Smart Contracts: Best Practices

JEFFREY D. NEUBURGER, WAI L. CHOY, AND KEVIN P. MILEWSKI, PROSKAUER ROSE LLP
WITH PRACTICAL LAW COMMERCIAL TRANSACTIONS

A Practice Note setting out best practices for using smart contracts on blockchains. This Note discusses functional and legal considerations for both standalone smart contracts and smart contracts used in conjunction with traditional written contracts (hybrid smart contracts) and discusses ways to maximize benefits and minimize risks when using smart contracts.

Smart contracting, which is the practice of using software coded to automatically execute an action on receipt of a certain input or the occurrence of another triggering event, is not a new concept. Software that automatically executes "if-then" logic has been used outside of the blockchain context for decades. It is the novel integration of that functionality with blockchain technology that may be significant in the new technological, business, and legal opportunities it creates. The combination of smart contract and blockchain technology provides the ability to:

- Automatically, securely, and verifiably execute agreed terms.
- Streamline electronic transactions.
- Remove "middlemen" from transactions.
- Establish enforceable digital legal contracts that are entered into and exist independently, without an associated traditional written agreement.

Blockchain-based smart contracts offer many unique benefits. For example, they:

- Do not need to rely on a traditional central authority.
- Are generally immutable.
- Execute automatically and irreversibly.

Nevertheless, there are numerous practical and legal aspects that parties must consider when using smart contracts. This Note sets out best practices and legal considerations for developing and implementing smart contracts on blockchains.

For more information about smart contracts and blockchain in a supply-chain context, see Practice Note, Blockchain and Supply Chain Management: Smart Contracts (W-017-3806).

FUNCTIONAL AND LEGAL CONSIDERATIONS

While neither blockchains nor smart contracts are required to enter into or perform legally enforceable contracts, using them provides both functional and potentially legal features, the benefits and risks of which legal advisors and their clients should consider when determining their suitability for specific transactions and applications.

FUNCTIONAL CONSIDERATIONS

From a functional perspective, some features that should be taken into account include:

- Limited or no central authority, see Limited or no Central Authority.
- Immutability, see Immutability.
- Automation, see Automation.
- Security and privacy, see Security and Privacy.
- Costs, see Costs.
- Functionality, see Functionality.
- Oracles, see Oracles.

Limited or no Central Authority

One significant feature of blockchain smart contracts is that they do not rely on an intermediary party or central authority to administer performance.

Many types of traditional written agreements ("traditional contracts"), such as escrow contracts, insurance contracts, or contracts for funds transfer contemplate a third-party central authority, be it an escrow agent, insurance agent, or bank, respectively. A smart contract on a blockchain can hold funds in escrow, issue an insurance payment on confirming the occurrence of a covered incident, or release funds on the occurrence of a pre-programmed condition, in each case without needing a central authority to coordinate and administer it.

Elimination of the central authority as a party necessary to administer transactions may minimize:

- Costs.
- Risk of system failure.
- Arbitrary delay.
- Mishandling of the transaction.

Many of the use cases for blockchain smart contracts involve consortia of founding or collaborating enterprises establishing and maintaining private or hybrid blockchains in which the consortium members exercise some degree of control that a traditional central authority might have (for example, the determination of who can participate in the blockchain and to what extent). This structure offers a middle-ground that leverages the technological benefits of blockchains and decentralization while retaining an overarching business framework.

Immutability

Blockchains and the smart contracts stored on them are immutable (practically impossible to change) because the code is distributed on the blockchain across the network and would require sufficient consensus of the network to alter. Once a smart contract is executed, its execution similarly cannot be reversed (although a new transaction could be made by the parties to effectively negate the result of that execution).

The deployment of a smart contract on a blockchain, therefore, helps to eliminate the risk of a party breaching its obligations, as the smart contract:

- Cannot be changed.
- Will be self-enforcing.
- Can be programmed to simultaneously execute both parties' obligations.

This makes it difficult (although not impossible, especially where the smart contract is linked to off-chain inputs or assets, which can be manipulated outside of the blockchain) for a party to a smart contract to avoid performance.

