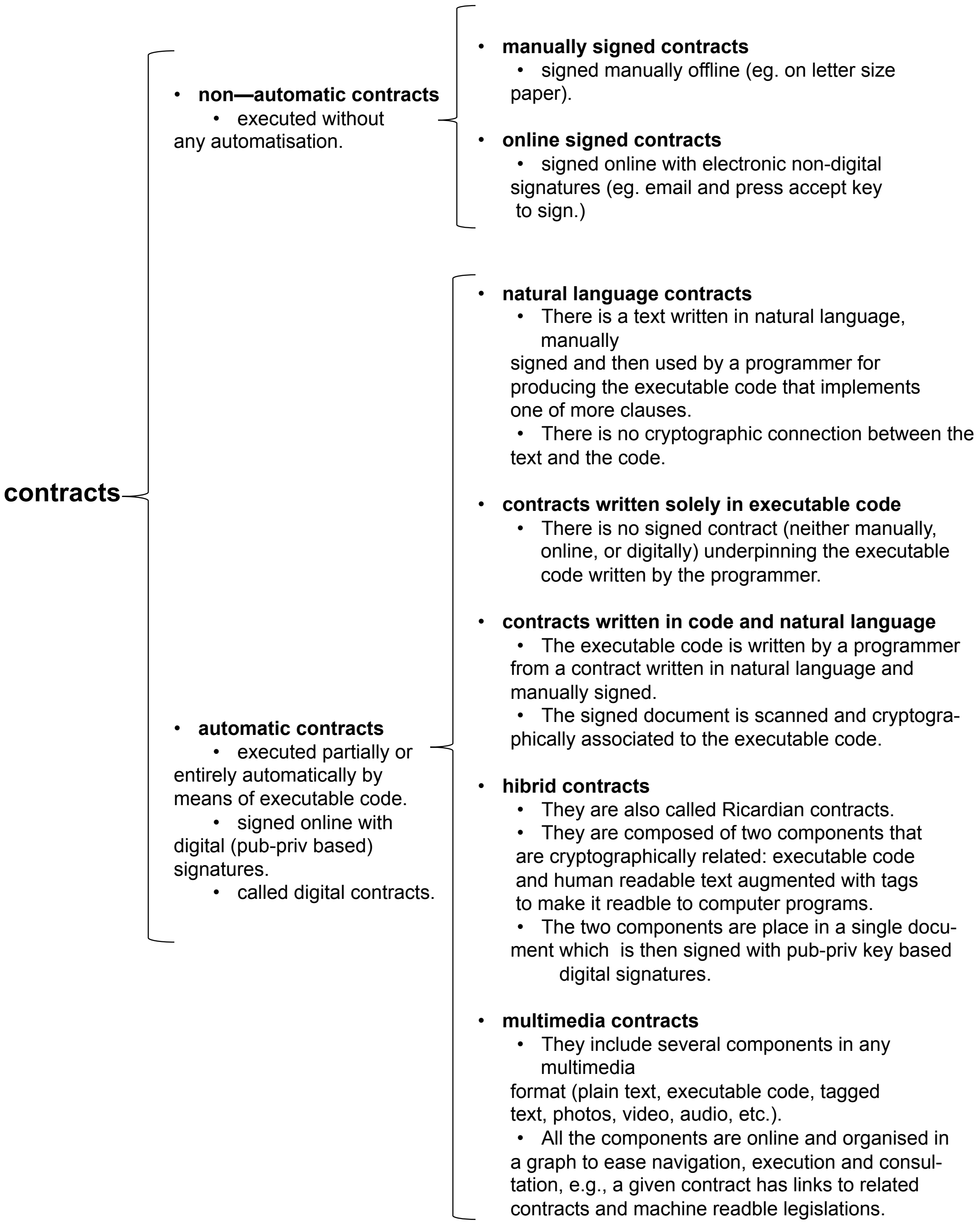
The work of both communities complement each other's and ideally it should be carried out in close collaboration. Unfortunately this is not straight forward. Lawyers find contract automation challenging to understand whereas computer engineers struggle to understand what a contract is. One of the major difficulties is the mismatch between the specialist languages used by lawyers and computer engineers. Incidentally, the term contract and related terminology that is used for describing the contract life cycle have different meaning to lawyers and com- puter engineers. To shed some light on the confusion, we will discuss the terms used by lawyers in this section. Next in Section 2.2 we will compare and relate them to the terms used by computer engineers. We believe that the comparison will help lawyers to place contracts used in the Internet in a juridical framework. For instance it will help them categorically determine what contracts used in the Internet meet all the requirements to be considered legal contracts (in a given jurisdiction) and which fall short to be considered legal.

To shed some light on the confusion, we will discuss the life cycle of non–automatic and automatic contract next. We will also compare compare and relate to each other the terminology used by both communities. We believe that the comparison will help lawyers to place contracts used in the Internet in a juridical framework. For instance it will help them categorically determine what contracts used in the Internet meet all the requirements to be considered legal contracts (in a given jurisdiction) and which fall short to be considered legal.

2.1. *Non–automatic contracts and lawyers' terminology*

Under a legal context, a **contract** is a formal agreement between two parties. By formal we mean that the agreement complies with all the elements that the legal system where the contract is formed, demands to be a binding arrangement that is legally enforceable in court. There are contracts celebrated between more than two parties (they are called multiparty contracts). However, we do not cover them in this paper because their automation is far more demanding than contracts between only two parties (also called bilateral contracts). The two parties that celebrate the contract and are legally bound are called the **signatories**. To ease the explanation in the examples of contract that we show in this paper the signatories are Alice and Bob. They represent two human beings that commit to that contracts, either on personal basis or as representatives of their respective companies.

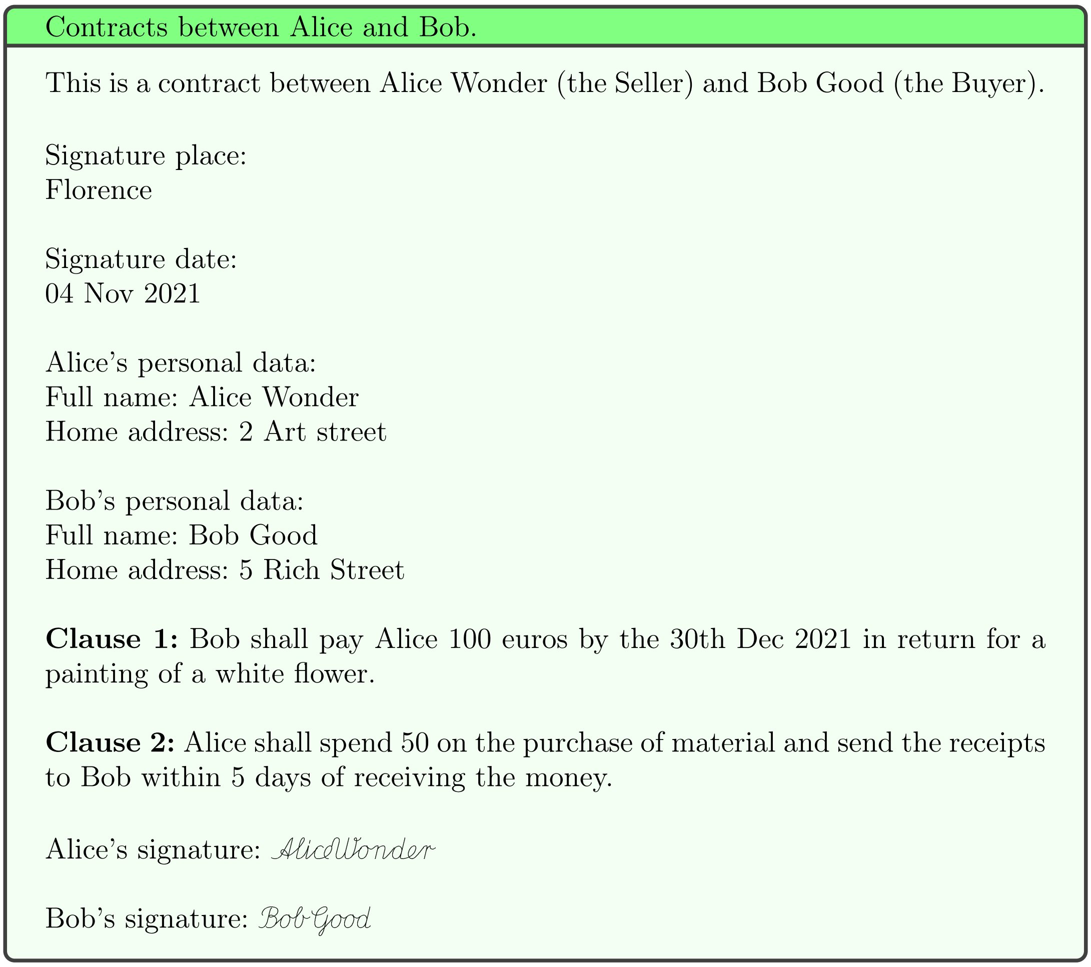
It is worth clarifying that some scholars use the terms contracts and agreements



contracts

- **non—automatic contracts**
  - executed without any automatisation.
    - **manually signed contracts**
      - signed manually offline (eg. on letter size paper).
    - **online signed contracts**
      - signed online with electronic non-digital signatures (eg. email and press accept key to sign.)
- **automatic contracts**
  - executed partially or entirely automatically by means of executable code.
  - signed online with digital (pub-priv based) signatures.
  - called digital contracts.
    - **natural language contracts**
      - There is a text written in natural language, manually signed and then used by a programmer for producing the executable code that implements one of more clauses.
      - There is no cryptographic connection between the text and the code.
    - **contracts written solely in executable code**
      - There is no signed contract (neither manually, online, or digitally) underpinning the executable code written by the programmer.
    - **contracts written in code and natural language**
      - The executable code is written by a programmer from a contract written in natural language and manually signed.
      - The signed document is scanned and cryptographically associated to the executable code.
    - **hibrid contracts**
      - They are also called Ricardian contracts.
      - They are composed of two components that are cryptographically related: executable code and human readable text augmented with tags to make it readble to computer programs.
      - The two components are place in a single document which is then signed with pub-priv key based digital signatures.
    - **multimedia contracts**
      - They include several components in any multimedia format (plain text, executable code, tagged text, photos, video, audio, etc.).
      - All the components are online and organised in a graph to ease navigation, execution and consultation, e.g., a given contract has links to related contracts and machine readble legislations.

Figure 1. Contract categories from the perspective of automation.



Contracts between Alice and Bob.

This is a contract between Alice Wonder (the Seller) and Bob Good (the Buyer).

Signature place:
Florence

Signature date:
04 Nov 2021

Alice's personal data:
Full name: Alice Wonder
Home address: 2 Art street

Bob's personal data:
Full name: Bob Good
Home address: 5 Rich Street

Clause 1: Bob shall pay Alice 100 euros by the 30th Dec 2021 in return for a painting of a white flower.

Clause 2: Alice shall spend 50 on the purchase of material and send the receipts to Bob within 5 days of receiving the money.

Alice's signature: *AliceWonder*

Bob's signature: *BobGood*

Figure 2. Contract written in English and signed manually by Alice and Bob.

interchangeably, yet strictly speaking, in legal contexts they are not equivalent. Agreements are regarded as informal arrangements that are not legally enforceable [20].

The elements that a contract must contain depend on the jurisdiction under which the contracts is formed. For example, in Common Law, a contract must contain an offer, acceptance and consideration. Normally, the contract is written on paper in natural language (for example English, Spanish, etc.) and signed under holographic signatures.

To be able to relate the elements of a contract to the elements of automatic contracts discussed in subsequent sections we will discuss the life cycle of a contract. To explain our argument we will use the example shown in Fig. 2.

Fig. 3 shows a simplified view of the life cycle of a traditional non–automatic contract. To focus our attention on concepts that we wish to discuss in this article we will assume that Alice and Bob agree to record their contract on a piece of paper (for example A4) in natural language (for example, in English) and sign it under holographic signatures. This assumption it leaves out contracts formed orally and in other forms. Also, the flow chart of Fig. 3 is simplifies as it shows only the core stages of the life cycle of traditional non–automatic contract; yet it can help to contrast non–automatic against automatic contracts.

We clarify that the contract that we are Fig. 2 using only an hypothetical example specifically tuned and simplified to help in the explanation of the concepts the we wish to highlight. Incidentally, it does not dictate over what medium the payment and receipt are to be delivered.

The stages included are familiar to lawyers who can immediately relate them to the law system. The life cycle consist of three stages with their respective sub–stages [21].

2.1.1. Contract life cycle in common law

- Formation: Is a non–biding negotiation process conducted by two potential contracting partners with the intention of reaching agreement on the terms and conditions to include in a legal contract.
    - **Offer** (box 2): An offer is the manifestation or willingness of one party to enter into a bargain with another party. Another word commonly used in legal language to refer to offer is **promise**. The party that presents the offer is called the **offeror or promisor** and the party that receives the offer is called the **offeree or promisee**.

      In the example of Fig. 3 Bobs is the offeror and Bob is the offeree. The offer is presented by Bob to Alice in step 2. For example, within the context of the contract shown in Fig. 2, and offer would be ”a payment of 100 euros to Alice in return for two paintings".
    - **Consideration** (box 3): Is something of value that the offeree's promise to give to the offeror in exchange for the offer. A common example of consideration is to give money but there are others such as to create a piece of art, implement a mobile phone application and to give personal data records. Incidentally, in the example of Fig. 2, Alice's consideration is a painting of a white flower.
    - **Acceptance** (rhombus 4): Is a manifestation to assent to the terms of the offer made by the offeree. As shown in rhombus 4, acceptance means a deal between Alice and Bob where Alice is satisfied with Bob's offer and Bob is satisfied with Alice's consideration. Accordingly, acceptance leads to contract execution. On the contrary, non–acceptance (that is, deal) leads to continuation of the negotiation process in rhombus 5. A negotiation process not always complete in acceptance, as shown in rhombus 5, Alice and Bob might opt to continue their negotiation or to abandon it. In the first alternative leads to box 6 where Bob alters his offer on the hope that the new version will satisfy Alice when presented to her in box 2. The second alternative of rhombus 5 leads to ellipse 12 where the negotiation is abandoned.
    - **Meeting of minds:** Is the process of writing the terms and conditions agreed upon by Alice and Bob on a piece of paper (for example, A4) and signing it. The contract shown in Fig. 2 has been signed by Alice and Bob under their holographic signatures.
- **Performance**: Is the act of executing the needed actions to fulfill the obligations promised in the contract. The execution of actions is represented by box 10 which assumes that the actions are executed sequentially. For example, in a typical financial contract the actions are submission of first payment, followed by payment of interest which is later followed by submission of second payment and so on [22].

  In practice, actions executed are not necessary contract–compliant. A given execution can potentially cause a contract breach. Rhombus 9 represents this possibility. If the execution of the action in curse is contract compliant the contract progresses smoothly to rhombus 11. However, a breach of the contract leads

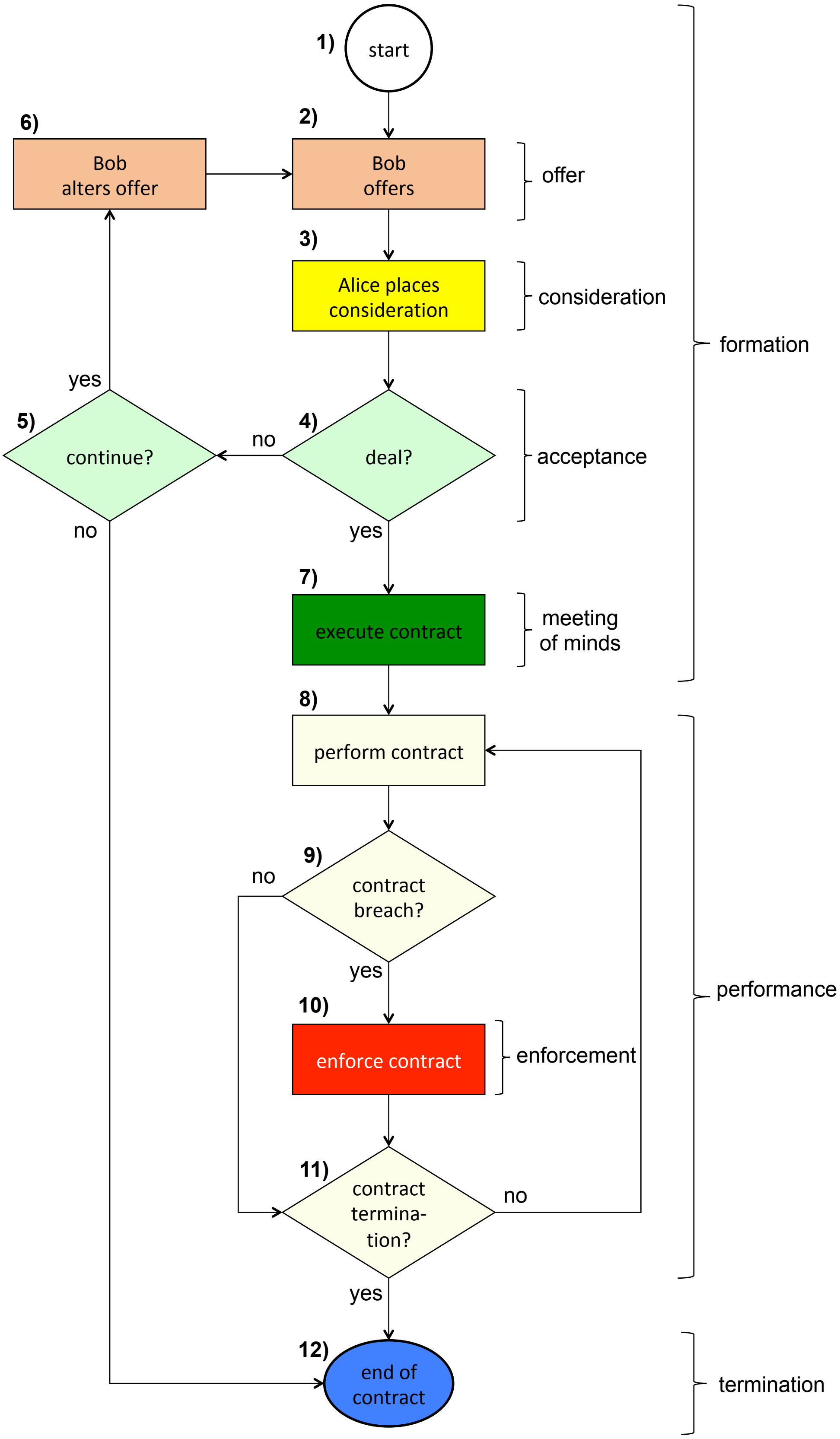


Figure 3. Life cycle of a civil law contract between Alice and Bob.

to contract enforcement as represented by box 10.

  - **Contract enforcement** (box 10): Is dispute resolution process aimed at the clarification of the situation to repair the damage caused by the misbehaviour of the offending party against the victim. Typically, contract enforcement takes place in court but other mechanisms exist such as arbitration [23]. As represented by rhombus 11, the contract remains in the performance state until terminates possibly after contract enforcement but not necessarily. Notice that completion of the contract enforcement leads to rhombus 11 which captures the possibility that Alice and Bob might terminate the contract after enforcement or alternatively, continue it performance as indicated by the arrow that leads from rhombus 11 to box 8.

- **Termination** (box 11:) Is the completion of the contract. In and ideal situation the contract completes when both parties have fulfilled their obligations (the work is done). However, there other situation that might cause contract termination such as mutual agreement to terminate the contract earlier and contract breach. How and when do legal contracts terminate completely?

2.1.2. Contract life cycle in continental law

Contracts play a central role in the ordering of human relations and therefore have an irreplaceable importance in societies. Hence, contract law is probably the most relevant private law institution of individual self-determination and autonomy and it has evolved continuously to respond to the emergence of new contract models. Today, like many other legal institutions, it faces the challenges of digitization. Smart contracts pave the way for a new era of contracting and pose a potential challenge to the prevailing concepts of contract law. We are aware that the contractual stages are insufficient in their rigidity for the resolution of specific cases. However, we consider that they are useful for dogmatic analysis.

Phases in the life of the contract:

There are usually four phases or moments in the life of a contract:

(1) Formation or precontractual,
(2) Execution,
(3) Performance
(4) Postcontractual.

The first period includes the preliminary phase from which consent can be derived. The second phase represents the birth of the contract, when the concursus volun tatum occurs - the manifestations of their will. The third period concerns compliance of the contract, on the performance phase which represents the realization of its natural destiny and the fourth period, arises after the termination of the contract when certains responsabilities (i.e. eviction) and duties (i.e. confidentiality) remain.

(1) Formation phase: is a non-binding negotiation process conducted by two potential contracting patners with the intention of reaching agreement on the terms and conditions to include in a legal contract. Sometimes consent, the core of the contract, occurs instantaneously with the agreement of the manifestations of the will of the parties. This happens frequently in small transactions or in contracts for adherence to general conditions. Other times, on the contrary, the parties need a previous period to deliberate and discuss the conditions of the contract. In the first case, the formation of the contract is a snapshot. In the second hy-

pothesis, the contract is formed ex intervallo tem poris; there is an antecedent stage, of variable duration, in which the parties deliberate, discuss and elaborate what, ultimately, will be a rule of private autonomy. There are contracts such as those of progressive formation that, before their execution have been preceded by preliminary discussions and deliberations. In this preliminary period there may be negotiations prior to the offer, or there may even be a moment of coinciding wills on the part of the future contractual content or partial agreements. This pre-contractual period can serve as a prelude to the contract. At this stage of the contract's gestation, there is still no concrete and precise offer, nor is there a definitive and total acceptance. It is a stage of dissent, trial and error, reservations and discussion. In the negotiation stage, their recess or break, in principle, constitutes something admissible and natural that is not abnormal or unlawful. The essential for the existence of this business is consent, not the preceding debate or prior discussion. Yet when the latter do not take place, the contract is perfected the same when a party is limited to adhere to the offer made by the other, without any kind of prior deliberation This is what happens in mass contracting and in the contract by adhering to general conditions.