If there are coding mistakes, however, in the absence of a governing traditional contract a court reviewing a smart contract in a dispute may not have any evidence of the parties' meeting of the minds other than the incorrectly coded smart contract. Therefore, the erroneous code may be deemed to represent the understanding of the parties.

On truly immutable blockchains, a smart contract cannot be removed or altered once deployed. In such cases, to create an effect similar to modifying a smart contract the parties must use a workaround, such as:

- Deploying a new one (with the desired alterations) at a new address, "killing" or "self-destructing" the original one and then directing users to the address of the new smart contract instead.
- Leveraging the technical ability of smart contracts to be programmed to "call" and run the code of another smart contract to implement a smart contract structure at the outset that enables the redirection of a master smart contract's call to an erroneous smart contract to an alternate smart contract.

Automation

Blockchain smart contracts self-execute automatically on the blockchain. For example:

- In financial contexts, a smart contract could be used to release funds on the occurrence of a pre-programmed condition.
- In the Internet of Things, a smart contract could be pre-programmed to unlock a car or hotel room on the receipt of funds. For more information on the Internet of Things, see Practice Note, The Internet of Things: Key Legal Issues (W-002-6962).

Because smart contracts execute automatically as programmed, there is little room for discretionary enforcement of smart contract terms by the parties. For example, if the parties have built a late-payment penalty into the smart contract, the party enforcing the penalty will not have the option to discretionarily waive the penalty for a given pay period unless that functionality is coded into the smart contract.

Security and Privacy

Parties seeking to transact using smart contracts should consider the security of their smart contract code and carefully audit it before deploying it on a blockchain, especially on public or hybrid blockchains. Poorly written or insufficiently tested code can leave a smart contract exposed to security threats, including unauthorized parties being able to trigger or otherwise interact with the smart contracts. For example, in the landmark attack on The DAO (The Distributed Autonomous Organization) in 2016, a hacker exploited coding vulnerabilities in a smart contract deployed on the Ethereum blockchain to extract cryptocurrency stored in it.

The level of privacy and confidentiality inherent in smart contracts can vary depending on whether the smart contract is stored on a public, private, or hybrid blockchain. If the blockchain is public, then the terms of the smart contract may be visible to all users of the blockchain, so parties should remain aware of the visibility of their transactions. Alternatively, private blockchains will, by their nature, be private, and the visibility of smart contracts on the blockchain will be limited to those participants that are given the appropriate permissions. Hybrid blockchains, by having both publicly visible elements as well as elements that are restricted to permissioned parties, provide an ability for greater security and privacy where desired while maintaining some of the transparency and other benefits offered by public blockchains.

Costs

Blockchain-based smart contracts can also offer reduced transaction costs as compared to the costs of conducting the same transactions under traditional contracts. Although a blockchain may require a payment to run a smart contract, depending on the smart contract and blockchain, these costs may be small in comparison to the amount traditionally spent on transfer fees, third-party agent fees, commissions, and escrow accounts.

Functionality

Although blockchains that support smart contracts offer vast possibilities, there are inherent limitations to smart contracts. Smart contracts can only execute conditional logic and may not have enough flexibility to apply more fluid concepts that might be embodied in a traditional contract.

Oracles

Smart contracts can be written to automatically execute based on external triggering events (such as weather, geographic location, or timing) through "oracles" which are off-blockchain sources of digital information that translate outside events into smart contract readable data. Although oracles open up numerous possibilities, they also:

- Introduce potential sources of failure, erroneous data or anomalies that may reduce the reliability of smart contract transactions.
- May reduce security by offering a point of entry.

LEGAL CONSIDERATIONS

Although properly prepared and deployed smart contracts are self-executing, they and their outcomes are not necessarily legally enforceable. To be legally enforceable, a smart contract and the process of "agreeing" to a smart contract must have all the attributes that make traditional contracts enforceable.

For example, a blockchain-based smart contract used for payroll purposes may be programmed to automatically execute a payment to employees on the occurrence of a condition such as a biweekly pay date, but the smart contract itself would not legally entitle the employee to payment unless the coded terms of the smart contract legally constitute:

- An offer and acceptance.
- An exchange of consideration (in this case, the employer's agreement to pay each employee and the employee's agreement to perform the services for which the employee is to be paid).

Smart contracts also may not be legally effective in transferring legal ownership of tangible assets tokenized on a blockchain (that is, using a security token to digitally represent the asset to be transferred). To effectively achieve a change in legal ownership of a tangible asset, a smart contract may need to satisfy applicable legal ownership transfer requirements (for example, in the real estate context, property transfer and recordation formalities).

Deploying and using a smart contract in the absence of a governing traditional contract also raises conceptual issues. In many cases, the function of the smart contract will match the intent of the parties, because the parties will code the functions properly into the smart contract. If there are mistakes in coding, issues with oracles or other unexpected problems, however, a party may not receive its expected benefit of the bargain. Without a governing traditional contract to look to for the parties' intent, however, there may not be any legal recourse for the purportedly erroneous result of the smart contract's execution. If the smart contract is structured to be a legally enforceable agreement with the parties' intent clearly indicated within it, or if the smart contract is governed by a traditional contract, the injured party would have a clearer means of holding the other party accountable to fulfill the parties' intent.

The threshold legal questions, then, are whether and under what conditions smart contracts can be legally binding contracts, how those that are differ from traditional written contracts, and what considerations should be taken into account when deciding whether to use a standalone smart contract or a smart contract that is governed by a traditional contract.

Formation

Contract law in the US is governed by each individual state's laws. While the laws vary from state-to-state, US state common law generally requires that, for an arrangement to be considered a legally binding contract, there must be:

- A meeting of the minds.
- An offer and acceptance.
- An exchange of consideration.

For example, one party may offer the other a product or service and the receiving party can accept the offer in a variety of ways depending on the nature of the offer, including by performance or an expression of agreement to be bound by the terms of the offer.

While the analysis will depend on the coding of the smart contract and any ancillary documents, a smart contract may be capable of manifesting:

- A meeting of the minds.
- An offer.
- Acceptance of the offer.
- Consideration.

A transfer of funds via a smart contract might serve as the offer or acceptance, and as a part of the consideration. For example, funding a smart contract with cryptocurrency, with subsequent payouts made contingent on a counterparty's performance, might be considered an offer to contract if it is meant to entice action by a counterparty. If, however, the sending of cryptocurrency to a smart contract triggers the execution of the smart contract, then that sending might be considered the acceptance of the terms of the smart contract and as consideration.

The transfer of payment to a smart contract that is coded to provide services on receipt of that payment (for example, unlock a hotel room door via an Internet of Things device) could serve as mutual consideration.

To support the position that a smart contract is legally binding, parties to the smart contract should work to make sure the terms are sufficiently definite and communicated to all contracting parties during the development and deployment process.

Because smart contracts are coded, there may be some problems with establishing a meeting of the minds if not all parties can understand the essence of the smart contract terms or if the code fails to accurately represent either the offer or the acceptance. If either party is mistaken about the terms of the smart contract, a court might be reluctant to consider it legally enforceable against the unaware party.

Consortia can play a helpful role in establishing standards and common practices among parties transacting on a blockchain using smart contracts.

Electronic Legal Contracts

If a smart contract contains the requisite contractual ingredients, the fact that it is executed digitally and without the immediate approval of a human agent should not inhibit its enforceability as a legal contract. State versions of the Uniform Electronic Transaction Act (UETA)

and the Federal E-SIGN Act make clear that an electronic record or electronic signature will not be denied legal effect simply because it is in electronic form. Further, the UETA and the Federal E-SIGN Act make clear that the use of "electronic agents" by parties (which include computer programs or automated means) can be a valid means of establishing and executing a binding contract if the actions of the electronic agent are legally attributable to the parties to be bound.

The cryptographic key with which blockchain-based smart contracts are signed and acknowledged, which use asymmetric public-private key encryption, might be considered the "electronic signatures." While the theory has not been tested in court, it is likely that a smart contract deployed on a blockchain can be an "electronic agent," and the parties' electronic signatures can serve as evidence of the parties' intent to contract.