Offer (box 2): An offer is the manifestation or willingness of one party to enter into a bargain with another party. Another word commonly used in legal language to refer to offer is promise. The party that presents the offer is called the offeror or promisor and the party that receives the offer is called the offeree or promisee. The offer may arise from preliminary negotiations.

In the example of Fig. 3 Bobs is the offeror and Alice is the offeree. The offer is presented by Bob to Alice in step 2. For example, within the context of the contract shown in Fig. 2, and offer would be "a payment of 100 euros to Alice in return for two paintings".

Acceptance (rhombus 4): Is a manifestation to assent to the terms of the offer made by the offeree. As shown in rhombus 4, acceptance means a deal between Alice and Bob where Alice is satisfied with Bob's offer and Bob is satisfied with Alice's consideration. Accordingly, acceptance leads to contract execution (second pase). On the contrary, non–acceptance (that is, deal) leads to continuation of the negotiation process in rhombus 5.. A negotiation process not always complete in acceptance, as shown in rhombus 5, Alice and Bob might opt to continue their negotiation or to abandon it. In the first alternative leads to box 6 where Bob alters his offer on the hope that the new version will satisfy Alice when presented to her in box 2. The second alternative of rhombus 5 leads to ellipse 12 where the negotiation is abandoned. Acceptance must be fully in accordance with the proposal and any modification would be a counter offer.

(2) Execution phase: Involves the signing of the contract -if required-and the initiation of the agreement. The preclusion or closure of the stage of preliminary negotiations (iter negocial) occurs when both parties express their consent. The common will of the parties shows that they want the legal business. The conclusive stage of the contract occurs when there is an agreement, which is the moment when the contract is born and acquires existence. This agreement can be reached by various procedures. The generic model is that of the offer followed by the acceptance. It involves a sequence in which one of the parties takes the initiative and issues a statement addressed to the other, proposing the conclusion of the contract, and the latter, in turn, issues another statement of will accepting the offer. The meeting between the offer and the acceptance is the normal way by which the agreement is generated. By the principle of freedom of contract,

the consent -consensu- of the parties produces the perfection of the contract and acquires binding force (with some exceptions such as those contracts which need specific form -ad solemnitatem)

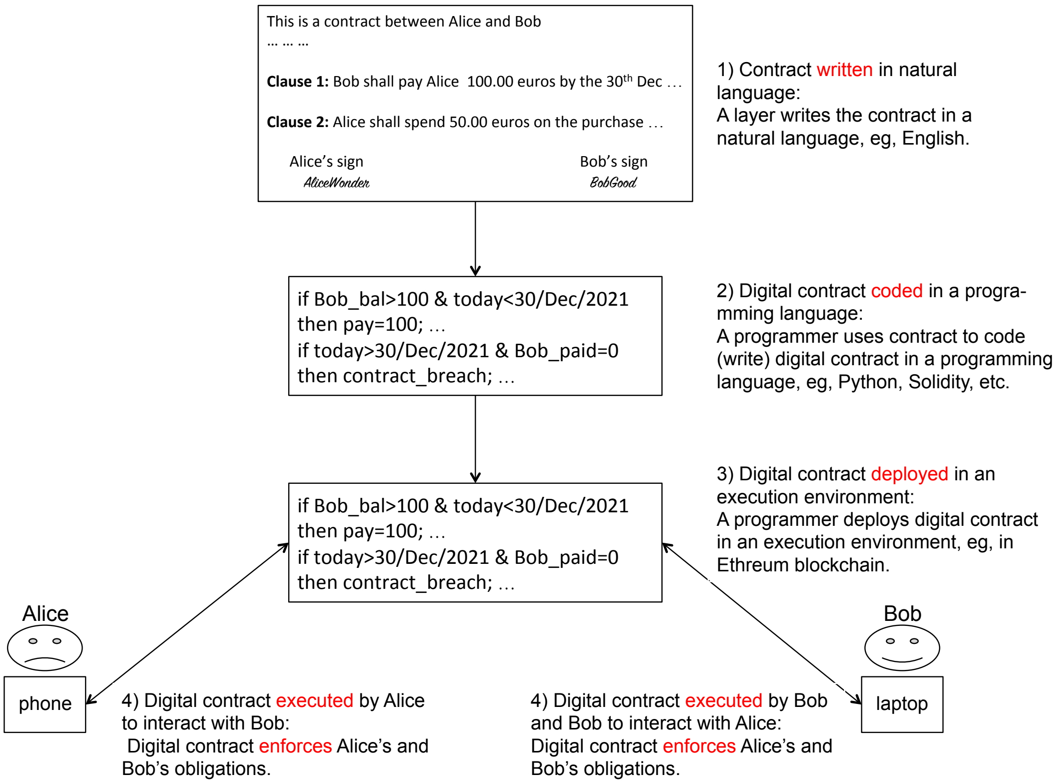
(3) Performance, enforcement and termination phase: Is the act of executing the needed actions to fulfill the obligations promised in the contract. The execution of actions is represented by box 10 which assumes that the actions are executed sequentially. For example, in a typical financial contract the actions are submission of first payment, followed by payment of interest which is later followed by submission of second payment and so on [22]. In practice, actions executed are not necessary contract–compliant. A given execution can potentially cause a contract breach. Rhombus 9 represents this possibility. If the execution of the action in curse is contract compliant the contract progresses smoothly to rhombus 11. However, a breach of the contract leads to contract enforcement as represented by box 10. Enforcement contract; Is dispute resolution process aimed at the clarification of the situation to repair the damage caused by the misbehaviour of the offending party against the victim. Typically, contract enforcement takes place in court but other mechanisms exist such as arbitration [23]. As represented by rhombus 11, the contract remains in the performance state until terminates possibly after contract enforcement but not necessarily. Notice that completion of the contract enforcement leads to rhombus 11 which captures the possibility that Alice and Bob might terminate the contract after enforcement or alternatively, continue it performance as indicated by the arrow that leads from rhombus 11 to box 8. Termination (box 11:) Is the completion of the contract. In and ideal situation the contract completes when both parties have fulfilled their obligations (the work is done). However, there other situation that might cause contract termination such as mutual agreement to terminate the contract earlier and contract breach. How and when do legal contracts terminate completely?

(4) Postcontractual phase: After concluding the contractual relationship, the stipulations agreed upon by the parties remain in force (restitution, reparation of damages, dispute resolution and any other that regulate the rights and obligations of the parties after the termination).

2.2. *Digital contracts and computer engineers' terminology*

The contract shown in Fig. 2 can be performed non–automatically, that is, without using digital technology in the performance state. Another alternative is to convert it into a digital contract that can be performed it automatically, that is, a contract where the actions that take place in box 10 of Fig. 3 are performed without human intervention. As briefly discussed in [19] and [24] computer engineers use a terminology to discuss digital contracts that does not match the terminology that lawyers use. In the discussion of this section we will use will use the terms used by computer engineers. In section 2.2 we will relate the terms used by lawyers an computer engineers to each other.

Fig. 4 shows the conversion of the contract shown in Fig. 2 into a digital contract. A digital contract is a piece of executable computer code that can be deployed and executed (in other words, run, launched or enacted) to enforce (that is, to perform in lawyers' terminology) a contract automatically. Since the executable code represents the contract, we will call it the contract code. Fig. 4 shows how the executable code that implements that contract is produced and executed.



This is a contract between Alice and Bob
… … …

Clause 1: Bob shall pay Alice 100.00 euros by the 30th Dec …

Clause 2: Alice shall spend 50.00 euros on the purchase …

Alice's sign AliceWonder

Bob's sign BobGood

1) Contract written in natural language:
A layer writes the contract in a natural language, eg, English.

```
if Bob_bal>100 & today<30/Dec/2021
then pay=100; …
if today>30/Dec/2021 & Bob_paid=0
then contract_breach; …
```

2) Digital contract coded in a programming language:
A programmer uses contract to code (write) digital contract in a programming language, eg, Python, Solidity, etc.

```
if Bob_bal>100 & today<30/Dec/2021
then pay=100; …
if today>30/Dec/2021 & Bob_paid=0
then contract_breach; …
```

3) Digital contract deployed in an execution environment:
A programmer deploys digital contract in an execution environment, eg, in Ethreum blockchain.

4) Digital contract executed by Alice to interact with Bob:
Digital contract enforces Alice's and Bob's obligations.

4) Digital contract executed by Bob and Bob to interact with Alice:
Digital contract enforces Alice's and Bob's obligations.

Figure 4. Coding, deployment and execution of a digital contract.

(1) The figure assumes that Alice and Bob have agreed and signed to the terms and conditions stipulated by clause 1 and clause 2 of Fig. 2. They hire a lawyer to write the contract in natural language as shown in the figure.

(2) Alice and Bob hire a programmer to produce the digital contract. The programmer translates the terms and conditions included in the text in natural language into a program coded (written) in a programming language, for example, in Solidity or Python. The resulting code is a digital contract that is equivalent to the original contract in English but executable. The lines shown inside the box give an idea about the representation of the contract in in executable code. Strictly speaking, these lines have to be compiled (translated to the binary language of 0s and 1s that computer can execute) before they are actually executable, but these technical details are irrelevant at the level of discussion of this paper. Thus, we will say that the digital contract produced by the programmer is executable.

The contract code is written, deployed and launched by programmer for the benefit of the signatory parties. Fig. 4 explains the process that leads to the implementation and execution of a digital contract.

The aim is to give the general idea of how digital contracts are created. We do not take into account the particularities of the contract in natural language or the category of digital contract (see Fig. 1). The figure assumes that Alice and Bob can connect to the Internet from their respective laptops and reach the execution environment where the digital contracts is in execution. It also assumes that the contract written in English has already been formed, written on paper and signed under holographic signatures.

(1) **Contract written in natural language:** A lawyer is responsible for writing

the contract that Alice and Bob have agreed to. The contract is written in a natural language, for example in English or Spanish, say on a A4 sheet, and signed under holographic signatures.

(2) **Digital contract coded:** Alice and Bob hire a programmer to produce the digital contract. The programmer translates the terms and conditions included in the original contract to a program coded (written) in a programming language, for example, in Solidity or Python. The result is a digital contract that is equivalent to the original contract but executable. The lines shown in the box give an idea about the structure of contract expressed in executable code. Strictly speaking, these lines have to be compiled before they are actually executable, but these technical details are irrelevant at the level of discussion of this paper. Thus, we will say that the digital contract produced by the programmer is executable.

(3) **Digital contract deployed:** The programmer deploys the digital contract in an execution environment. For example, on a blockchain like Ethereum or Hyperledger. The digital contract is now ready to react to actions sent by Alice and Bob from their respective computer (for example, laptops, mobile phones, tablets, etc.). We say that the digital contract is now in execution. The execution environment guarantees that the digital contract is protected from alterations and remains operational when the business interaction between the participants is on course. We return to this issue in Section 6.1.

(4) **Digital contract executed:** Alice and Bob execute the digital contract to interact with each other. The digital contract is responsible for enforcing Alice's and Bob's obligations stipulated in the original contract

2.2.1. Rights, obligations, prohibitions and operations

Form a technical perspective, a contract is conceived as a set of clauses that stipulate **rights**, **obligations** and **prohibitions** that the signatories are expected to comply with. Rights, obligations and prohibitions are associated to at least one operation. An **operation** is a business action executed by a party that changes the state of the contract development.

Intuitively, a right is an operation that a party is entitled to execute. An obligation is an operation that a party is expected to execute. Finally, a prohibition is an operation that a party is not expected to execute. For example, in the contract of Fig. 2 Bob has the obligation to pay Alice 100.00 under certain conditions (by 31/Dec/2021). To honor this obligation Bob needs to successfully execute through some mechanism the corresponding operation "pay 100.00 to Alice" by the deadline. In practice, contracts include several obligations. For example, a buyer has the obligation to pay and a seller the obligation to deliver and obligation to refund. Therefore to comply with whole contract, the signatory parties need to honor each obligation by means of executing the corresponding operation— pay, deliver and refund in this example. The motivation for using digital contracts is that they automate the execution of operations in compliance with the rights, obligations and prohibitions stipulated in the contract. Automatic execution frees the signatory parties from the hassle of performing them manually to honour the corresponding obligations.

To understand enforcement in digital contracts, it help to regard the execution of a digital contract as the execution of several interrelated operations where the execution of one of them honours an obligation and might enable or disable others.

Contracts normally include several right, obligations and prohibitions, therefore, they involve the execution of several interrelated operations where the execution of

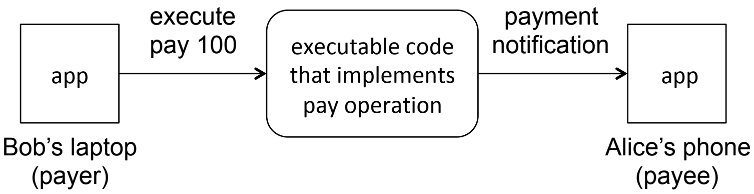


Figure 5. Execution of pay operation.

one of them honours an obligation and might enable or disable others. Some authors regard each right, obligation and prohibition as an individual contract. In this model, a contract is composed of one or more interrelated sub–contracts.

2.2.2. Contract execution

In automatic contracts, operations are executed through the execution of executable code. Fig. 5 shows the execution of a payment operation. Pay 100 is assumed to be stipulated in a contract agreed upon between Bob (the payer) and Alice (the payee). Bob's application is installed in his laptop and Alice's is installed in her mobile phone. The executable code that implements the pay operation is assumed to be implemented in a programming language and deployed in a computer, a local one, in a cloud server or on a blockchain, these implementation details are irrelevant here.

To pay Alice, Bob's application executes the operation 'pay 100' against the executable code. As a response, the executable code executes the operation and as a result, Alice's application receives a notification of payment, for example, bank evidence of the payment.

Notice that Fig. 5 shows only the execution of the pay operation. There are no digital mechanisms in place to detect Bob's failure to execute the pay operation or to enforce him to execute it automatically. This limitation is addressed with the help of a digital contract as shown in Fig. 6 and Fig. 7.

2.2.3. Contract monitoring

Contract monitoring is a technique where a digital contract is execute passively observe the development of a contract and to store records of the operation executed by the signatory parties. Monitoring is passive in the sense that it does not interfere with development of the contract; it only observes and keeps records for potential post–morten examination, for example, if a dispute is risen.

Fig. 6 shows a monitoring digital contract that can be used for monitoring the execution of the payment operation. Notice that the digital contract is directly interrelated with executable code that implements the payment operation. In fact, in existing literature the two components are frequently discussed as a single one. In our view, a separation helps understand how automatic contracts work.

(1) Bob's application places the operation ”execute pay 100” against the executable code.
(2) The executable code executes the operation and as a result, Alice's application receives ”payment notification”, for example, a bank receipt.
(3) The executable code provides the digital contract that is responsible for monitoring with records of the execution of the pay 100 operation initiated by Bob.
(4) The monitoring contract analyses the records, determines if ”pay 100” operation is

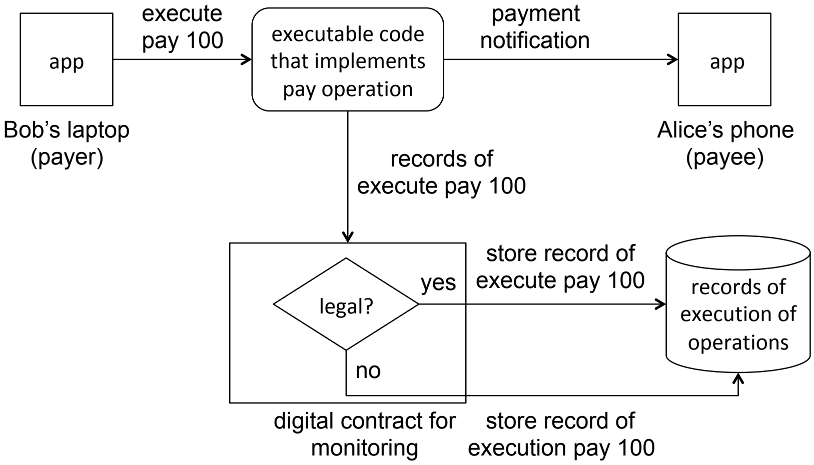

app
execute pay 100
executable code that implements pay operation
payment notification
app
Bob's laptop (payer)
Alice's phone (payee)
records of execute pay 100
legal?
yes
no
store record of execute pay 100
records of execution of operations
digital contract for monitoring
store record of execution pay 100

Figure 6. Monitoring of the execution of pay operation.

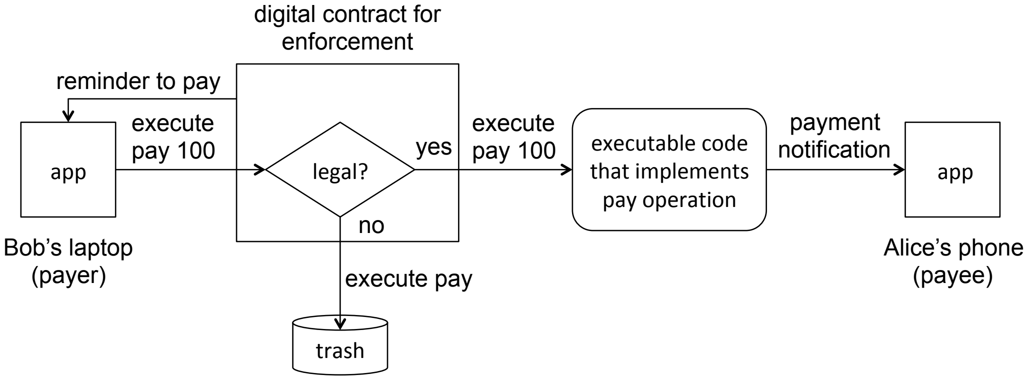


Figure 7. Enforcement of the execution of pay operation.

legal (contract compliance) or not and stores the record along its results (contract compliant or non–contract compliant) in a data base. The records accumulated in the data base can be used for conducting post–mortem (off–line) examination of the contract development.

2.2.4. Contract enforcement

Contract enforcement is a technique where a digital contract is executed to observe the development of a contract and to interfere with execution of operations to ensure that the parties do not breach the contract. As shown Fig. 7, to be preventive, enforcement needs to operate intrusively, rather than non–intrusively like in contract monitoring.

In the example of the figure, a digital contract is deployed for enforcing the execution of the "pay 100" operation shown in Fig. 5 and Fig. 6. The digital contract is responsible for following the development of the contract and ensuring that the "pay 100" operation is executed when needed as stipulated in the contract, for example, within the deadline.

(1) Bob's application places the execution of the "pay 100" operation against the executable code.
(2) The operation is intercepted by the digital contract and analyzed for contract compliance. If it is, the contract forwards the operation to the code that im-

plements the pay operation, otherwise (if the operation is illegal) the contract trashes the operation so that its execution is prevented.

(3) If the ”pay 100” operation reaches the executable code, the ”pay 100” is executed. Alice’s application does not necessarily receive the actual money, it might receive only a payment notification as shown in the figure, for example, a bank receipt.

Hazem: I don’t understand this sentence about enforcing contract.

(4) The enforcement contract is responsible for assuring that Bob honors his obligation to pay. Accordingly, it includes a reminder mechanism (reminder to pay) that is activated when the contract notices that Bob’s application has not paid yet and the deadline to pay is about to expiry. Notice also that the contract allows the execution only of legal operations; for instance it will not allow Alice’s application to execute a ”pay 100” operation outside the pay window, that is, before or after the agreed upon pay days.

Hazem: Yes but doesn’t it then force alice to make the payment automatically even if she no longer wants to?

At implementation level, the digital contract is a finite state machine that follows the development of the contract. It knows precisely what operations have been fulfilled, violated and pending; as a result that contract is capable of stopping the execution of illegal operations and of enforcing the execution of pending operations or at least of sending warning messages.

The power to enforce depends on the particularities to execute the operations; for instance, a contract is able to enforce Bob’s ”pay 100” operation if it is provided with money in advance, such as in escrow, or linked to Alice’s accounts via some API; otherwise, the contract can only send a warning message to Bob’s application to remind him of his pending obligation; next it is up to Bob’s application to honor or violate the contract.

From the perspective of the level of interference that the digital contract causes in the execution of the contractual operations, digital contracts can be categorized into two classes: monitoring and enforcing contracts which are shown, respectively in Fig. 6 and Fig. 7. The distinction between monitoring and enforcement is crucial in Lex Cryptographia and envisioned by Primavera De Filippi [25] and other authors. Unfortunately, they do not seen to account for monitoring, their main focus is on enforcement.

Hazem: Whats the traditional manner? Relying on the good will of the parties and when that fails relying on the courts? And then on other government or public officials to enforce it?

2.3. *Degrees of automation*

Notice that that the automation in the performance of contracts is not a binary (all or nothing) but a line that extends from no automation to increasing degrees of automation. Understanding this as a spectrum is helpful. In fact, contracts that, to a lesser or greater extent provide some degree of automation are already in use in current online businesses. For instance, users normally sign contracts online with Internet Service Providers to subscribe to broadband services; likewise users gain access to social networks like FaceBook and Twitter after signing contracts online that stipulate the terms and conditions of access to the social network platform, online shopping also involve the signing of contracts online. However, the the aim is to implement more sophisticated contracts that can be used to automate the enforcement of the law as envisioned in the Lex Cryptographia [25] and Computational Law [?], Computable Contracts [?] and Machine Readable Legislations [26] and Wise Contracts [26, 27]. Progress in contract automation has resulted in a large variety of contracts that we categorise as shown in Fig. 1. We will discuss the subcategories of automated contracts separately in subsequent sections.

2.4. *Comparison of layers' and software engineers' terminology*

A close look at Fig. 3 and Fig. 4 will reveal that there is no consensus between the terminology used by layers and software engineers. The mapping of legal to technical concepts is not one to one and straight forward. For example, the separation of stages in a non–automatic contract as shown in Fig. 3 are not blurred in a digital contract [28]. Also, some concepts exist in legal contracts but not in digital contracts and vice versa. For example, in non–automatic contracts signatures are placed simultaneously as the two signatory parties are physically located in the same room; therefore the notion of simultaneous legal binding is natural. In contrast, in digital contracts, simultaneity is lost the because the contract is signed remotely using contract signing protocols [29]. To mention another example, in non–automatic contracts a signature is a graphic component, made from some chemical substance, on its own right as it can be seen independently of documents. This notion of signature does not exist in digital contracts; it is replaced by the concept of signed document. For example, a program can be used for verifying if a document is either signed or not signed by Alice but there is no means of seeing Alice's signature separate from a document, say, before the signs.

To explain these and other issues we will use the contract example shown in Fig. 8 to contrast the terminology. The example is borrowed from [28] where the Solidy code is also provided. We have left out some of the functions to simplify our discussion.

Fig. 9 shows the life cycle of contract example of Fig. 8. We have labeled the boxes with the terms used by lawyers and software engineers.

Let us analyse Fig. 9 from the technical perspective.

- **Contract in natural language:** The life cycle of the digital contract begins in box 8 where Alice and Bob have a contract written in natural language, that is the text shown in Fig. 8, presumably drawn up by a lawyer. Notice that the text is not signed by Alice or Bob under holographic signatures.
- **Digital contract coding:** this stage consists of two steps: the coding of the digital contract which takes place in box 9 and the testing of the digital contract which takes place in rhombus 10. Notice that for the sake of simplicity Fig. 4 does not show the testing step. A failure to satisfies the testing requirements takes the development back to the contract written in natural language (box 8) where Peter the programmer along with Alice and Bob review the contract text. Though not explicitly shown in the figure, it is quite possible that the contract text is altered, for example, to correct ambiguities that prevent Peter from writing a sound digital contract.
- **Digital contract debugging:** As shown on the left side of the figure, the aggregation of steps 8, 9 and 10 is known as digital contract debugging as it is aimed at uncovering and removing bugs (errors) from the code of the digital contract written in a programming language and the contract text written in natural language. In practice, the testing requirements are imposed by the users of the digital contract (Alice and Bob in this example). Additional requirements are likely to be imposed by other parties like the owners of the execution envi-

Alice and Bob bet about the weather.

This is an contract between Alice Wonder (bettor 1) and Bob Good (bettor 2).

Signature place:
Florence

Signature date:
04 Nov 2021

Alice's personal data:
Full name: Alice Wonder
Home address: 2 Art street

Bob's personal data:
Full name: Bob Good
Home address: 5 Rich Street

Clause 1: Alice and Bob agree to bet about midday temperature in New York City on 30 Dec 2021.

Clause 2: They will hire a programmer (Peter) to write a digital contract to collect their bids and to pay the money to the winner.

Clause 3: Alice shall use her discretion to place a bet of 20 dollars with the digital contract by 15 Nov 2021.

Clause 4: Bob shall use his discretion to place a bet of 10 dollars with the digital contract after Alice's and by 30 Nov 2021.

Clause 5: Alice and Bob shall use their discretion to cancel the bet by 30 Nov 2021. The digital contract will refund the betting money to Alice and Bob.

Clause 6: The digital contract will determine the temperature and send the money to the winner.

Alice's signature:

Bob's signature:

Figure 8. A text for coding a digital contract between Alice and Bob.

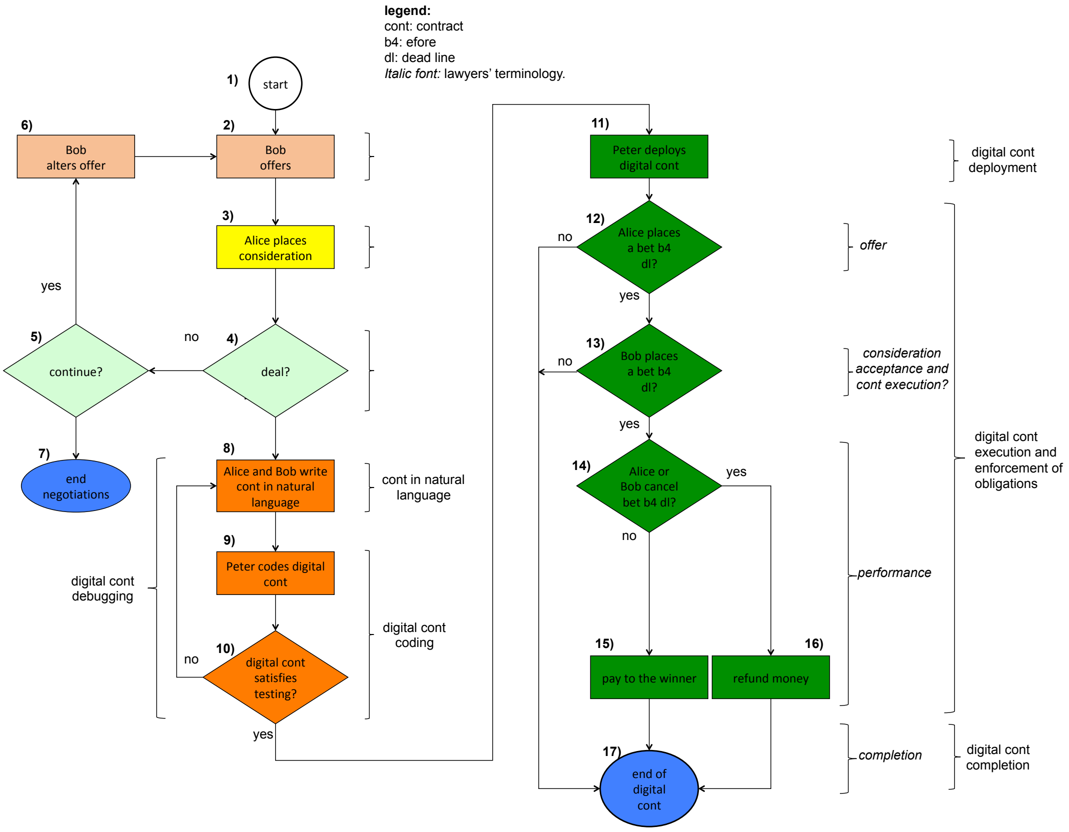


Figure 9. Life cycle of a digital contract between Alice and Bob.

ronment where the digital contract will be deployed and standardization bodies like governments to ensure safe operation of the digital contract.

- **Digital contract deployment:** Once the digital contract satisfies all the testing requirements, it is ready for deployment on an execution environment, for example, on the Ethereum blockchain like in particular example. In box 11 Peter the programmer deploys the digital contract and leaves it ready for Alice and Bob to execute.
- **Digital contract execution and enforcement of obligations:** boxes 12–16 represent the execution of the digital contract, in computer engineers' terminology. The rights, obligation and prohibitions stipulated in the contract are enforced by the digital contract. The digital code allows the execution only of operations that are contract compliance. The enforcing mechanism used by the digital contract is similar to the one shown in Fig. 7.
  - Rhombus 12 represents Alice's right to place a bet and corresponds to clause 3 of the text shown in Fig. 8. The digital contract allows Alice to execute a "bet" operation as long as the operation is executed before the deadline. The contract terminates if Alice does not exercise her right to bet.
  - Rhombus 13 represent Bob's right to place a bet and corresponds to clause 4 of the text shown in Fig. 8. The digital contract allows Bob to execute a "bet" operation as long as the operation is executed before the deadline. The contract terminates if Bob does not exercise his right to bet.
  - Rhombus 14 represents Alice's and Bob's rights to cancel the bet and corresponds to clause 5 of the text shown in Fig. 8. The digital contract allows Alice to execute a "cancel" operation as long as the operation is executed before the deadline.
  - Box 16 represents the refund of money where either Alice of BoB cancel the bet and corresponds to clause 5 of the text shown in Fig. 8. The digital contract automatically refund the bet money to Alice and Bob
  - Box 15 represents payment to the winner and corresponds to clause 6 of the text shown in Fig. 8. Firstly, the digital contract automatically reads the current temperature (for example from the web page of weather service). Next, it automatically determines who the winner is and send the money to Alice or Bob and and terminates, that is, it progresses to ellipse 17.
- **Digital contract completion:** The digital contract completes when the execution of its code reaches ellipse 17. In this stage the digital contract is inactive in the sense that it is still stored in it execution environment (for example on the Ethereum blockchain) but it no longer reacts to operations initiated by Alice or Bob.

Let us analyse Fig. 9 from the legal perspective which is captured by boxes 12–17.

- *Offer:* Rhombus 12 corresponds to clause 3 of the text shown in Fig. 8. Alice's act of betting 29 dollars represents Alice's offer that Bob can accept or reject. In legal contract terminology 20 dollars is an offer.

  Betting is a right that the contract grants to Alice. She can either place and offer and progress to box 13 or abandon the negotiation and made the digital contact progress to ellipse 17 where it terminates.
- *Consideration, acceptance and contract execution:* Rhombus 12 corresponds to clause 4 of the text shown in Fig. 8. Bob's act of betting 10 dollars represents the acceptance of Alice's offer. In legal contract terminology 10 dollars is a con-

sideration. Betting is a right that the contract of Fig. 8 grants to Bob. He can abandon the negotiation and made the digital contract progress to ellipse 17 where it terminates. Alternatively, Bob can respond with a consideration which would in legal terminology *execute* the legal contract. Bob's act of executing the "bet 10 dollars" operation electronically is equivalent to signing the contract by hand using ink on paper and a holographic signature. Bob's acceptance of Alice's offer progresses the digital contract to box 14 where the *performance* begins.

- *Performance:* The performance of the contract is captured by rhombus 14 and boxes 15 and 16. Notice that rhombus 14 grant the right to Alice and Bob to cancel the bet. In this example, we assume that the cancellation operation is initiated directly by Alice or Bob, that is, it involves human intervention. However, there are no technical difficulties in programming Alice's and Bob's devices to cancel automatically, for example, depending on their current bank balances. The performance captured by boxes 15 and 16 is entirely automatic. It does not require Alice's or Bob's intervention.
- *Completion:* The legal contract completes in ellipse 17.

Tab. 1 present a summary of the terminology used by layers and computer engineers.

Comments 6: Automatic execution means you do not need enforcement. Consider where I have to pay you X amount and you deliver to me apples, executions is adhering to the rights and obligations, enforcement is where a party need to be forced to do them.

Comment 4: Are you confusing execution with enforcement? Execution: adhering to terms of the contract by the parties. Enforcement is usually what you do when the party does not execute the contract.

2.5. *Delete these Section?*

The flow chart shown in Fig (we need a figure) describes the typical workflow of a business contracts. It includes the following steps.

(1) Informal negotiations.
(2) Letter of intent.
(3) Signature (contract formation or contract execution or conclusion?)
(4) Start of contracts.
(5) Contract development (corresponds to contract performance?)

See[30].

In [24] they consider Enforcement, Execution, and Performance. In [31] they consider that the life cycle of a smart contract consists of four stages: Creation, freezing, execution, and finalization.

In [21] they explain that in common law a contract is created through 1) offer, 2) acceptance, and 3) consideration.

Fig. **??** shows the stages of the formation of a civil contract. The contracts involves Alice and Bob where Bob takes the lead to offer.

Technical people distinguish between four phases of the life cycle of smart contracts: Contract creation, deployment, execution, and completion [32].

There are also semantic differences between the terminologies used by lawyers and programmers. The word execution of a contracts means ... (see p8 [19]).

(6) Conflicts (corresponds to contract breach?): yes or no?
   (a) If there is a conflict take offending party to court.
   (b) Either resume contract or complete it after conflict resolution.

Terminology **Term**	Lawyers	Computer engineers
Contract	A formal agreement written in natural language and enforceable in court.	A piece of executable code that enables or disable the execution of operations when certain conditions hold.
Contract formation	A formal agreement written in natural language and enforceable in court.	A piece of executable code that enables or disable the execution of operations when certain conditions hold.
Contract coding	No equivalent term.	The translation of a contract written in a natural language to an equivalent written in a programming language.
Contract deployment	—	The installation of the executable code that represents the contract in an execution environment to have it ready for execution.
Contract execution	The act of signing contract after mutual agreement.	The act of launching (running, executing) the executable code that represents the contract.
Contract transaction	Action is the closes term.	An operation (eg. pay, cancel, deliver, etc.) to be executed to fulfill a contract.
Contract enforcement	A legal dictation to force an individual that has breached a contract to restore the damage.	The automatic execution of all the operations included in the terms of the contracts to fulfil the contract.
Operation enforcement	No equivalent term???	The digital contract intrusively allows, initiates and prevents the execution of operations related, respectively, to rights, obligations and prohibitions.
Operation monitoring	Collection of documentary evidence of contract actions.	The digital contract automatically and non–intrusively observes and records the operations (eg, pay) executed by the signatories for post–morten examination.
Contract performance	The fulfillment of the terms of the contract.	A measure of the efficiency of a contract in terms of execution speed and amount of resources consumed (memory, disk, network width.)
Contract termination	The end of the performance of a contract.	A contract terminates (completes) when it stops its execution because the contract has been fulfilled (no obligations are pending).

Table 1. Comparison of contract terminologies.

(7) Contract enforcement (What is contract enforcement for lawyers?).
(8) Close contract.

Some questions to keep in mind to elaborate the discussion:

(1) Do contracts in the conventional world exist now?
(2) Can we currently conduct business without unintentionally involving automatic contracts to a greater or lesser extent. For example, an informal agreement conducted face to face seems to me less commitment than an informal agreement conducted over the phone as the parties have evidence of the call; an informal agreement conduced over whatsapp messages bear a commitment because of the written evidence left on the mobile phone.
(3) We can use the steps listed in contract workflow to argue in conclusions why smart contract are not contracts.

3. Signatures and contracts

Several types of signatures are currently in use in e–commerce. From the perspective of the technology used to place them they can be grouped into three categories:

- Holographic signatures: also called hand written signatures are signatures placed on paper using chemical ink.
- Electronic signatures: are signatures placed by means of electronic technologies such as fax, email, pin numbers, accept buttons, pub–priv key and other technologies [33].
- Pub–key based signatures: are the most sophisticated electronic signatures; therefore they are normally treated separately under the name of digital signatures or cryptographic signatures. In our work we will use the term electronic non–digital signatures we refer to any electronic signature with the exception of priv–key based signatures.

3.1. *Hand–signed and cryptographically–signed contracts*

With regard to the type of signature placed on a legal contract we distinguish between two classes of contracts. See Fig. 10.

- Hand–signed digital contracts.
- Cryptographically–signed digital contracts.

Hand–signed digital contracts are protected by the placement of a signature placed in the conventional manner: by and on a piece of paper, for example on a A4 sheet or a digital signature (as in where you have an image of your signature and place it in a pdf, or where you draw it on a tablet!). As we will explain below, the hand–signed document is used to derive the executable code that is used for enforcing the contractual rights and obligations. Fig. 11, shows a traditional contract signed under Alice's and Bob's hand–written signatures. Cryptographically–signed digital contracts are protected by the placement of a cryptographic signature, which is a string of characters (say 256 of them) produced from a cryptographic key. We return to cryptographic signatures in Section **??**.

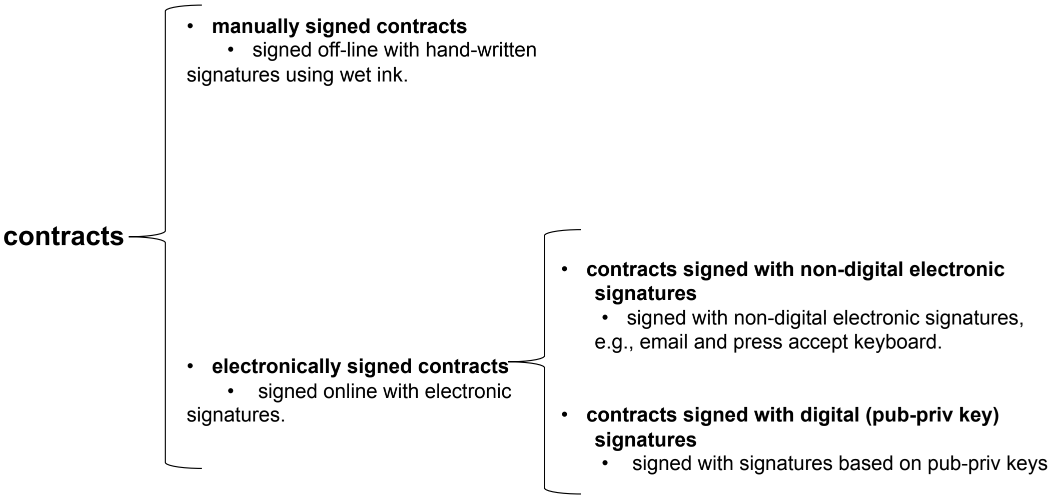

contracts
manually signed contracts
signed off-line with hand-written signatures using wet ink.
electronically signed contracts
signed online with electronic signatures.
contracts signed with non-digital electronic signatures
signed with non-digital electronic signatures, e.g., email and press accept keyboard.
contracts signed with digital (pub-priv key) signatures
signed with signatures based on pub-priv keys

Figure 10. Contract categories from the perspective of type of signature.

4. Categorisation of non–automatic contracts

The salient feature of non–automatic contracts is that they are *performed* manually, that is, the execution of their *actions* does not involve the execution of computer code. There is a large variety of contracts that fall within this category; to appreciate their strengths and limitations, it might help to group them into two large sub–categories: namely, manually signed contracts and online signed contracts.

We will use examples to explain the definition of non–automatic contracts.

- Manually signed contracts: within this category fall contracts signed on pieces of paper under hand–written signatures.
- Online signed contracts: Within this category fall contracts signed online with electronic non—-digital signatures, for example, by means of email or through ”press accept to agree to terms and conditions”.

4.1. *Manually signed contracts*

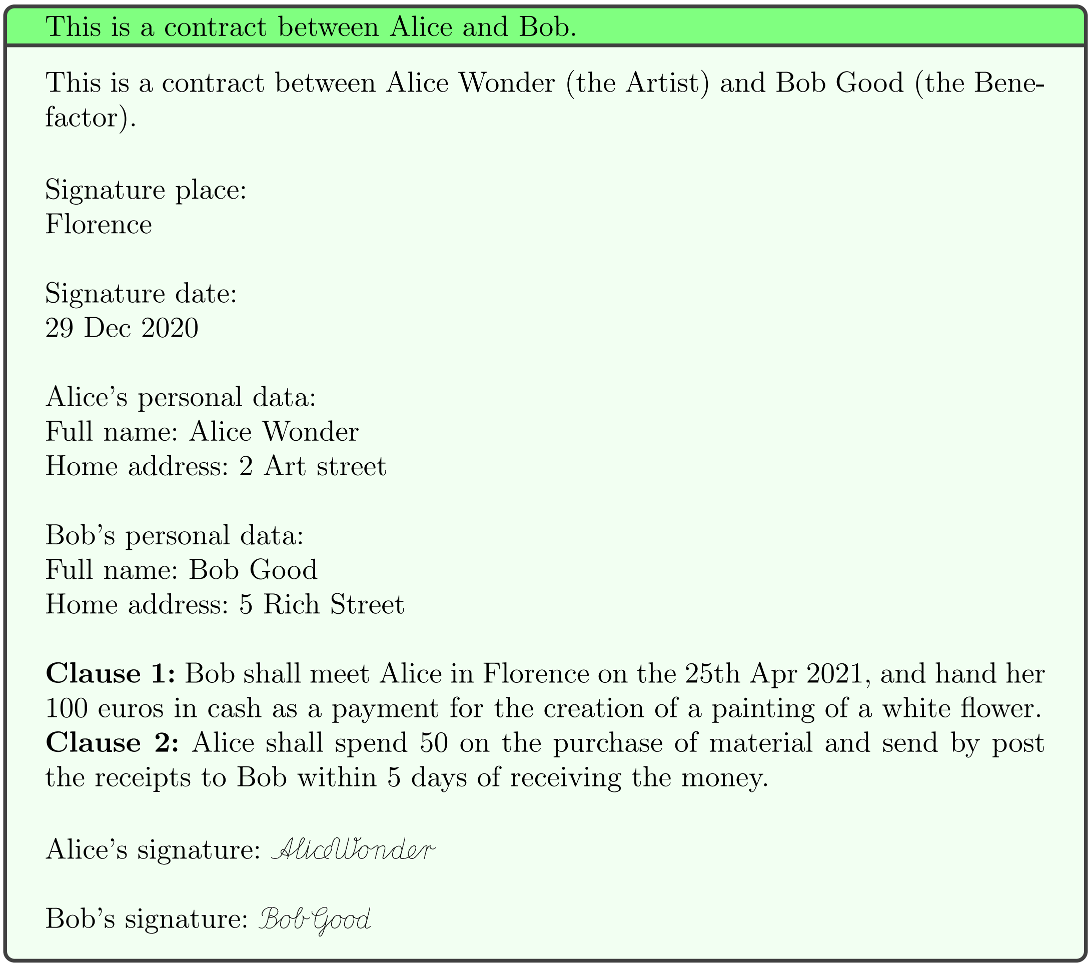
We define a manually signed contracts as a legal contract negotiated, written and agreed exclusively in human language on a piece of paper, signed by means of hand signatures and *performed* without the assistance of computer technology. It may be digital in format. By the latter we mean that the signed document is and remains the only piece of information that commits the participants. Fig. 11 shows an example of an hypothetical manually signed contract written in English and agreed upon between Alice and Bob (the signatories).