Some states have taken steps to remove any doubt about the enforceability of smart contracts by adopting pro-smart contract laws. For example, rather than waiting for courts to confirm the contractual validity of smart contracts, in March 2018, Tennessee adopted a law that states, "Smart Contracts may exist in commerce. No contract relating to a transaction shall be denied legal effect, validity, or enforceability solely because that contract contains a Smart Contract term." (Tenn. Code. Ann. § 47-10-201 and §47-10-202.) Under the Tennessee law, smart contracts are defined as "an event-driven program, that runs on a distributed, decentralized, shared, and replicated ledger and that can take custody over and instruct transfer of assets on that ledger" (Tenn. Code. Ann § 47-10-201).

Other states, such as Arizona, Wyoming, Nevada, and Ohio have passed similar smart contract and blockchain legislation to various extents, which generally give recognition to blockchain transactions (including those carried out by smart contracts) by including blockchain within the definition of electronic records.

Enforceability

Even if smart contracts meet the basic legal requirements for enforceability, some types of contracts must observe certain formalities to be considered legally binding.

States with a statute of frauds (which may be based on the Uniform Commercial Code's model statute of frauds, for example) require that certain agreements be in writing to constitute a legally enforceable contract. Written code and any data stored in a smart contract that incorporates the requisite elements of a legally enforceable contract likely can satisfy any statute of frauds requirements but the mechanics of finalizing the code for a given smart contract and deploying it on a blockchain will be relevant to the analysis. Under most statute of frauds provisions, the writing must be signed by the parties to be contractually bound. Typewritten signatures and electronic records stored by computing devices are well-established to be sufficient to satisfy the requirements of statutes of frauds, so private key digital signatures should be capable of satisfying the writing requirement.

Additionally, smart contracts may be unenforceable despite compliance with formation requirements. Smart contracts that might otherwise satisfy legal requirements may be void or voidable based on a lack of contractual capacity (for example, where an individual attempting to enter into a contract is a minor) or where the performance of the smart contract violates public policy or advocates criminal activity.

It remains to be seen how courts will go about interpreting smart contracts. For smart contracts that implement terms of or are otherwise associated with a traditional written contract, parties will have to be wary about how they characterize the smart contract in relation to the traditional contract, as courts may or may not view the smart contract code as an extension of the traditional contract.

Consumer Contracts

Courts are reticent to enforce contracts against individual consumers where the consumer has not received adequate notice of the terms of the contract. A court may determine that a consumer did not have sufficient notice if they cannot review the code of the smart contract. As such, parties seeking to use smart contracts with consumers should consider:

- Carefully memorializing the smart contract terms in a separate document (for example, in the Terms of Service on a vendor's website through which smart contract-powered transactions are initiated).
- All of the applicable state and federal laws that generally apply to consumer contracts.

Jurisdiction/Choice of Law

Blockchains can be borderless and enable participants in multiple jurisdictions to transact with each other. Parties and courts may grapple with choice of law and jurisdiction issues when dealing with smart contract disputes, especially where there is no governing traditional contract or, in the case of a smart contract on a consortium blockchain, an overarching consortium traditional contract in which the participants in that blockchain have agreed on applicable terms.

Generally, in the US, courts tend to respect the parties' choice of law provisions. Courts may disregard choice of law provisions where the chosen jurisdiction lacks a substantial relationship with the parties or where application would violate public policy. In the absence of a choice of law provision, depending on the substance of the smart contract and choice of law rules applied by the relevant court, courts may look to:

- The domicile of the parties.
- The parties' IP addresses.
- Where the contract was:
  - negotiated;
  - coded;
  - executed; and
  - performed.
- Prior agreements between the parties.

Despite attempts by the parties to a smart contract to prepare for any future legal action, courts may still struggle to enforce the terms of a smart contract given their jurisdictional restrictions. To hear a matter, a court must generally have both:

- Personal jurisdiction over the parties.
- Subject matter jurisdiction over the issues in the case.