Hazem, clarify?

Contract format: Two copies of the contract are printed on A4 or letter paper and signed by Alice and Bob by hand-written signatures in a face-to-face meeting.

Contract storage: The signed A4 sheets are stored in traditional file cabinets.

Contract execution: Fig. 12 shows the execution of a manually signed contract. Notice that Alice or Bob are shown without computer devices. This is to emphasise that they do not use computer technology to execute the contract. In fact, since the operations specified in the clauses of the contract involve the manipulation of physical objects (cash and paper documents, respectively), this contract cannot be executed



This is a contract between Alice and Bob.

This is a contract between Alice Wonder (the Artist) and Bob Good (the Benefactor).

Signature place:
Florence

Signature date:
29 Dec 2020

Alice's personal data:
Full name: Alice Wonder
Home address: 2 Art street

Bob's personal data:
Full name: Bob Good
Home address: 5 Rich Street

Clause 1: Bob shall meet Alice in Florence on the 25th Apr 2021, and hand her 100 euros in cash as a payment for the creation of a painting of a white flower.
Clause 2: Alice shall spend 50 on the purchase of material and send by post the receipts to Bob within 5 days of receiving the money.

Alice's signature: *AliceWonder*

Bob's signature: *BobGood*

Figure 11. Contract written in English and signed manually by Alice and Bob.

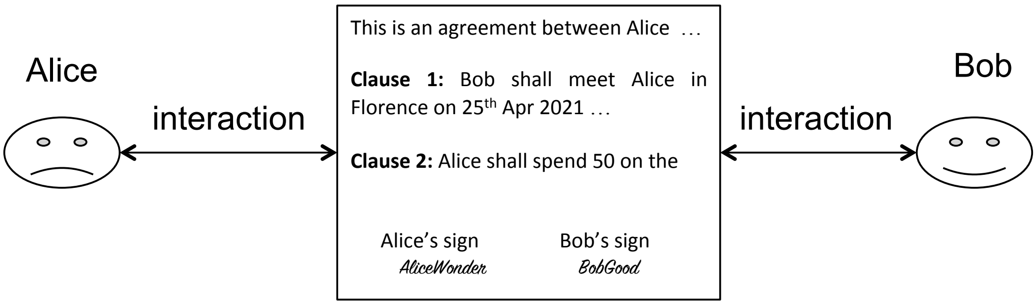

Alice
interaction
This is an agreement between Alice …
Clause 1: Bob shall meet Alice in Florence on 25th Apr 2021 …
Clause 2: Alice shall spend 50 on the
Alice's sign
AliceWonder
Bob's sign
BobGood
interaction
Bob

Figure 12. Execution of manually signed contract.

automatically. It is a natural manually–executable contract. In the figure, the arrows labeled interaction indicate that Alice and Bob interact with the piece of A4 signed paper in the traditional manner.

We clarify that there might be examples of contracts that include operations that can be automated; however, if Alice and Bob decide to execute them in the traditional manner they will not be converted to digital code and therefore fall within the category of contracts executed manually.

4.2. *Online signed contracts*

We define an online signed contract as a legal contract negotiated and signed by means of an electronic non–digital signature and *performed* without the automation of any of it contractual operations. Fig. 13 shows an example of an hypothetical online signed contract signed by Bob through an email message sent from his device and signed by Alice through an email response sent from her device. We emphasise that the devices are used by Bob and Alice only for signing the contract.

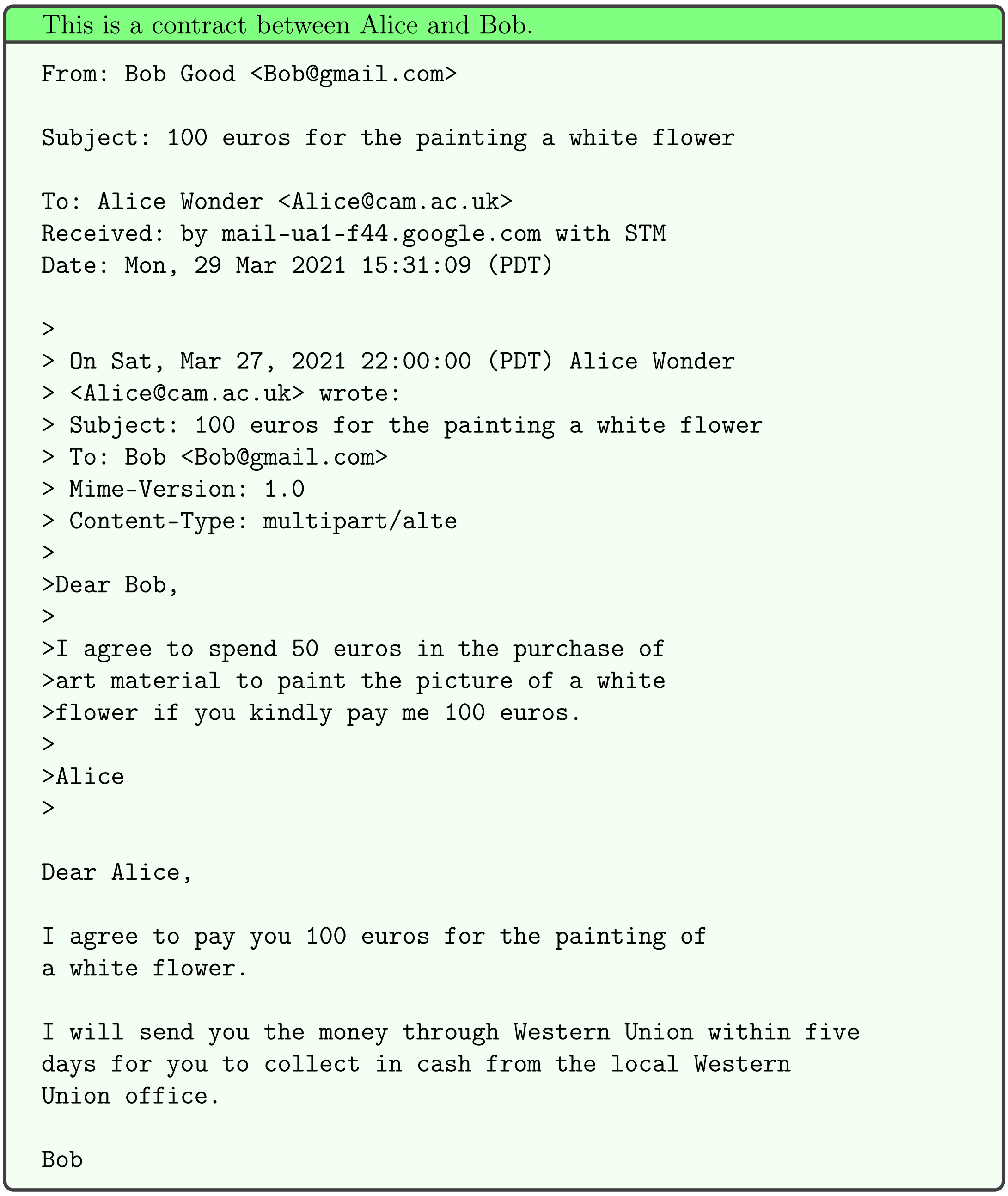
As shown in Fig. 14 the contract is *performed* (executed, in computer science parlance) without any technology, Alice is expected to call in the local Western Union office to collect Bob's donation in Cash. In other words, the operation, ”pay 100 to Alice” is not *performed* automatically.

The question that arises here is the validity of Bob's signature. Observe that Bob sends his email from a gmail.com account, thus Bob@gmail.com is a signature, one can ask, who certifies it? It is well known that it is not difficult to open an email account in gmail and other free email services like Hotmail under false personal data such as false name, age, sex, location, etc. From the perspective of trust, Bob's email address is different from Alice's Alice@cam.ac.uk which was granted to her by the University of Cambridge–a trusted institution.

We believe that these question needs an in deep analysis; we argue that to say that emails are accepted as legal electronic signatures is misleading; perhaps not all of them are.

5. Categorisation of automatic contracts

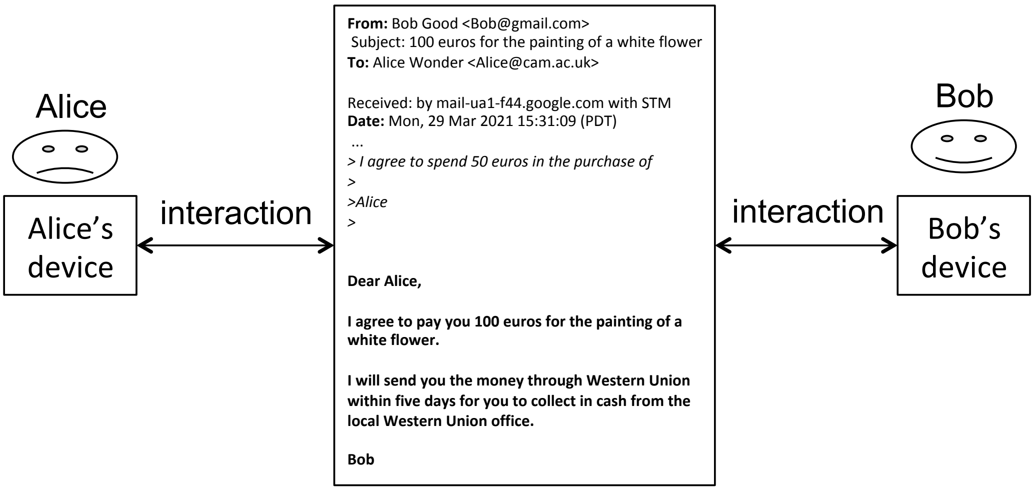
Hazem: Automatic execution means you do not need enforcement. Consider where I have to pay you X amount and you deliver to me apples, executions is adhering to the rights and obligations, enforcement is where a party need to be forced to do them



This is a contract between Alice and Bob.

```
From: Bob Good <Bob@gmail.com>

Subject: 100 euros for the painting a white flower

To: Alice Wonder <Alice@cam.ac.uk>
Received: by mail-ua1-f44.google.com with STM
Date: Mon, 29 Mar 2021 15:31:09 (PDT)

>
> On Sat, Mar 27, 2021 22:00:00 (PDT) Alice Wonder
> <Alice@cam.ac.uk> wrote:
> Subject: 100 euros for the painting a white flower
> To: Bob <Bob@gmail.com>
> Mime-Version: 1.0
> Content-Type: multipart/alte
>
>Dear Bob,
>
>I agree to spend 50 euros in the purchase of
>art material to paint the picture of a white
>flower if you kindly pay me 100 euros.
>
>Alice
>

Dear Alice,

I agree to pay you 100 euros for the painting of
a white flower.

I will send you the money through Western Union within five
days for you to collect in cash from the local Western
Union office.

Bob
```

Figure 13. An online signed contract written in English.



From: Bob Good <Bob@gmail.com>
Subject: 100 euros for the painting of a white flower
To: Alice Wonder <Alice@cam.ac.uk>

Received: by mail-ua1-f44.google.com with STM
Date: Mon, 29 Mar 2021 15:31:09 (PDT)
...
> *I agree to spend 50 euros in the purchase of*
>
>*Alice*
>

Dear Alice,

I agree to pay you 100 euros for the painting of a white flower.

I will send you the money through Western Union within five days for you to collect in cash from the local Western Union office.

Bob

Figure 14. *Execution* of an online signed contract.

Hazem: You shift around using execution and enforcement, lets separate them out early on and use execution which thereby limits enforcement, not fully but sometimes

Legal contracts can be classified into traditional contracts (no automation) and digital contracts with different degrees of automation. In this section, we will discuss five sub–classes of digital contracts that exist on the spectrum between no automation and full automation:

- Contracts written in a natural language.
- Contracts written solely in code.
- Contracts written in code but supported by a traditional contract.
- Hybrid contracts.
- Multimedia contracts

In [34] they distinguish four types of contracts...

5.1. *Natural language contract*

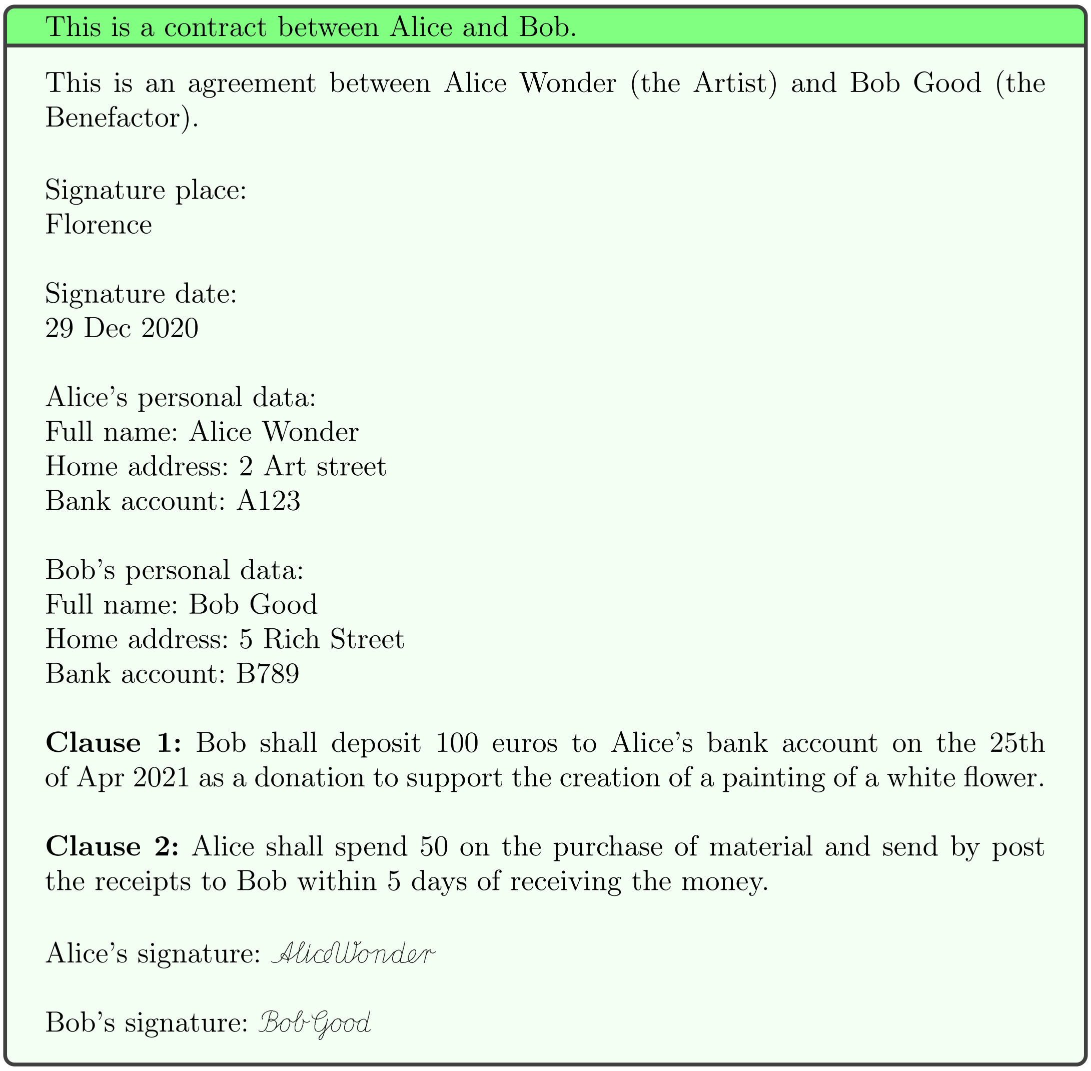
We define a natural language contract as a contract that i) is written in a natural language (for example, English) on a piece of paper (for example A4) ii) signed under hand–written signatures and iii) includes one or more contractual commitments that are enforced automatically by means of executable computer code.

Hazem: Would natural language contract not include any commitments enforced by computer code??

An example of a natural language contracts shown in Fig. 15. We will use this figure to highlight the salient features of this class of digital contracts. In the example, only clause 1 can be translated into executable computer code, clause 2 can be executed only in the traditional manner as it stipulates that Alice needs to send by post a receive printed on paper. Accordingly, this contract can be enforced automatically only partially.

Contract format: Two copies of the document shown in Fig. 15 are printed on A4 paper and signed by Alice and Bob by hand-written signatures in a face–to–face meeting.

Contract storage: The signed A4 sheets are stored in traditional file cabinets.



This is a contract between Alice and Bob.

This is an agreement between Alice Wonder (the Artist) and Bob Good (the Benefactor).

Signature place:
Florence

Signature date:
29 Dec 2020

Alice's personal data:
Full name: Alice Wonder
Home address: 2 Art street
Bank account: A123

Bob's personal data:
Full name: Bob Good
Home address: 5 Rich Street
Bank account: B789

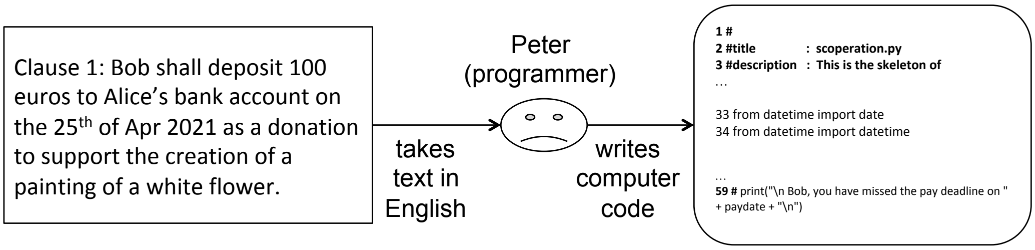
Clause 1: Bob shall deposit 100 euros to Alice's bank account on the 25th of Apr 2021 as a donation to support the creation of a painting of a white flower.

Clause 2: Alice shall spend 50 on the purchase of material and send by post the receipts to Bob within 5 days of receiving the money.

Alice's signature: AliceWonder

Bob's signature: BobGood

Figure 15. A natural language contract.



Clause 1: Bob shall deposit 100 euros to Alice's bank account on the 25th of Apr 2021 as a donation to support the creation of a painting of a white flower.

```
#
#title          :  scoperation.py
#description    :  This is the skeleton of
...

from datetime import date
from datetime import datetime


...
# print("\n Bob, you have missed the pay deadline on "
+ paydate + "\n")
```

Figure 16. Conversion of a contract clause into executable computer code.

Contract execution: Alice and Bob agree that they will use a piece of executable code to perform Bob's obligation stipulated in Clause 1. They also agree that Alice will perform her obligation stipulated in Clause 2 manually. Executable code for performing Clause 1: The executable code that is used for enforcing Clause 1 is developed, deployed and executed as follows:

(1) Alice and Bob agree to hire Peter (a professional programmer or software engineer) to produce the executable code that will try to enforce the performance of Bob's obligation expressed in Clause 1, automatically. The coding of the contract is conceptually shown in Fig. 16.
(2) Peter translates the English text of Clause 1 into a programming language. At this level of discussion, the specific language is irrelevant, for instance, he can use a blockchain language such as Bitcoin's script, and Ethereum's Solidity or a general purpose programming language such as C++ or Python. For the sake of illustration we will use the latter. The code shown in List 1 is far from being an accurate translation of Clause 1. Its main purpose is to give a rough idea about what a contractual clause looks like when translated into executable code.
(3) Peter produces the source code CClause1.py shown in List 1. For the sake of this discussion, let us say source codes like CClause1.py are executable in the sense that they can be executed by a computer either directly if presented to an interpreter or after compiling them with the help of a compiler.
(4) CClause1.py can be regarded as a mechanism (a software tool or agent) that blindly performs the operation stipulated in Clause 1. CClause1.py has no notion of contracts; in this manner, the potential disputes that emerge from the execution of the obligation that CClause1.py tries to execute will be resolved with the assistance of the signed document shown in Fig. 15 which will be retrieved from a file cabinet.

Presumably, before delegating the responsibility of enforcing the execution of Clause 1, Alice and Bob will thoroughly verify that CClause1.py does exactly what Clause 1 stipulates. They will thoroughly test (possibly with the help of auditors) that Peter translated Clause 1 into CClause1.py correctly. Equally importantly, after accepting CClause1.py as valid, Alice, Bob and Peter will resort to cryptography mechanisms (for example, hashes, also called digests) to ensure that nobody can alter CClause1.py without Alice, Bob and Peter being able to detect the alteration.

It is worth emphasizing that, as shown in Fig. 17, the execution of CCluase1.py takes place in a single computer, that is, the code is executed in a centralized manner. Decentralised execution of Cclause1.py is also possible and a better alternative in some applications. We will discuss this approach and the issues that it raises in subsequent sections; for the time being, the centralized approach that we take in Fig. 17 is enough to explain the core ideas about contracts. It is also important to note that, the code is not directly bound to the document shown in Fig. 15, in the sense that there are no automatic mechanisms (for example, a URL link or a cryptographic relationship) to relate the execution of the code to the text of clause 1.

Listing 1 Pseudocode of clause 1 of contract shown in Fig. 15

```
#
 #title            : scpayperation.py
 #description      : This is the skeleton of a smart contract that
 #                 : shows the execution of a payment operation.
 #                 : It is only a toy to show the ideas behind
 #                 : executable codes that are called smart contracts.
 #                 : It is meant to execute the following clause:
 #                 : Clause 1: Bob shall deposit 100 euros to
 #                 : Alice's bank account on the 25th of Apr 2021 as a
#                 : donation to support the creation of a painting of
#                 : a black.
#author           : Carlos Molina-Jimenez
#date             : 28 Dec 2020 (Computer Laboratory Univ of Cambridge)
#version          : 1
#notes            :
#python_version   : Python 3.7.4
 #
 #compile and run  : python scpayoperation.py
#                 :
#result           : The program is meant to be launched on
 #                   the 25th Apr 2021. After launch it checks
 #                   the current date and a pay date. If it is
 #                   pay date, it opens Bob's (the payer)
 #                   bank account, withdraws 100 euros and
 #                   deposits them to Alice's (the payee)
 #                   bank account
 #
 # online run      : I have run it succesfully on
 #                   https://repl.it/languages/python3
 #=====================================================


from datetime import date
```

```
  from datetime import datetime

  aliceAcc= "A123"      # Alice's bank account
  bobAcc= "B789"        # Bob's bank account
 amount= 100            # amount to transfer from Alice to
    Bob.
 paydate= "25/04/2021" # pay date

 getDate= date.today()

 today= getDate.strftime("%d/%m/%Y")
 print("\n\ntoday is:", today)

if today == paydate:
    print("\n Bob, this code will pay Alice today, right
       now!\n")
    #
   # Actual code to transfer money from Bob's account
    # to Alice's account to be included here.
    #
    print(str(100) + " euros have been transferred from Bob
       's to Alice's account\n")
 else:
   now = datetime.now()
   deadline= datetime.strptime(paydate, "%d/%m/%Y")
   if now < deadline :
       print("\n Bob, you have a pending deadline to pay on
           " + paydate + "\n")
   elif now > deadline:
      print("\n Bob, you have missed the pay deadline on "
         + paydate + "\n")
```

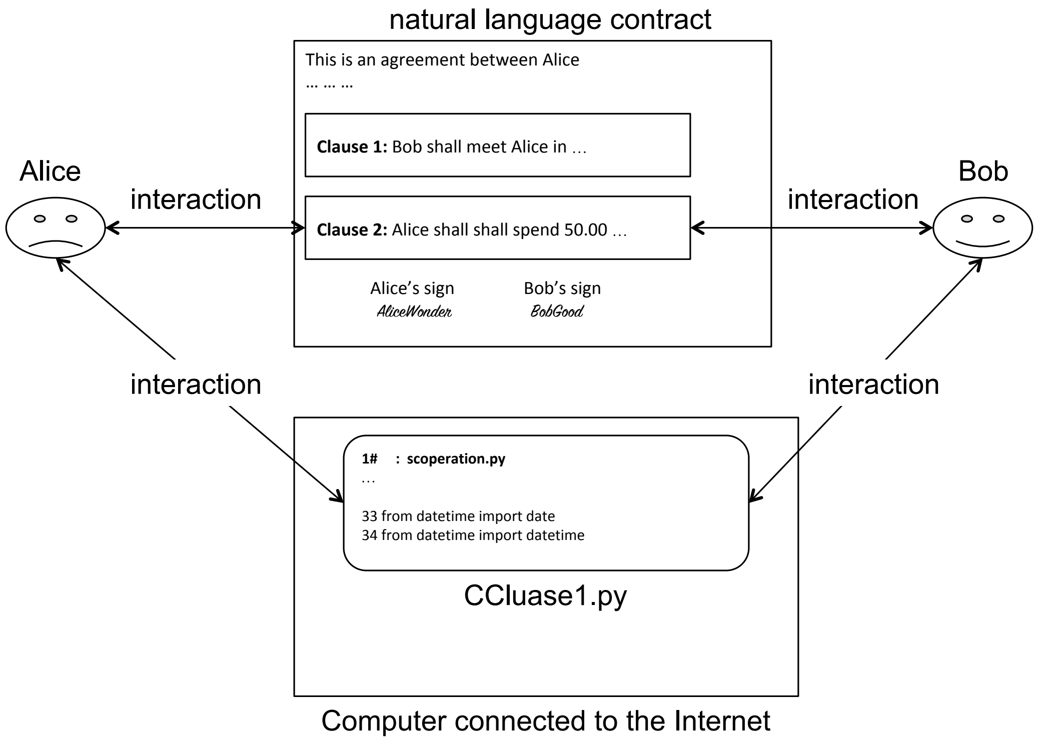
5.2. *Contract recorded solely in code*

We define a contract that is recorded solely in code as a contract with clauses that are expressed only in executable code and bears neither hand–written or cryptographic signature.

Fig. 18 shows an example of a document that is used for producing the contract code shown in List 2. Notice that in this example, both Clause 1 and Clause 2 are amenable to translation to computer executable code.

Contract format: Alice and Bob agree to the commitments stipulated in clause 1 and clause 2. In addition, they agree to trust each other; accordingly they do not sign the document. This is a sensible solution when Alice and Bob know each other or when the difficulties of hand–signing the document out weight the benefits, for example, because Alice and Bob are located in different countries. From a legal perspective, so long as the requirements of contract formation are met; offer, acceptance, consideration, and intention to create legal relations then a contract has been formed.

Contract storage: No paper version of the agreement is used. The only representation of the agreement is the executable code that implements clause 1 and clause


natural language contract
This is an agreement between Alice
… … …
Clause 1: Bob shall meet Alice in …
Clause 2: Alice shall shall spend 50.00 …
Alice's sign
AliceWonder
Bob's sign
BobGood
Alice
interaction
Bob
interaction
interaction
interaction
CCluase1.py
Computer connected to the Internet

```
#    : scoperation.py
…

from datetime import date
from datetime import datetime
```

Figure 17. Execution of a contract written in natural language.

2.

Contract execution: The contract is executed automatically, that is, without any human intervention or action. Automatic execution is possible only when, like in this example, all the clauses included in the agreement are amenable to translation to executable code.

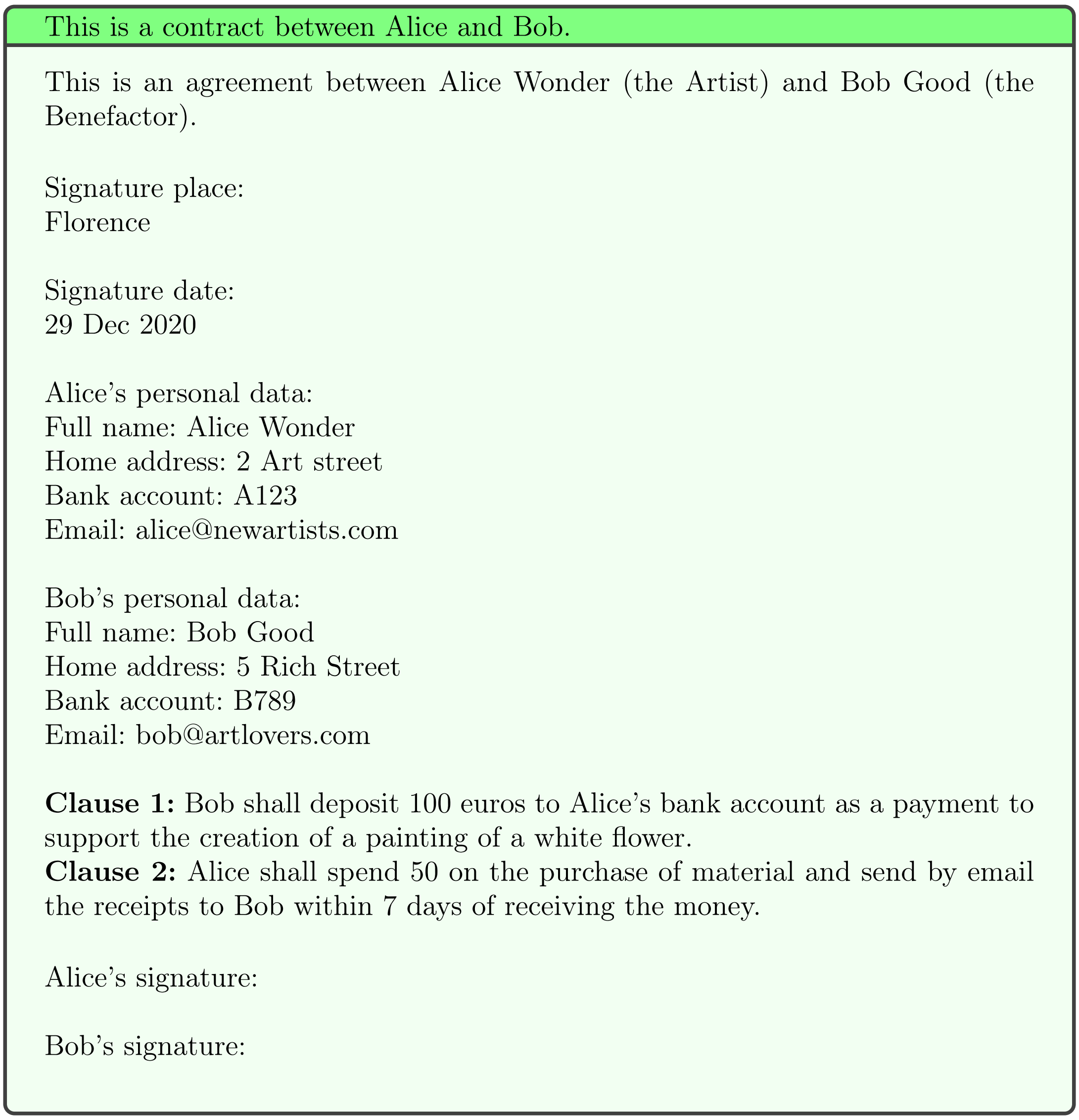
Hazem: Is it true there's no human intervention? E.g., need to move money to escrow or link the system with my bank account…

A limitation that might not be acceptable in some applications is that the execution of the code is not bound to any legal contract signed by Alice and Bob. This in the event of dispute, the Alice and Bob can only resort to the executable code to revolve it or to background evidentiary information to demonstrate the commitments and terms of the contract. Nevertheless, so long as the computer code reflects the requirements of contract formation then a contract has been formed. This, however introduces an additional query, namely that the computer code must, in some sense be translated in order to ascertain the true linguistic meaning. This is a shortcoming and will not be as effective as it is thought to be, causing reliance on background information and agreement, such as conversations, emails and the like, should they exist.

Executable code for performing Clause 1 and Clause 2: The executable code for the enforcement of Clause 1 and Clause 2 is produced by a professional programmer, for example, Peter. The process is similar to the one shown in Fig. 16, except that this time has to translate both Clause 1 and Clause 2 into code. The results are shown in List 2.

Listing 2 Code of contract shown in Fig. 18

```
#
#
```



This is a contract between Alice and Bob.

This is an agreement between Alice Wonder (the Artist) and Bob Good (the Benefactor).

Signature place:
Florence

Signature date:
29 Dec 2020

Alice's personal data:
Full name: Alice Wonder
Home address: 2 Art street
Bank account: A123
Email: alice@newartists.com

Bob's personal data:
Full name: Bob Good
Home address: 5 Rich Street
Bank account: B789
Email: bob@artlovers.com

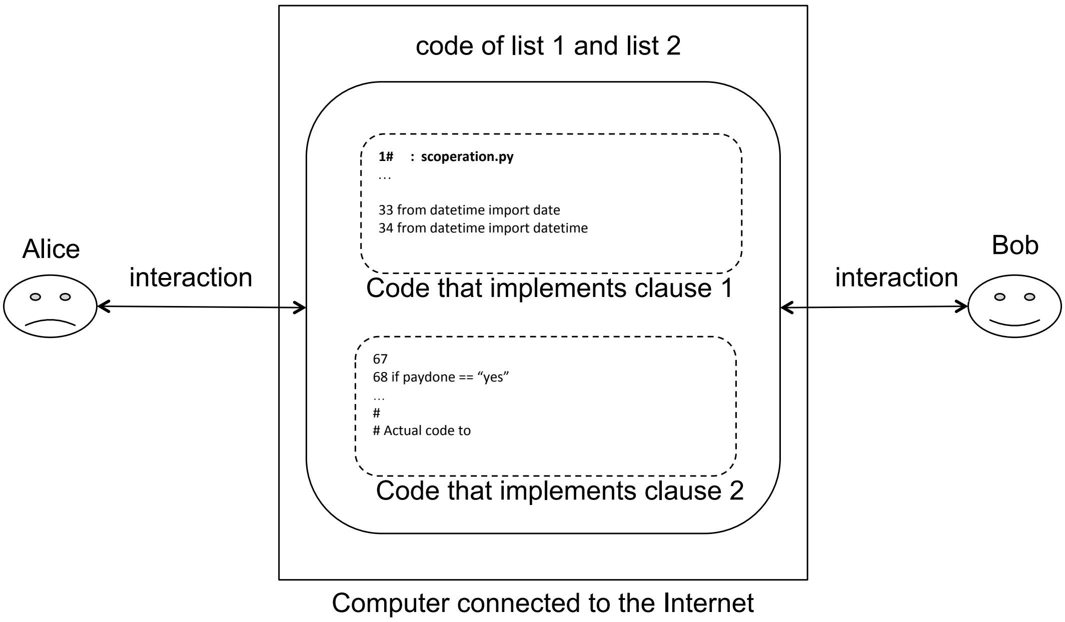
Clause 1: Bob shall deposit 100 euros to Alice's bank account as a payment to support the creation of a painting of a white flower.
Clause 2: Alice shall spend 50 on the purchase of material and send by email the receipts to Bob within 7 days of receiving the money.

Alice's signature:

Bob's signature:

Figure 18. Contract recorded solely in code.

```
#title           : scpayreceiptperations.py
#description     : This is the skeleton of a smart contract that
#                : shows the execution of a payment and send
#                : receipt operations.
#                : It is only a toy to show the ideas behind
#                : executable codes that are called smart contracts.
#                : It is meant to execute the following clauses:
#                : Clause 1: Bob shall deposit 100.00 euros to
#                : Alice's bank account on the 25th of Apr 2021 as a
#                : donation to support the creation of a painting of
#                : a black.
#                : Clause 2: Alice shall spend 50.00 on the purchase of
#                : material and send by post the receipts to Bob within
#                : 5 days of receiving the money.
#author          : Carlos Molina-Jimenez
#date            : 28 Dec 2020 (Computer Laboratory Univ of Cambridge)
#version         : 1
#notes           :
#python_version  : Python 3.7.4
#
#compile and run : python scpayreceiptoperations.py
#                :
#result          : The program is meant to be launched on
#                  the 25th Apr 2021. After launch it checks
#                  the current date and a pay date. If it is
#                  pay date, it opens Bob's (the payer)
#                  bank account, withdraws 100.00 euros and
#                  deposits them to Alice's (the payee)
#                  bank account. Next it prints that the
#                  receipts have sent sent to Bob.
#
# online run     : I have run it successfully on

#                  https://repl.it/languages/python3

#================================================

  from datetime import date
  from datetime import datetime

  aliceAcc= "A123"      # Alice's bank account
  bobAcc= "B789"        # Bob's bank account
  amount= 100           # amount to transfer from Alice to Bob.
  paydate= "28/12/2020" # pay date
  paydone= "no"         # pay not performed yet
  getDate= date.today()

  today= getDate.strftime("%d/%m/%Y")
  print("\n\ntoday is:", today)

 if today == paydate:
   print("\n Bob, this code will pay Alice today, right now!\n")
   #
   # Actual code to transfer money from Bob's account
   # to Alice's account to be included here.
   #
    print(str(100) + " euros have been transferred from
     Bob's to Alice's account\n") paydone= "yes"
 else:
     now = datetime.now()
     deadline= datetime.strptime(paydate, "%d/%m/%Y")
    if now < deadline :
       print("\n Bob, you have a pending deadline to pay on " + paydate + "\n")
     elif now > deadline:

        print("\n Bob, you have missed the pay deadline on " + paydate + "\n")

#
/*** clause 2

if paydone== "yes":
#
# Actual code to contact Alice (for ex by email) to get the
# expenses receipts and forward them to Bob (for example
# by email. The code waits for seven days for Alice's reply. If
# Alice fails to reply the code sends a notification to Alice to
# let her know about the missing deadline. This code assumes that
# everything goes smoothly.
#
```


code of list 1 and list 2
Alice
interaction
Code that implements clause 1
interaction
Bob
Code that implements clause 2
Computer connected to the Internet

```
#   : scoperation.py
...

from datetime import date
from datetime import datetime
```

```

if paydone == "yes"
...
#
# Actual code to
```

Figure 19. Execution of a contract written solely in code.

```
print("Alice, I've forward the receipts to Bob, well done!\n")
```

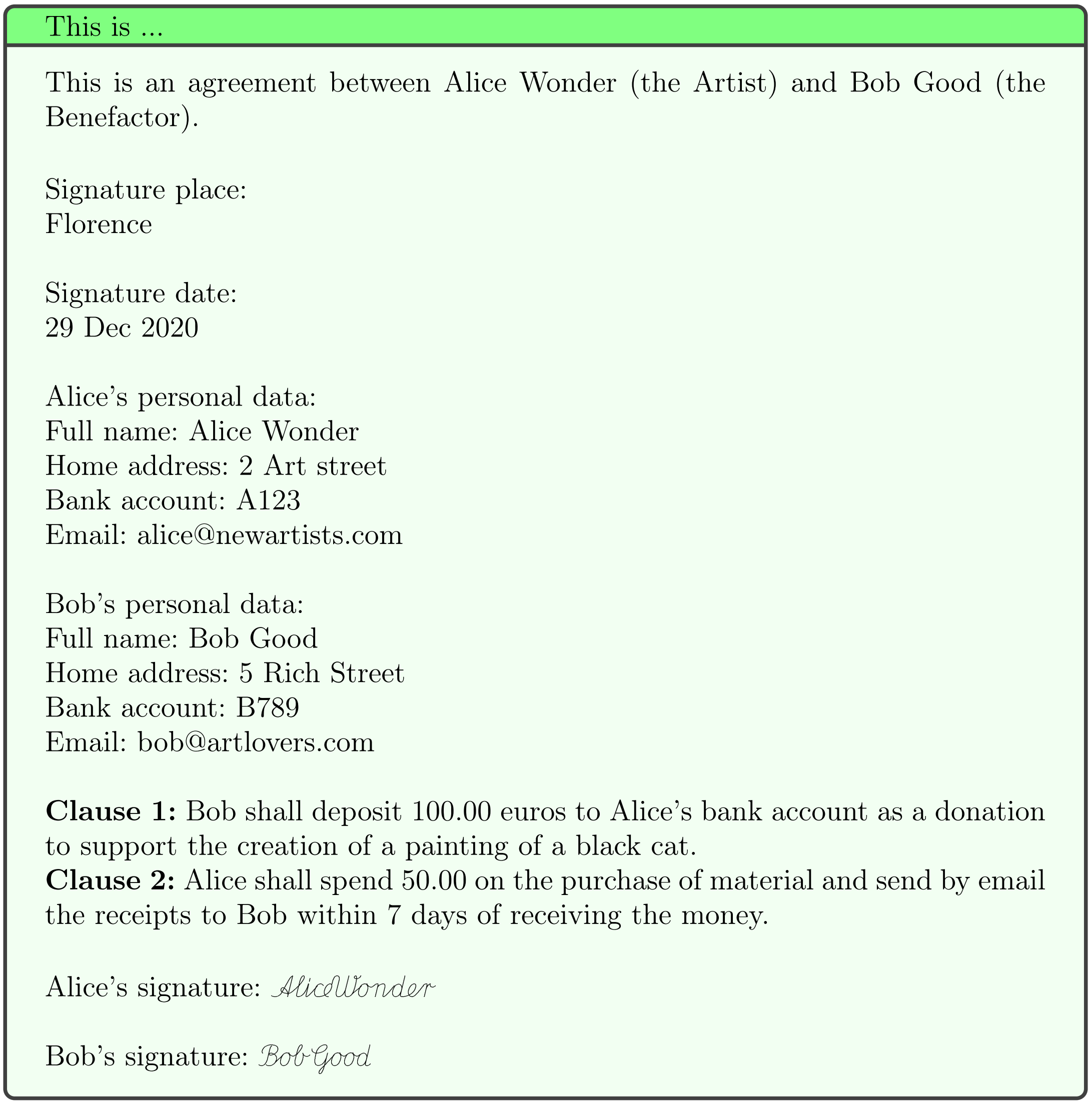
5.3. *Contract recorded in paper and in code*

We define a contract recorded in paper and code as a contract with clauses that are expressed in executable code and cryptographically bound to a hand-signed contract.

The motivation for the binding of the executable code of the clauses to a hand-signed contract is to address a salient limitation of contracts recorded solely in code. A particularity of the latter is that their code is not derived from signed documents. A close look at Fig. 18 and Fig. 19 will reveal that neither the text of the contract or the executable code that implements its clauses (clause1 and clause 2) are signed.

This particularity might be unacceptable in some applications as the contract recorded solely in code might not satisfy the requirements of the jurisdiction where the code is executed. A potential solution to this problem is to cryptographically bind the signed text of the contract to its executable code, in the following manner: Alice and Bob hand–sign the text of the contract as shown in Fig. 20.

Next the programmer (say Peter) can scan the signed paper to produce a digital document (for example in jpeg or pdf formats); let us call it signedContrText.jpeg. Next Peter can hash signedTextContr.jpeg to produce a hash (also called a digital finger print) of signedTextContr.jpeg, for example QmSNZBkCaRsAnDrAX3JgYsHm4n2aZmoRbSo. The hash guarantees that the text of the contract cannot be altered. It can also be used as an address to store the signedTextContr.jpeg document in a file system, for example in the IPFS. Equally importantly, the hash can be encoded in the executable code that executes the clauses of the contract in a manner that the code and signedTextContr.jpeg are cryptographically bound to each other. Another security measure that Alice and Bob can take is to hash the executable code after encoding the hash QmSNZBkCaRsAnDrAX3JgYsHm4n2aZmolUoR it to



This is ...

This is an agreement between Alice Wonder (the Artist) and Bob Good (the Benefactor).

Signature place:
Florence

Signature date:
29 Dec 2020

Alice's personal data:
Full name: Alice Wonder
Home address: 2 Art street
Bank account: A123
Email: alice@newartists.com

Bob's personal data:
Full name: Bob Good
Home address: 5 Rich Street
Bank account: B789
Email: bob@artlovers.com

Clause 1: Bob shall deposit 100.00 euros to Alice's bank account as a donation to support the creation of a painting of a black cat.
Clause 2: Alice shall spend 50.00 on the purchase of material and send by email the receipts to Bob within 7 days of receiving the money.

Alice's signature: AliceWonder

Bob's signature: BobGood

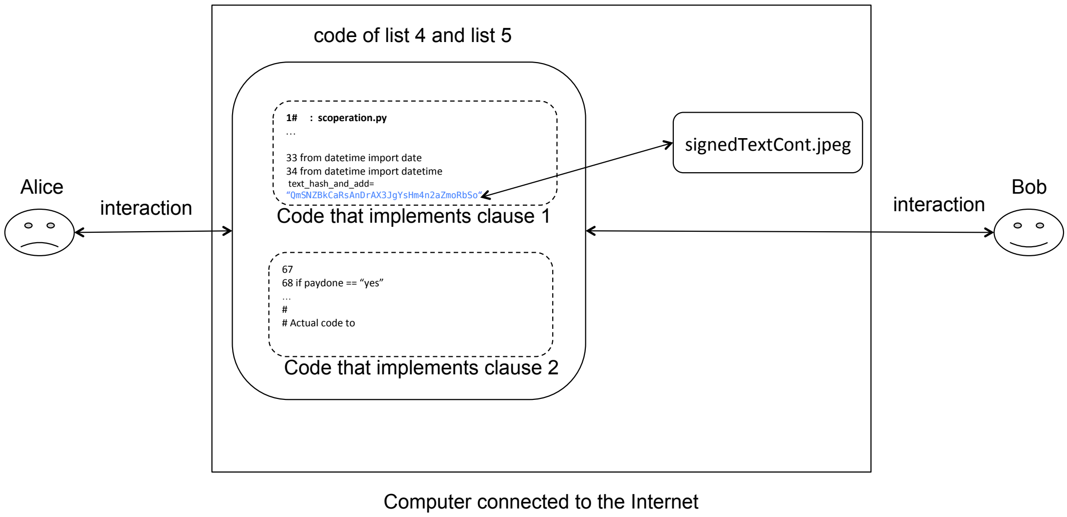
Figure 20. Contract recorded on paper and code.

guarantee that the code cannot be altered. The idea if show in List 3 where line 30 (in blue) includes the hash of signedTextContr.jpeg.

Listing 3 CClause1.py implements clause 1 and 2 of contract of Fig. 20

```
1  #
2  #
3  #title              : scpayreceiptperations.py
4  #description        : This is the skeleton of a smart contract that
5  #                    : shows the execution of a payment and send
6  #                    : receipt operations.
7  #                    : It is only a toy to show the ideas behind
8  #                    : executable codes that are called smart contracts.
9  #                    : It is meant to execute the following clauses:
10 #                    : Clause 1: Bob shall deposit 100.00 euros to
11 #                    : Alice's bank account on the 25th of Apr 2021 as a
12 #                    : donation to support the creation of a painting of
13 #                    : a black.
14 #                    : Clause 2: Alice shall spend 50.00 on the purchase of
15 #                    : material and send by post the receipts to Bob within
16 #                    : 5 days of receiving the money.
17 #author              : Carlos Molina-Jimenez
18 #date                : 28 Dec 2020 (Computer Laboratory Univ of Cambridge)
19 #version             : 1
20 #notes               :
21 #python_version      : Python 3.7.4
22 #
23 #compile and run     : python scpayreceiptoperations.py
24 #                    :
25 #result              : The program is meant to be launched on
26 #                      the 25th Apr 2021. After launch it checks
27 #                      the current date and a pay date. If it is
28 #                      pay date, it opens Bob's (the payer)
29 #                      bank account, withdraws 100.00 euros and
30 #                      deposits them to Alice's (the payee)
31 #                      bank account. Next it prints that the
32 #                      receipts have sent sent to Bob.
```

```
 #
 # online run     : I have run it successfully on

#                  https://repl.it/languages/python3
#======================================================


 from datetime import date
 from datetime import datetime
 text_hash_and_add= ’’QmSNZBkCaRsAnDrAX3JgYsHm4n2aZmoRbSo’’

 aliceAcc= "A123"      # Alice’s bank account
 bobAcc= "B789"        # Bob’s bank account
 amount= 100           # amount to transfer from Alice to Bob.
 paydate= "28/12/2020" # pay date
 paydone= "no"         # pay not performed yet
 getDate= date.today()

 today= getDate.strftime("%d/%m/%Y")
 print("\n\ntoday is:", today)

 if today == paydate:
    print("\n Bob, this code will pay Alice today, right now!\n")
    #
    # Actual code to transfer money from Bob’s account
    # to Alice’s account to be included here.
    #
    print(str(100) + " euros have been transferred from Bob’s to Alice’s account\n")
    paydone= "yes"
 else:
    now = datetime.now()
    deadline= datetime.strptime(paydate, "%d/%m/%Y")
    if now < deadline :
       print("\n Bob, you have a pending deadline to pay on " + paydate + "\n")
    elif now > deadline:

      print("\n Bob, you have missed the pay deadline on " + paydate + "\n")


  /***** clause 2: list 5  *///
 if paydone== "yes":
    #
    # Actual code to contact Alice (for ex by email) to get
    # the expenses receipts and forward them to Bob (for example
```


code of list 4 and list 5
Alice
interaction
Code that implements clause 1
Code that implements clause 2
signedTextCont.jpeg
interaction
Bob
Computer connected to the Internet

```
# : scoperation.py
...

from datetime import date
from datetime import datetime
 text_hash_and_add=
“QmSNZBkCaRsAnDrAX3JgYsHm4n2aZmoRbSo“
```

```

if paydone == “yes”
...
#
# Actual code to
```

Figure 21. Execution of a contract written on paper and code (see Fig. 20.

```
    # by email. The code waits for seven days for Alice’s reply. If
    # Alice fails to reply the code sends a notification to Alice to
    # let her know about the missing deadline. This code assumes that
     # everything goes smoothly.
    #
    print("Alice, I’ve forward the receipts to Bob, well done!\n")
```

Fig. 21 shows the execution of the code of List 3. Notice that the code of clause 1 is bound to the file signedTextContr.jpeg that Peter produced after scanning the document shown in Fig. 20

5.4. *Hybrid contracts*

A hybrid contract is a contract that is executed and enforced by computer executable code in collaboration with humans. Namely, it exists part in computer code and in linguistic text, and its execution occurs partly in computer automation and in human intervention. Precisely, we define a hybrid contract as a contract that meets two conditions:

- The contract is recorded in code that can be retrieved by a reader program from a file, interpreted and displayed on a computer screen as text that is readable to humans.
- The contract is signed under cryptographic signatures.

Normally, a hybrid contract includes one or more contractual commitments expressed as executable code that can be executed to fully or partially automate their execution and enforcement. By fully and partially automatic we mean that the executable code can be programmed to either execute to completion without any human intervention or to stop during execution to request the intervention of humans when human judgement is needed. Central to hybrid contracts is the cryptographic bound between the executable code and the document that contains the human readable text. We elaborate these points with the help of the example shown in Fig. 22.

Hazem: Cant we have a contract that's fully computer code but have human intervention?

A salient limitation of the execution of the natural language contracts shown in Fig. 15 is the physical and logical separation between the contract and the code that implements clause 1 (see execution in Fig. 17) The connection between the contract in natural language shown in Fig. 15 and the code of List 1 is done manually is the sense that Alice and Bob are not provided with automatic mechanisms to relate the code of List 1 to the text in Fig. 15

For example, they do not have mechanisms to navigate (for example, by means of mouse clicks) from the text of Fig. 15 to the executable code in execution shown in Fig. 17 or in reverse, from the executable code in Fig. 17 to the text shown in Fig. 15. This drawback is exacerbated in the contract recorded solely in code like the example of Fig. 18. Notice that the executions of Clause 1 and Clause 2 shown in Fig. 19 is not backed up by a legal contract, there is no text that Alice, Bob or somebody else (for example, a judge) can refer to, to asses the execution that takes place in the figure. These limitations can be addressed with a help of hybrid contracts of which show an example in Fig. 22.

Let us assumes that Alice and Bob hire Peter (a professional programmer) and delegate to him the responsibility of producing a corresponding hybrid contract from the text shown in Fig. 22. The programmer produces List 4. which is the text shown in Fig. 22 but augmented with tags and the public keys of the participants, Alice's and Bob's respectively. To simplify the discussion, in this example we assume that Peter uses JSON notation, yet other tag languages can be used, including XML.

Listing 4 ...

```

{
''contract_id'': ''SMF567890'',
''start date'': ''10-Jan-2021'',
''expiry_date'': ''30-Dec-2021'',

''artist'':
 {
  ''full_name'':            ''Alice Wonder'',
  ''home_address'':   ''2 Art Street'',
 ''bank account'':      ''A123'',
  ''email'':                    ''alice@wonderland.com''
 },

''benefactor'':
{
 ''full_name'':            ''Bob Good'',
  ''home_address'':   ''5 Rich Street'',
 ''bank account'':      ''B789'',
  ''email'':                    ''Bob@artlovers.com''
},
```

This is ...

This is an agreement between Alice Wonder (the Artist) and Bob Good (the Benefactor).

Signature place:
Florence

Signature date:
29 Dec 2020

Alice's personal data:
Full name: Alice Wonder
Home address: 2 Art street
Bank account: A123
Email: alice@newartists.com

Bob's personal data:
Full name: Bob Good
Home address: 5 Rich Street
Bank account: B789
Email: bob@artlovers.com

Clause 1: Bob shall deposit 100 euros to Alice's bank account as a donation to support the creation of a painting of a black cat.
Clause 2: Alice shall spend 50 on the purchase of material and send by post the receipts to Bob within 5 days of receiving the money.

Figure 22. Agreement to be translated into a hybrid contract.

```

''clauses'':
[
  [''text'' : '' Clause 1: Bob shall deposit 100 euros to Alice's
                     bank account as a donation to support the
                     creation of a painting of a black cat.'',
  ''implementation'': ''automatic'',
   ''code_hash_and_addr'': ''QmSNZBDkCaX3JgYsHm4n2aZmolVUoR''],
  [
   ''text'' :  ''Clause 2'': Alice shall spend 50 on the purchase of
                    material and send by post the receipts to Bob
                    within 5 days of receiving the money.'',
   ''implementation'': ''manual'',
   ''code_location'': ''Nil'']
 ],

''artist_public_key'':           ''AAAAB3NzaC1yc2EAAAA...T0Haf8b'',
''benefactor_public_key'': ''BBAAB3NzaC1yc2EAAAA...6dR/m7 '',

}
```

As shown in List 4 we assume that Alice and Bob agreed that clause 1 is to be executed automatically and clause 2 manually. Accordingly, Alice and Bob delegate the responsibility of translating clause 1 to executable code. The process is the same as the conversion of clause 1 of contract in natural language shown in Fig. 16 except that as Peter produces a digitally signed contract.

We make the same assumption regarding the correctness of the conversion, namely, Alice and Bob need to verify that the code produced by Peter correspond to what is stipulated in the text of clause 1. Presumably, they will test Peter's code thoroughly before proceeding to sing the contract under their cryptographic signatures. Let us assume that the verification of the code succeeds and that Alice and Bob cryptographically sign hybrid contract. The result is shown in List 5. Alice's and Bob's signatures can be seen in the last lines of the text.

Listing 5 Code signed by Alice and Bob to produce a hybrid contract.

```
{
''contract_id'': ''SMF567890'',
''start date'': ''10-Jan-2021 '',
''expiry_date'': ''30-Dec-2021'',

''artist'':
 {
  ''full_name'':             ''Alice Wonder'',
  ''home_address'':    ''2 Art Street'',
 ''bank account'':       ''A123'',
  ''email'':                   ''alice@wonderland.com''
 },

''benefactor'':
{
 ''full_name'':              ''Bob Good'',
  ''home_address'':    ''5 Rich Street'',
 ''bank account'':       ''B789'',
  ''email'':                   ''Bob@artlovers.com''
},

''clauses'':
[
  [''text'' : '' Clause 1: Bob shall deposit 100 euros to Alice's
                     bank account as a donation to support the
                     creation of a painting of a black cat.'',
  ''implementation'': ''automatic'',
   ''code_hash_and_addr'': ''QmSNZBDkCa3...Hm4n2aZmolVUoR''],
  [
   ''text'' :  ''Clause 2'': Alice shall spend 50 on the purchase of
                     material and send by post the receipts to Bob
                      within 5 days of receiving the money.'',
   ''implementation'': ''manual'',
   ''code_hash_and_addr'': ''Nil'']
 ],

''artist_public_key'':            ''AAAAB3NzaC1yc2EAAAA...T0Haf8b'',
''benefactor_public_key'': ''BBAAB3NzaC1yc2EAAAA...6dR/m7 '',
}

-----BEGIN artist's SIGNATURE-----
iQKBBAEBAgBrBQJThXbMZBxEciBXYXNoaW5ndG9uIFlhbWFuZHU
gU2FuY2hleiAoMjAxNCBLZX30dic2020tBcSaMiFewYWlyIqgZiwWR
cmVzcykgPHdhc2hpbmd0bHnf8cltotoFuFY2hlekBvdXSMFxRvsCMJ
...   ...   ...
ZX9YNFFPguwsoztYyWQGfhQItzUxfnsBFn4JvU6LyHI/orbzmtUSR4
FHKPpTRmIBYDcVMMAv/u1eXOL8zd43kLcD0qp551FRjHVPhuOcu
R7tPkhdKirlZ4anbaEBMA fdUl5j/whtLQ7afe3zsQ+9R4rpA= =zW6Y
```

```
                          -----END artist's SIGNATURE-----

                           -----BEGIN benefactor's SIGNATURE-----
3zbjE+WjXjOGHjuAd/u5fdezTzG26BwSb0jahgvLwr+DvaOH5
HX3jDtpQH5Vi8p1ULW9dVeAX7ZHC//as0IY+6uuhLBy7+HlIz
mn2ka63ML7Kqk+w54jdjggja0mJ3PXmjvbEyGYDpDtks0jps3nUvB
...    ...    ...
pRp+85ArCMJHEoUyp11D1c2018mWNf6FDKclbeBAKMJSaMiFeuv
                      -----END benefactor's SIGNATURE-----
```

One of the strengths of hybrid contracts is shown in keys of the clauses, namely ”implementation” and `code_hash_and_addr`. The ”implementation” key in clause 1 indicates that clause 1 is to be executed automatically, in this example, with the assistance of the code produced by Peter out of the text of clause 1. The value (QmSNZBDkCa3...Hm4n2aZmolVUoR) of the key `code_hash_and_addr` is the hash of the code produced by Peter. It is used for three purposes: it is a digital finger print that guarantees the integrity of the code—it cannot be altered without detection; the hash is also used as a unique identifier of the code and also, as the address to store the code in a file system. If Alice and Bob were to store the code in the IPFS file system, it would be under the QmSNZBDkCa3...Hm4n2aZmolVUoR address. As shown in Fig. 24, the hash is used to bind the code to the hybrid contract.

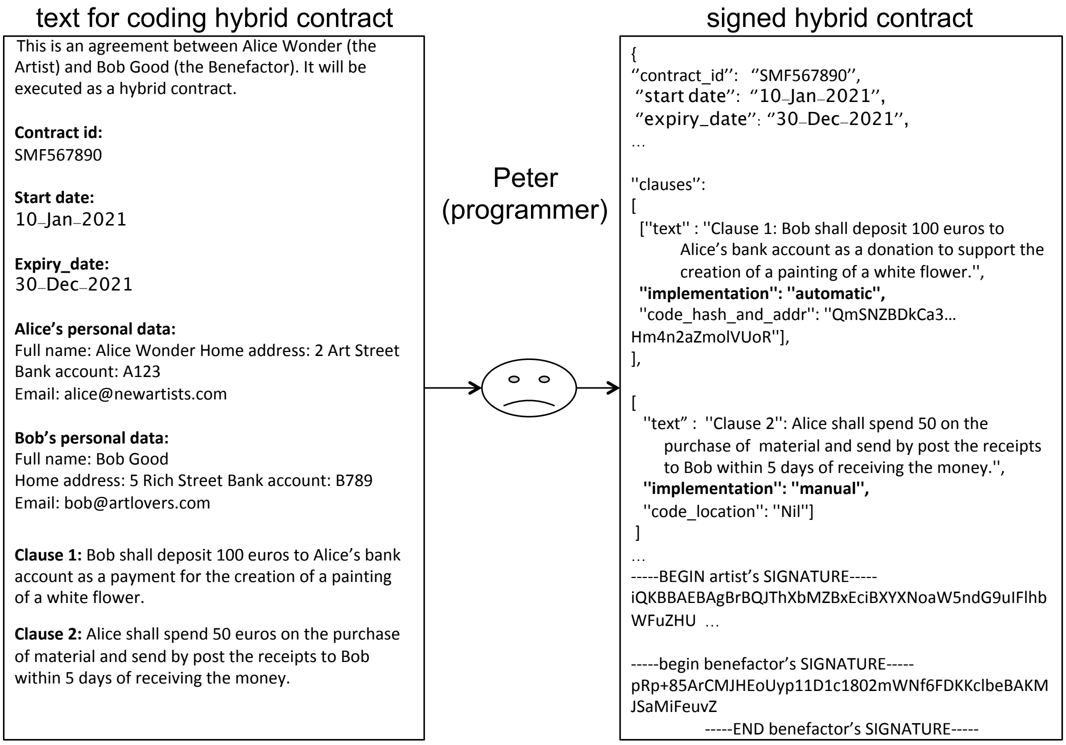
The ”implementation” key shows that clause 2 is to be executed manually, accordingly, the value of `code_hash_and_addr` is ”NIL”. Fig. 23 shows a summary of the procedure to produce the signed hybrid contract with Peter's assistance.

A salient feature of a hybrid contract is that it can be read by both humans and computer programs:

- Readable by humans: it means that a human like Alice and Bob can look at the text bear-eyed (see List 4) and understand what the document stipulates and on this basis sign it cryptographically to produce the document shown in List 5. Likewise, if List 5 is presented to a judge (for example, to solve a dispute) she or he will be able to examine the content of the document which is in fact a signed legal contract.
- Readable by computer program: it means that a computer program is able to parse and interpret the tags included in the text. This facility can be used for building several mechanisms for interacting with the hybrid contract.

  Hazem: Should we call it hybrid or should we call it machine readable?

  For example, to save herself from the annoyance of looking at the tags included in the text, Alice can use a reader program (also called a viewer) that is capable of interpreting the tags and presenting a clean plain text on the computer screen. As an analogy, we can mention that readers are included in conventional browsers (for example Internet explorer and Fire Fox) to display web pages which are also tagged documents. A good reader will be able to display the whole document on the screen or only the component requested by Alice or Bob, such as the full name of the artist, the home address of the benefactor or the clauses of the contract. A more sophisticated reader will include functionality to help Alice and Bob to interact with executable code of the contract (with CClause 1 in this example); for example, to enable Alice to request the current status of the execution, such as what clauses are pending, what have been fulfilled or violated and so on. The



This is an agreement between Alice Wonder (the Artist) and Bob Good (the Benefactor). It will be executed as a hybrid contract.

Contract id:
SMF567890

Start date:
10_Jan_2021

Expiry_date:
30_Dec_2021

Alice's personal data:
Full name: Alice Wonder Home address: 2 Art Street
Bank account: A123
Email: alice@newartists.com

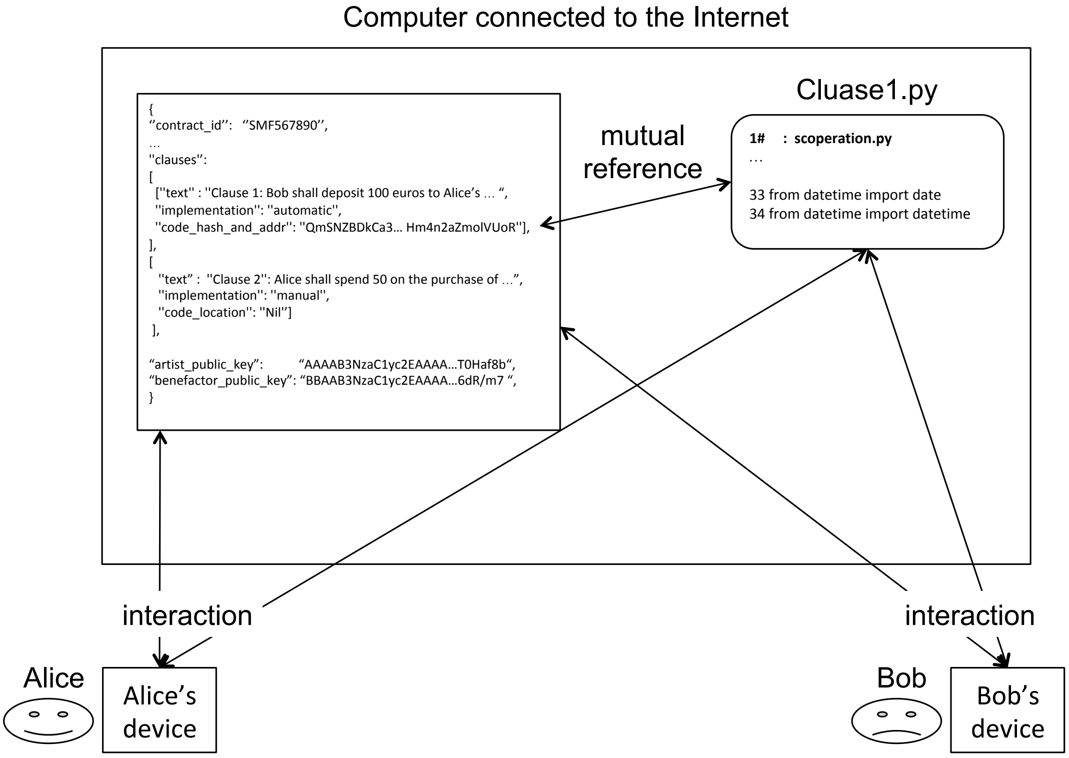
Bob's personal data:
Full name: Bob Good
Home address: 5 Rich Street Bank account: B789
Email: bob@artlovers.com

Clause 1: Bob shall deposit 100 euros to Alice's bank account as a payment for the creation of a painting of a white flower.

Clause 2: Alice shall spend 50 euros on the purchase of material and send by post the receipts to Bob within 5 days of receiving the money.

```
{
"contract_id":   "SMF567890",
 "start date":  "10_Jan_2021",
 "expiry_date": "30_Dec_2021",
...

"clauses":
[
 ["text" : "Clause 1: Bob shall deposit 100 euros to
          Alice's bank account as a donation to support the
          creation of a painting of a white flower.",
 "implementation": "automatic",
 "code_hash_and_addr": "QmSNZBDkCa3…
Hm4n2aZmolVUoR"],
],

[
  "text" :  "Clause 2": Alice shall spend 50 on the
      purchase of  material and send by post the receipts
      to Bob within 5 days of receiving the money.",
  "implementation": "manual",
  "code_location": "Nil"]
 ]
...
-----BEGIN artist's SIGNATURE-----
iQKBBAEBAgBrBQJThXbMZBxEciBXYXNoaW5ndG9uIFlhb
WFuZHU  ...

-----begin benefactor's SIGNATURE-----
pRp+85ArCMJHEoUyp11D1c1802mWNf6FDKKclbeBAKM
JSaMiFeuvZ
            -----END benefactor's SIGNATURE-----
```

Figure 23. Coding of a hybrid contract.



```
{
"contract_id":   "SMF567890",
...
"clauses":
[
 ["text" : "Clause 1: Bob shall deposit 100 euros to Alice's ... ",
 "implementation": "automatic",
 "code_hash_and_addr": "QmSNZBDkCa3… Hm4n2aZmolVUoR"],
],
[
  "text" :  "Clause 2": Alice shall spend 50 on the purchase of ...",
  "implementation": "manual",
  "code_location": "Nil"]
 ],

"artist_public_key":         "AAAAB3NzaC1yc2EAAAA…T0Haf8b",
"benefactor_public_key": "BBAAB3NzaC1yc2EAAAA…6dR/m7 ",
}
```

```
1#    : scoperation.py
...

33 from datetime import date
34 from datetime import datetime
```

Figure 24. Execution of the hybrid contract

reader can also be instrumented to verify that the executable code of CCaluse 1 has not be tampered with. A reader used for interacting with a more complex contract (see the multimedia contract in List 6) that includes several components, can be instrumented to locate and retrieve them.

Notice that in Fig. 24, the hybrid contract and the executable code that implements clause 1 refer to each other cryptographically. In the figure, they are collocated in the same computer, however, there are no technical difficulties in deploying them in different computers remotely connected through the Internet. Also, it is worth noticing that clause 2 can also be treated as clause 1: it can be converted to code and executed automatically and not necessarily in the same computer as the code of clause 1; the location of the codes that implement the clauses is irrelevant as long as they can be reached. In general, an agreement can be converted to a hybrid contract fully (all the clauses) or partially (only some of the clauses); all depending on the nature of the clauses; some of then might bear accidental or intentional ambiguity that makes then not amenable to computer codification; others clauses might be coded but the participants might decide to execute then manually, say, for efficiency reasons or to keep the flexibility that human judgment provides; other clauses might be implemented semi-automatically, that is, implemented in manner that in some situation they will execute fully to completion without requiring human intervention but in other situations their execution will stop and send an signal alarm (for example, an email message) to the participants to request human intervention to proceed to the next step of the execution.

5.5. *Multimedia contracts*

It is worth observing that the code of CClause1.py in Fig. 24 can be regarded as an object that contributes to the composition of the hybrid contract. In this example, it is a piece of executable code, but in general, it can be any piece of digital information, including, plain text, figures, videos, music and so on. This generalization would convert the hybrid contract shown in the figure to a multimedia contract.

Intuitively, we define a multimedia contract as a contract that is composed of several components (objects or pieces) of different formats such as text, executable code, images, sound, video, etc.

Precisely, we define a multimedia contract as a contract that meets the following conditions:

- The contract has a main component that is recorded in code that can be retrieved by a reader program from a file, interpreted and displayed on a computer screen as text that is readable to humans.
- In addition to the main component, the contract includes one or more multimedia components arbitrarily located in the web but referred to each other by mean of cryptographic links.
- The contract is signed under cryptographic signatures.

The execution of a multimedia contract is similar to the execution of a hybrid contract in the sense that it accounts for executable code that enforces contractual commitments automatically or semi–automatically that require human intervention and for clauses that are executed only in the traditional manner. We will use List **??** to elaborate our arguments.

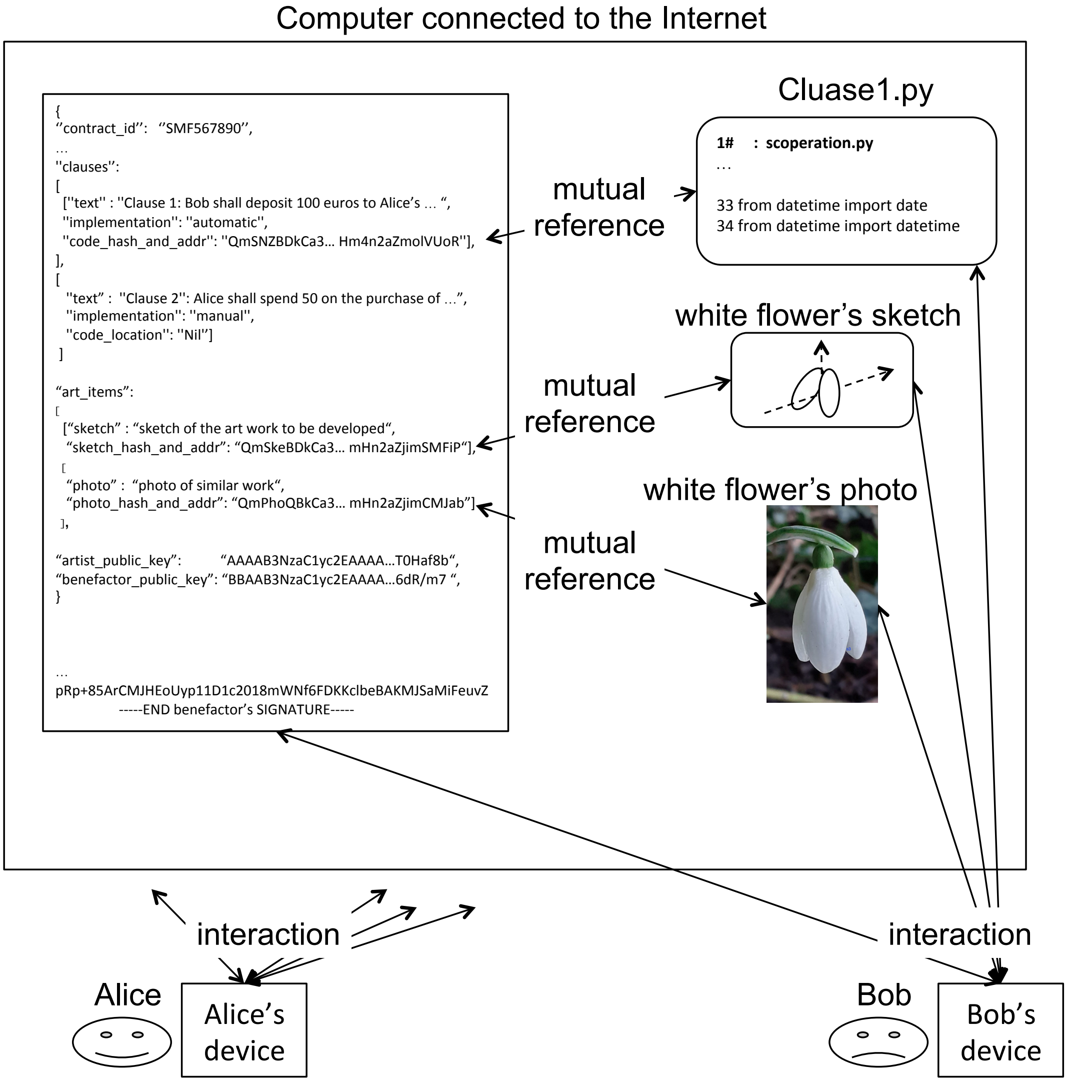
The advantages of multimedia contracts are numerous. We believe that they can significantly contribute to the automation of law: a law where most of the rules are executed automatically by multimedia contracts that refer to multimedia objects including other contracts.

List 6 is an example of a multimedia contract that includes, in addition to the contractual clauses of the previous example, two additional components: a sketch of the piece of art that the artist is committed to paint and a photo of similar work by the artist.

Listing 6 Multimedia contracts signed by Alice and Bob.

```

 {
''contract_id'': ''SMF567890'',
''start date'': ''10-Jan-2021 '',
''expiry_date'': ''30-Dec-2021'',

''artist'':
 {
  ''full_name'':             ''Alice Wonder'',
  ''home_address'':    ''2 Art Stree'',
 ''bank account'':       ''A123'',
  ''email'':                      ''alice@wonderland.com''
 },

''benefactor'':
{
 ''full_name'':              ''Bob Good'',
  ''home_address'':    ''5 Rich Street'',
 ''bank account'':       ''B789'',
  ''email'':                      ''Bob@artlovers.com''
},

''clauses'':
[
  [''text'' : '' Clause 1: Bob shall deposit 100 euros to
     Alice's
                     bank account as a donation to support
                        the
                     creation of a painting of a black cat.
                        '',
  ''implementation'': ''automatic'',
   ''code_hash_and_addr'': ''QmSNZBDkCa3...Hm4n2aZmolVUoR'']
     ,
  [
   ''text'' :  ''Clause 2'': Alice shall spend 50 on the
      purchase of
                     material and send by post the receipts
                        to Bob
                      within 5 days of receiving the money.
                          '',
   ''implementation'': ''manual'',
   ''code_hash_and_addr'': ''Nil'']
 ],


''art_items'':
[
  [''sketch'' : ''sketch of the art work to be developed'',
   ''sketch_hash_and_addr'': ''QmSkeBDkCa3...mHn2aZjimSMFiP''],
  [
   ''photo'' :  ''photo of similar work'',
   ''photo_hash_and_addr'': ''QmPhoQBkCa3...mHn2aZjimCMJab'']
 ],

''artist_public_key'':            ''AAAAB3NzaC1yc2EAAA...T0Haf8b'',
''benefactor_public_key'': ''BBAAB3NzaC1yc2EAAAA...6dR/m7 '',
}


                   -----BEGIN artist's SIGNATURE-----
iQKBBAEBAgBrBQJThXbMZBxEciBXYXNoaW5ndG9uIFlhbWFuZHU
gU2FuY2hleiAoMjAxNCBLZX30dic2020tBcSaMiFewYWlyIqgZiwWR
cmVzcykgPHdhc2hpbmd0bHnf8cltotoFuFY2hlekBvdXSMFxRvsCMJ
...   ...   ...
ZX9YNFFPguwsoztYyWQGfhQItzUxfnsBFn4JvU6LyHI/orbzmtUSR4
FHKPpTRmIBYDcVMMAv/u1eXOL8zd43kLcD0qp551FRjHVPhuOcu
R7tPkhdKirlZ4anbaEBMA fdUl5j/whtLQ7afe3zsQ+9R4rpA= =zW6Y
                         -----END artist's SIGNATURE-----


                       -----BEGIN benefactor's SIGNATURE-----
3zbjE+WjXjOGHjuAd/u5fdezTzG26BwSb0jahgvLwr+DvaOH5
HX3jDtpQH5Vi8p1ULW9dVeAX7ZHC//as0IY+6uuhLBy7+HlIz
mn2ka63ML7Kqk+w54jdjggja0mJ3PXmjvbEyGYDpDtks0jps3nUvB
...   ...   ...
pRp+85ArCMJHEoUyp11D1c2018mWNf6FDKclbeBAKMJSaMiFeuv
                     -----END benefactor's SIGNATURE-----
```

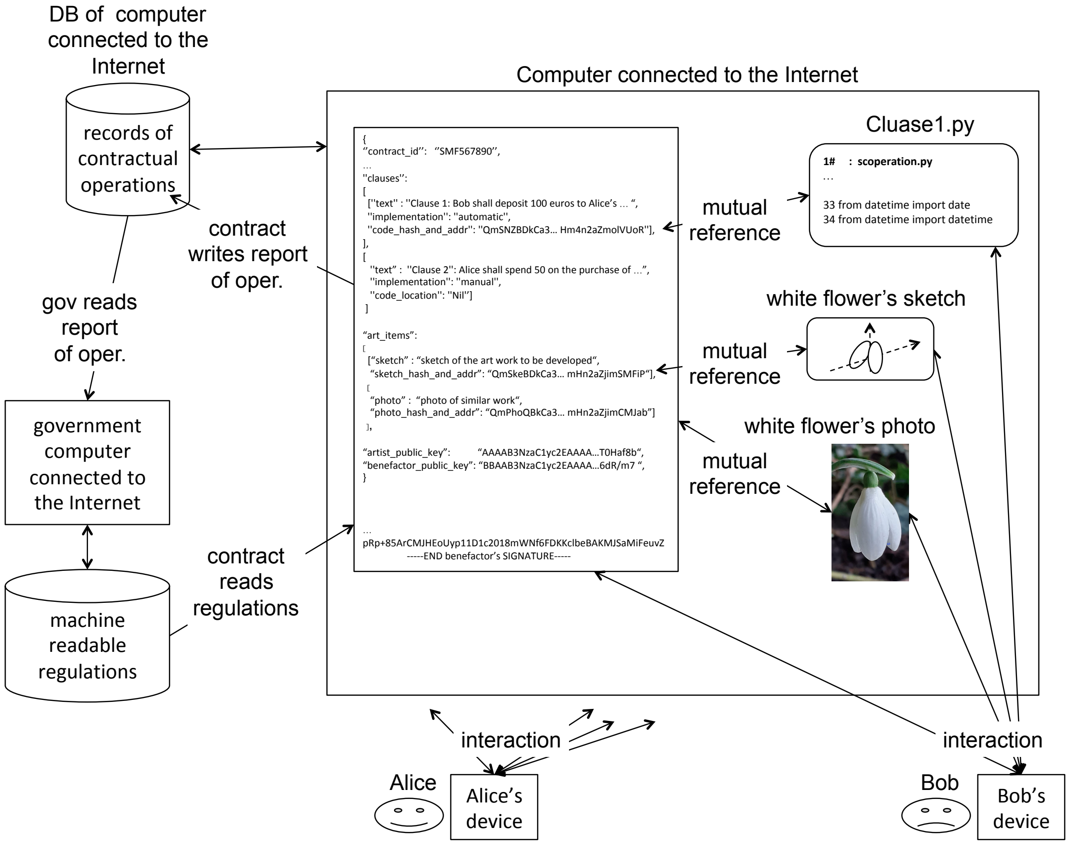
Fig. 25 shows the execution of the multimedia contract shown in List 6. Notice that this multimedia contract is related to three components (objects): the code of clause 1 and two art items, namely, a sketch of the art work that the artist is committed to produce and a photo of similar work by the artist. The multimedia contract refers to the pieces of art through their hashes. The hashes are used to identify art pieces and as addresses to locate them in the file systems where they are stored. In this manner QmSkeBDkCa3...mHn2aZjimSMFiP and QmPhoQBkCa3...mHn2aZjimCMJab uniquely identify the sketch and the photo, respectively. We emphasise that the multimedia contract and all its components are collocated in



```
{
''contract_id'':   ''SMF567890'',
…
''clauses'':
[
 [''text'' : ''Clause 1: Bob shall deposit 100 euros to Alice's … ",
 ''implementation'': ''automatic'',
 ''code_hash_and_addr'': ''QmSNZBDkCa3… Hm4n2aZmolVUoR''],
],
[
  ''text" :  ''Clause 2'': Alice shall spend 50 on the purchase of …",
  ''implementation'': ''manual'',
  ''code_location'': ''Nil'']
 ]

"art_items":
[
 ["sketch" : "sketch of the art work to be developed",
  "sketch_hash_and_addr": "QmSkeBDkCa3… mHn2aZjimSMFiP"],
 [
  "photo" :  "photo of similar work",
  "photo_hash_and_addr": "QmPhoQBkCa3… mHn2aZjimCMJab"]
 ],

"artist_public_key":         "AAAAB3NzaC1yc2EAAAA…T0Haf8b",
"benefactor_public_key": "BBAAB3NzaC1yc2EAAAA…6dR/m7 ",
}



…
pRp+85ArCMJHEoUyp11D1c2018mWNf6FDKKclbeBAKMJSaMiFeuvZ
            -----END benefactor's SIGNATURE-----
```

```
1#    :  scoperation.py
…

from datetime import date
from datetime import datetime
```

Figure 25. Execution of a multimedia contract.


DB of computer connected to the Internet
records of contractual operations
contract writes report of oper.
gov reads report of oper.
government computer connected to the Internet
machine readable regulations
contract reads regulations
Computer connected to the Internet
Cluase1.py
mutual reference
white flower's sketch
mutual reference
white flower's photo
mutual reference
interaction
interaction
Alice
Alice's device
Bob
Bob's device

```
{
''contract_id'':   ''SMF567890'',
…
''clauses'':
[
 [''text'' : ''Clause 1: Bob shall deposit 100 euros to Alice's … ",
 ''implementation'': ''automatic'',
 ''code_hash_and_addr'': ''QmSNZBDkCa3… Hm4n2aZmolVUoR''],
],
[
  ''text" :  ''Clause 2'': Alice shall spend 50 on the purchase of …",
  ''implementation'': ''manual'',
  ''code_location'': ''Nil"]
 ]

"art_items":
[
 ["sketch" : "sketch of the art work to be developed",
  "sketch_hash_and_addr": "QmSkeBDkCa3… mHn2aZjimSMFiP"],
 [
  "photo" :  "photo of similar work",
  "photo_hash_and_addr": "QmPhoQBkCa3… mHn2aZjimCMJab"]
 ],

"artist_public_key":         "AAAAB3NzaC1yc2EAAAA…T0Haf8b",
"benefactor_public_key": "BBAAB3NzaC1yc2EAAAA…6dR/m7 ",
}



…
pRp+85ArCMJHEoUyp11D1c2018mWNf6FDKKclbeBAKMJSaMiFeuvZ
            -----END benefactor's SIGNATURE-----
```

```
1#    : scoperation.py
…

from datetime import date
from datetime import datetime
```

Figure 26. Execution of the multimedia contract with machine readable legislations.

the same computer for convenience of presentation, in practice, there are more likely to be located in different computer. For simplicity, the multimedia contract and its components are shown in isolation, in practice they are more likely to be part of arbitrarily graph of other similar contracts and components, just like conventional Internet web pages.

Machine readable legislations is discussed in [26, 35].

6. Further contract categorisation

In this section resume the topic of contract categorisation and discuss three additional categories.

6.1. *Centralised and decentralized deployment of contracts*

From the perspective of the infrastructure used for deployment, we classified digital contracts into. See Fig.27.

- Centralized digital contracts.
- Decentralized digital contracts.

We say that a digital contract is centralized if only a single replica (copy) is executed (instantiated or run). In contrast, we say that a digital contract is decentralized if

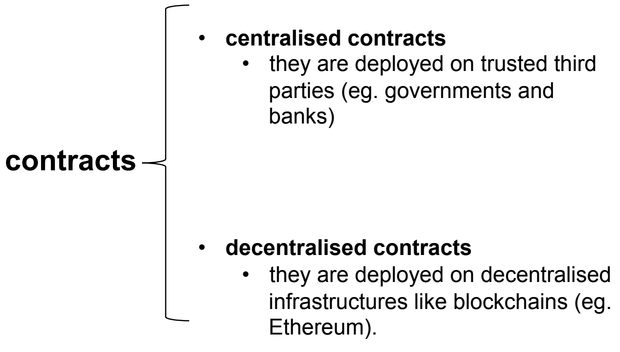

contracts
centralised contracts
they are deployed on trusted third parties (eg. governments and banks)
decentralised contracts
they are deployed on decentralised infrastructures like blockchains (eg. Ethereum).

Figure 27. Contract categories from the perspective of decentralisation.

more or two replicas are executed. We will use Fig.?? to explain the main issues involved in one or the other alternative. Notice that in the figure, we use the term smart contract (SC) rather than "digital contract" to match the term used by the blockchain community.

As shown the figure, the unique replica of a centralised contract is deployed on a computer (PC) run by a Trusted Third Party (TTP) located in the Internet, for example, it can be a cloud provider. From the technical perspective, the physical location of the computer is irrelevant, as long as it remain reachable to Alice and Bob. The figure shows a decentralized digital contract that has been replicated four times. Replica SC_1 runs on PC_1, replica SC_2 runs in PC_2 and so on. PC_1, PC_2, PC_3 and PC_4 are computer connected to the Internet and members (node) of a blockchain. Examples of blockchains are Bitcoin, Ethereum and Hyperledger. The sync protocol that the four replicas execute is a consensus algorithm that they need to execute together to synchronise their states. For example, if SC_2 has enforced and recorded the execution of a payment operation, it has to propagate the information to SC_1, SC_3 and SC_4 to ensure that all the replicas record the fulfillment of the pay obligation.

As explained at large in Chapter Decentralization, the blockchain infrastructure that hosts the digital contract offers several advantages and opens the door to a large class of contract applications. Unfortunately, it raises several legal issues that are currently poorly understood.

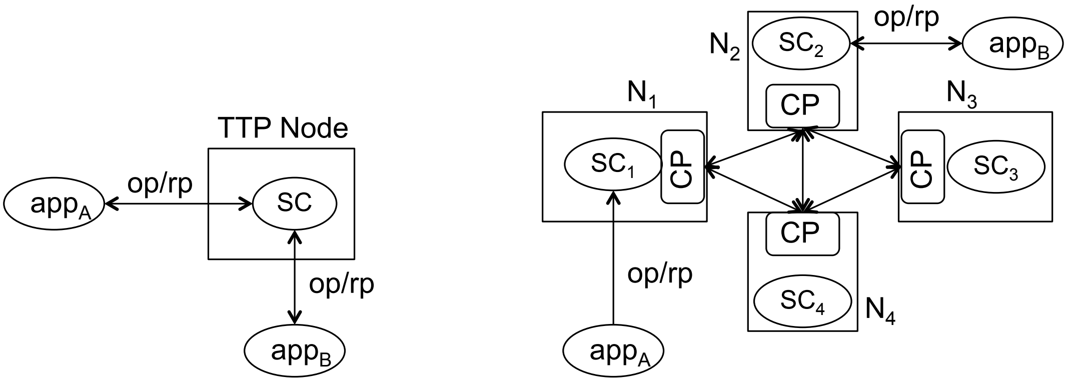
6.2. *Bilateral and multilateral contracts*

From the perspective of the number of signatories, contracts can be classified into bilateral which involved two parties only and multilateral which are signed by more than three parties. Since bilateral contracts are more frequently used and easier to understand, in our work will focus only on them. Multilateral contracts are also used in some applications such as multinational agreements non–surprisingly, they have also been studied by academics but are still immature to reach practical applications.

7. Future work and discussion

7.1. *Why do we call them digital contracts?*

It can be argued that the term digital contract ...



a) centralised deployment.

b) decentralised deployment.

legend:
TTP: Trusted Third Party.
app_A and app_B: Alice's and Bob's application, respectively.
SC: smart contract (a digital contract in blockchain terminology).
op: operation executed by Bob or Alice, ex, send-pay, send-receipt.
rp: response to operation, ex., execution OK, execution pending, etc.
N: node (computer connected to the Internet, ex, Peter's PC).
CP: consensus protocol to synchronise the state of the N replicas of the SC.
N: a number larger than 1, ex, 2,3,4, 5, 20, 500.

Figure 28. Centralised and decentralised deployment of digital contracts.

We prefer it instead of the term smart contract because we do not find the latter intuitive. In fact, we find the word ”smart” misleading because we are only talking about computer executable code with no notion of intelligence. Some authors use the term electronic contract; our reservations about it is that in computer science and related fields, the word ”electronic” is used to refer to the actual hardware technology (for example, circuits and cables) that underpins the execution of contracts and other software applications. We regard a contract as software concept that is independent of the electronics of the computer; therefore, we prefer not to associate the term electronic with them.

We agree that ”executable contract”, ”automatic contract”, ”programmable contract” and ”computational contracts” are more intuitive than ”digital contract”. We feel that the last two terms especially capture the idea more accurately because we are talking about contracts that can be programmed in a programming language to automate the execution of contractual commitments. Having said that, we still decided to embrace the term digital contract to support the term that is currently the widely used.

One of the most ambitious and promising usages of digital contracts is in law automation. This is a recently emerging concept referred to by different terms. In [25] is is called Lex Cryptographia, in [36, 37, 38] it is called Computational Law, in [26] it is called Replacement of Law with Computer Code, in [39] its is called Computable Contracts. It can fairly be called Programmable Law too. The general idea behind all these concept is to take advantage of recent advances in computer technology to automate the enforcement of regulations in different fields of our society ranging from regulation within private companies to governments. ... speculate about AI, IoT, etc.

8. Related work

The overall goal of our work is to help in the understanding the relationship between legal contracts as conceived traditionally by lawyers and digital contracts as conceived by software engineers. Early attempts to clarify this question have been made before by academics [19, 40, 41, 42, 26, 39, 30] governments [43] and industry. A common feature of existing work is that they focus only on smart contracts, that is, on decentralised digital contracts deployed blockchain execution environments like Ethereum. In our view, the inclusion of the blockchain from the very beginning of the discussion adds unnecessary complexity to the understanding of digital contracts. A salient feature of our work is that we take a different approach to avoid this unnecessary complexity. We discuss digital contracts independently from blockchains. This approach allows us to focus on the issues that are inherent to digital contracts. As explained in [44], digital contract can be deployed on blockchains which is suitable in several applications but they they can be also deployed on other execution environments such as trusted third parties.

Our discussion of the categorisation of automatic contracts was inspired by and motivated by the report produced by the Law Commission [43] (summary is available [45]). They distinguish between three categories (forms in their terminology) of contracts. Our work extends theirs in several aspects. Firstly, We distinguish between five categories of automatic contracts. Secondly, we provide a precise definition of each category supported with examples. Equally, importantly, to help readers from the law and computer science communities, we make and effort to clarify and relate the different terminologies used.

The issues about the disagreement between terminologies used by the law and computer science communities is raised in [28]. The authors point out that the contract stages between contracts as understood by lawyers and the stages of smart contracts as conceived by the blockchain community clash; however, they do not show when and how. The issue is briefly mentioned in [19] and [24]. To progresses the state of knowledge in this direction, our work thoroughly analyses the stages included in the life cycle of legal and digital contracts and compares terminologies. Early attempts to show the life cycle of smart contracts have been made by some authors. Stages included in the life cycle of legal contracts have been widely studied. Most scholars agree that the life cycle of a legal contract in the Common Law includes four stages: offer, acceptance consideration, performance and termination [21]. Though the terminology and stages might vary slightly, it is fair to say that the life cycle of legal contracts is well understood. Incidentally, in [30] the author considers formation, performance, and breach. Regarding the life cycle of digital contracts most software engineers consider four stages: creation, deployment, execution and completion. See for example [32].

Attempt to understand the legal aspects involves in the life cycle of digital contracts can be traced back to the 1990s. For example a pioneering discussion can be found in [46] where the authors raised the question about contract validity within the context of digital contracts. They observe that a contract is not legally valid unless it meets legal requirements: agreement, consideration competence (capacity) and legal purpose. The discussion is continued in [47] where the authors bring the stages of legal contracts into the discussion. They consider contract preparation (or information phase), the contract negotiation (or agreement) and the contract fulfillment. Efforts to clarify the question have been made recently by the legal community. See for example, in [45] and [41]. We feel that the flowchart approach that we take in our work explores the issue at a deeper level to help lawyers and software engineers to visualise the differences and

Property **Contract**	Form	Signature	Execution	Legal contracts
Manually signed	paper	holographic	manual	yes
On-line signed	computer text	electronic	manual	yes
Natural language	paper	holographic	...	yes
Solely in exec code	exec. comp. code	digital (Pub-sub)	automatic	no
Natural lang and exec code	...	...	...	...
Hybrids	tagged comp. text and executable code.	digital (Pub–Sub)	semiautomatic	yes
Multimedia	tagged comp. text with URL links to multimedia docs.	digital (Pub–Sub)	semiautomatic (?)	yes

Table 2. Cont.

similarities between legal and digital contracts.

In spite of the some technical problems and open questions [18], we believe in the advantages (cost reduction, clarity, freedom from human bias, and so on) of automatic enforcement of contracts. The advantages have been discussed by several authors. See for example [24]. A general discussion of contract enforceability is presented in [48].

9. Conclusions

Table 2 summarises ...

Automatic contract benefits [24].

Some of the limitations of smart contracts are discussed in [42].

Table **??** presents a summary of the main features that distinguish existing

Contract type **Property**	Non–automatic	Automatic
Legal framework	**Legal regulations are well understood:** centuries of experience.	**Legal regulations are missing:** there is little experience in their regulations.
Remote formation	**No:** requires face to face interaction, at least to hand sign the contract.	**Yes:** the entire life cycle of the contract can be done remotely.
Technology	**No:** No technology.	**Yes:** Sophisticated digital technology needed.
Formation time	**Long:** Can take days or months.	**Short:** Can be done in minutes.
Trust	**In parties:** parties rely on each other's willingness and reputation to honour the contract.	**In computer code:** parties rely on computer code to enforce the contract.
Clarity of terms and conditions	**Low:** the terms and conditions expressed in legal language tend to be plagued with accidental or intentional ambiguities and omissions.	**High:** the terms and conditions expressed in natural language need to be clearly written to be amenable to programming and automation.
Enforcing	**Corrective:** based on post morten examination, compensation and penalties.	**Preventive:** prevents the occurrence of violations.
Cost of execution	**High:** all the obligations are fulfilled by operations executed manually by humans.	**Low:** the contract is executed at low cost because it is executed automatically by computers.
Efficiency of execution	**Low:** human execution is slow, conducted only within business working hours and prone to errors.	**High:** computers are highly efficient in the execution of repetitive tasks.
Flexibility to give allowances	**High:** human judgement can be used to give allowances in contract violations when it is judged reasonable to do so.	**Low:** terms and conditions (eg. deadlines) are strictly followed and applied.
Granularity of enforcement	**Flexible:** human judgement can accept partial fulfillment of obligation as valid.	**All or nothing:** the code operates in a binary fashion: an obligation is either fulfilled or not fulfilled.
Security risks of exposing sensitive data	**Low:** the contract is concealed in physical cabinet.	**High:** the code of the contract is executed on computers reached from the Internet and exposed to cyberattacks.
Amendments	**Easy:** it involves paper work only.	**Hard:** it involves paper work, re-programming of the code that implements the contracts and redeployment.

Table 3. Comparison of non–automatic and automatic contracts.

Contract type **Property**	Non–automatic	Automatic	Explanation
Legal framework	**Legal regulations are well understood:** centuries of experience.	**Legal regulations are missing:** there is little experience in their regulations.	The ...
Remote formation	**No:** requires face to face interaction, at least to hand sign the contract.	**Yes:** the entire life cycle of the contract can be done remotely.	The ...
Trust	**In parties:** parties rely on each other's willingness and reputation to honour the contract.	**In computer code:** parties rely on computer code to enforce the contract.	The ...
Clarity of terms and conditions	**Low:** the terms and conditions expressed in legal language tend to be plagued with accidental or intentional ambiguities and omissions.	**High:** the terms and conditions expressed in natural language need to be clearly written to be amenable to programming and automation.	The ...
Enforcing	**Corrective:** based on post morten examination, compensation and penalties.	**Preventive:** prevents the occurrence of violations.	The ...
Cost of execution	**High:** all the obligations are fulfilled by operations executed manually by humans.	**Low:** the contract is executed at low cost because it is executed automatically by computers.	The ...
Efficiency of execution	**Low:** human execution is slow, conducted only within business working hours and prone to errors.	**High:** computers are highly efficient in the execution of repetitive tasks.	The ...
Flexibility to give allowances	**High:** human judgement can be used to give allowances in contract violations when it is judged reasonable to do so.	**Low:** terms and conditions (eg. deadlines) are strictly followed and applied.	The ...
Granularity of enforcement	**Flexible:** human judgement can accept partial fulfillment of obligation as valid.	**All or nothing:** the code operates in a binary fashion: an obligation is either fulfilled or not fulfilled.	The ...
Security risks of exposing sensitive data	**Low:** the contract is concealed in physical cabinet.	**High:** the code of the contract is executed on computers reached from the Internet and exposed to cyberattacks.	The ...
Amendments	**Easy:** it involves paper work only.	**Hard:** it involves paper work, re-programming of the code that implements the contracts and redeployment.	The ...

Table 4. Comparison of non–automatic and automatic contracts (old).

Acknowledgment

Thanks to ABC for proof–reading and commenting on a draft of this article.

References

[1] Translated by Benjamin Jowett, *Laws by Plato*, Global Grey Ebooks Translated by Benjamin Jowett, 2018.

[2] A. Y. Ali, *The Meaning of The Holy Quran*, I.D.C.I Islamic Dawah Centre International, 2010.

[3] Emptoris Contract Management, "Introduction to emptoris contract management (10.1.0)." `https://www.ibm.com/docs/en/ecm/10.1.0?topic=guide-introduction-emptoris-contract-management`, 2021. Visited on 1 jul 2021.

[4] CMx Contract Management Software, "9 stages of contract lifecycle management." `https://www.contractexperience.com/resources/resources-main.html`, July 2021. Visited on 1 jul 2021.

[5] R. George, "8 tips to optimize service contract lifecycle management." `https://global.hitachi-solutions.com/blog/contract-management`, July 2021. Visited on 1 jul 2021.

[6] N. H. Minsky and A. D. Lockman, "Ensuring integrity by adding obligations to privileges," in *Proc. 8th Int'l Conf. on Software Engineering*, pp. 92–102, 1985.

[7] L. F. Marshall, "Representing management policy using contract objects," in *Proc. IEEE First Int'l Workshop on Systems Management*, pp. 27–30, 1993.

[8] N. Szabo, "Smart contracts: Formalizing and securing relationships on public networks," *First Monday* **2**, Sept. 1997.

[9] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system." `http://nakamotoinstitute.org/bitcoin/`, Visited 13 Nov 2017 2008.

[10] Bitcoin Project, "Bitcoin: Bitcoin is an innovative payment network and a new kind of money." `https://bitcoin.org`, 2019.

[11] Ethereum Foundation, "Ethreum: Blockchain app platform." `https://www.ethereum.org`, Visited 23 Oct 2017 2018.

[12] Hyperledger, "Hyperledger fabric." `https://hyperledger-fabric.readthedocs.io/en/release-1.1/`, 2018.

[13] N. H. Minsky, "Law-governed systems," *Software Engineering Journal* **6**(5), 1991.

[14] Z. Milosevic, A. Josang, T. Dimitrakos, and M. Patton, "Discretionary enforcement of electronic contracts," in *Proc. 6th IEEE Int'l Enterprise Distributed Object Computing Conf.(EDOC'02)*, pp. 39–50, IEEE CS Press, 2002.

[15] H. Ludwig and M. Stolze, "Simple obligation and right model (SORM)-for the runtime management of electronic service contracts," in *Proc. 2nd Int'l Workshop on Web Services, e–Business, and the Semantic Web(WES'03), LNCS vol. 3095*, pp. 62–76, 2003.

[16] P. Linington, "Automating support for e–business contracts," *Int'l Journal of Cooperative Information Systems* **14**, pp. 77–98, June–Sep. 2005.

[17] C. Molina-Jimenez, S. Shrivastava, and M. Strano, "A model for checking contractual compliance of business interactions," *IEEE Trans. on Service Computing* **5**(2), pp. 276–289, 2012.

[18] K. O'Hara, "Smart contracts– dumb idea," *IEEE Internet Computing* **21**, Mar/Apr 2017.

[19] C. D. Clack, “Smart contract templates: legal semantics and code validation.” http://www0.cs.ucl.ac.uk/staff/C.Clack/research/JDigitalBanking-Clack-AuthorPreprint.pdf, 2019. Visited on 5 Dec 2020.

[20] JetForm, “The difference between an agreement and a contract.” https://www.jotform.com/blog/agreement-vs-contract/, July 2021. Visited on 8 Jul 2021.

[21] Stanford Encyclopedia of Philosophy, “Formation of contract at common law.” https://plato.stanford.edu/entries/contracts-theories/, Sept. 2015. Visited on 1 jul 2021.

[22] M. D. Flood and O. R. Goodenough, “Contract as automaton: The computational representation of financial agreements.” https://www.financialresearch.gov/working-papers/, Mar 2017. The Office of Financial Research (OFR) Working Paper Series.

[23] Richard Silberglitt and Brian G. Chow and John S. Hollywood and Dulani Woods and Mikhail Zaydman and Brian A. Jackson, “Visions of law enforcement technology in the period 2024–2034: Report of the law enforcement futuring workshop.” https://www.rand.org/pubs/research_reports/RR908.html, Dec 2015. Visited on 5 Jan 2021.

[24] M. di Angelo, A. Soare, and G. Salzer, “Smart contracts in view of the civil code,” in *Proc. 34th ACM/SIGAPP Symposium on Applied Computing (SAC ’19)*, 2019.

[25] P. D. Filippi and A. Wright, *Blockchain and the Law: The Rule of Code*, Hardvard University Press, 2018.

[26] E. Micheler and A. Whaley, “Regulatory technology: Replacing law with computer code,” *European Business Organization Law Review* **21**, pp. 349–377, 2020.

[27] J. Hazard and H. Haapio, “Wise contracts: Smart contracts that work for people and machines,” in *Proc. 20th Int’l Legal Informatics Symposium (IRIS’17)*, 2017.

[28] R. Koulu, “Blockchains and online dispute resolution: Smart contracts as an alternative to enforcement,” *A Journal of Law, Technology and Society* **13**, May 2016.

[29] V. Shmatikov and J. C. Mitchell, “Finite–state analysis of two contract signing protocols,” *Theoretical Computer Science* **283**, pp. 419–450, June 2002.

[30] M. Raskin, “The law and legality of smart contracts,” *Georgetown Law Technology Review* **1**(2), 2017.

[31] C. Sillaber and B. Waltl, “Life cycle of smart contracts in blockchain ecosystems,” *Datenschutz und Datensicherheit—DuD* **41**, Aug. 2017.

[32] Z. Zheng, S. Xie, H.-N. Dai, W. Chen, X. Chen, J. Weng, and M. Imran, “An overview on smart contracts: Challenges, advances and platforms.” https://www.henrylab.net/wp-content/uploads/2019/12/SmartContractFGCS__arXiv_.pdf, Dec. 2019. Visited on 1 jul 2021.

[33] R. Anderson, *Security Engineering: A Guide to Building Dependable Distributed Systems*, John Wiley & Sons, third ed., 2020.

[34] L. C. et al., “Smart contracts: 12 use cases for business and beyond.” http://digitalchamber.org/assets/smart-contracts-12-use-cases-for-business-and-beyond.pdf, Dec. 2016. Prepared by: Smart Contracts Alliance — In collaboration with Deloitte, Visited on 15 Jun 2021.

[35] T. Harrop, “What are ’machine readable laws’?.” https://medium.com/@tjharrop/what-are-machine-readable-laws-8c9f39159cb5, 2018. visited on 16 Mar 2021.

[36] Law.MIT.edu, “Computational law.” https://law.mit.edu, 2021. Visited on 5

Jul 2021.

[37] C. Law, "Nathaniel love and michael genesereth," in *Proc. 10th Int'l Conf. on Artificial Intelligence and Law*, 2005.

[38] M. Genesereth, "What is computational law?." https://law.stanford.edu/2021/03/10/what-is-computational-law/, Mar. 2021. Stanford University.

[39] H. Surden, "Computable contracts," *UC Davis Law Review* **46**(629), 2012.

[40] C. D. Clack and C. McGonagle, "Smart derivatives contracts: the isda master agreement and the automation of payments and deliveries." https://arxiv.org/pdf/1904.01461.pdf, Apr. 2019. arXiv:1904.01461 [cs.CY].

[41] C. D. Clack, V. A. Bakshi, and L. Braine, "Smart contract templates: foundations, design landscape and research directions." https://arxiv.org/pdf/1608.00771.pdf, Aug. 2016. arXiv:1608.00771 [cs.CY].

[42] S. D. Levi and A. B. Lipton, "An introduction to smart contracts and their potential and inherent limitations." https://corpgov.law.harvard.edu/2018/05/26/an-introduction-to-smart-contracts-and-their-potential-and-inherent-limitations/, May 2018. Harvard Law School Forum on Corporate Governance.

[43] Law Commission, "Smart contracts call for evidence." https://s3-eu-west-2.amazonaws.com/lawcom-prod-storage-11jsxou24uy7q/uploads/2020/12/201216-Smart-contracts-call-for-evidence.pdf, Dec 2020. Visited on 19 Dec 2020.

[44] C. Molina-Jiménez, I. Sfyrakis, L. Song, H. D. A. Nakib, and J. Crowcroft, "The benefits of deploying smart contracts on trusted third parties." https://www.researchgate.net/publication/332165550_The_Benefits_of_Deploying_Smart_Contracts_on_Trusted_Third_Parties, Apr. 2019. Visited on 11 Aug 2019.

[45] Law Commission, "Smart contracts: Summary of call for evidence." https://s3-eu-west-2.amazonaws.com/lawcom-prod-storage-11jsxou24uy7q/uploads/2020/12/Smart-Contracts-summary.pdf, Dec 2020. Visited on 19 Dec 2020.

[46] Z. Milosevic, A. Josang, T. Dimitrakos, and M. Patton, "Supporting business contracts in open distributed systems," in *Proc. Second Int'l Workshop on Services in Distributed and Networked Environments*, 1995.

[47] H. Weigand and L. Xu, "Contracts in e–commerce," in *Semantic Issues in E–Commerce Systems, IFIP - The International Federation for Information Processing, Vol. 111*, R. M. et al., ed., pp. 3–17, Springer, 2003.

[48] R. Craswell, "Contract law: General theories." https://www.sfu.ca/~allen/contracts%20general.pdf, 1999. Visited on 23 jul 2021.