With blockchain, it is possible that some parties to a contract may be anonymous. Unless the parties to a smart contract are aware of the identities of the parties or beneficiaries of the smart contract, courts

may struggle to establish personal jurisdiction necessary to hear a case in sole reliance on a public key or pseudonymous identifier as opposed to an identity. Additionally, courts may be limited in their ability to hear a case depending on the nature of the underlying transaction and whether they have subject matter jurisdiction over the issues in the case.

Limits to Enforcement

Discussions of judicial enforcement of smart contracts are complicated by the fact that smart contracts are self-enforcing. If a smart contract is programmed to execute a certain function based on satisfaction of a certain condition, then it will execute on the occurrence of that condition, and there is little a court can do to trigger or prohibit its execution (given the immutability of blockchain smart contracts).

Courts ultimately may find ways to levy their power to address smart contract disputes, whether by using their equitable power to make injured parties whole:

- Through an off-blockchain transaction.
- By compelling defaulting parties to:
  - use their private keys to digitally sign a transaction; or
  - initiate kill mechanisms within the smart contract.
- Through a subsequent smart contract or blockchain transaction that effectively reverses the effect of a previously executed erroneous smart contract.

At present, parties should factor the uncertainty in how courts will handle technical limitations to smart contract enforcement into their risk calculations when deciding to contract using a blockchain-based smart contract, especially in the absence of a governing traditional contract.

UCC Application

It remains to be seen how the Uniform Commercial Code (UCC) will be interpreted in the smart contract context. For example, it is not entirely clear how smart contracts should be characterized for purposes of UCC Article 9. Whether smart contracts are considered general intangibles or investment property, for example, will ultimately help determine how parties to a financial transaction could perfect a security interest in a smart contract or the underlying assets. The Uniform Law Commission, together with the American Law Institute, is currently engaged in a study to review the UCC in light of blockchain and other emerging technologies.

BEST PRACTICES

Although smart contracts are not a new concept, their use cases in the blockchain context are still developing. Parties would be best advised to build in fulsome protections within the smart contract itself, in the governing traditional contract if there is one, or in both to prepare for unintended scenarios.

PRELIMINARY CONSIDERATIONS

Hybrid or Standalone

First, parties should determine whether the smart contract needs to be legally enforceable or have legal effect, and, if so, whether it should be the sole legal instrument or linked to a traditional contract as a "hybrid smart contract." Best practice is to employ a hybrid smart contract construct, with a traditional contract that references the execution of certain of its terms through a smart contract and that addresses, among other things, issues that cannot be easily addressed in the code of a smart contract such as:

- Governing law.
- Jurisdiction.
- Venue.
- Dispute resolution.
- Force majeure.
- Indemnification and other remedies for:
  - coding errors or oversights;
  - erroneous oracles or external data sources; or
  - other technological failings.

If deployed on a consortium blockchain, the parties could broadly cover their respective smart contract activities with a generally-applicable set of legal terms embodied in the governing consortium agreement.

Public or Private

Enterprises considering using blockchain smart contracts should decide whether to use a public, private, or hybrid blockchain. The threshold consideration is whether the existence and terms of smart contracts should be kept private or confidential to the participants on the blockchain, or whether the parties are comfortable with distributing the smart contract and transaction details among public nodes on the blockchain.

Further, from a confidentiality and data security perspective, the parties should evaluate whether the pertinent data and information should be stored on the blockchain or, instead, off-chain, with the smart contract programmed to call out to off-chain data sources only when inputs of that data are necessary for the smart contract to execute. This will be dictated by the confidentiality and data security needs of the parties.

For most enterprises, a private or hybrid blockchain that maintains the core benefits that blockchain offers while enabling some degree of access control and opacity will likely be most appropriate.

Drafting

The parties should ensure that negotiated business and legal terms are adequately communicated to the programmer. Unlike with the drafting of traditional contracts, where the lawyer may have been privy to the negotiations, programmers might have little insight into the objectives of the parties. As such, they should be given a clear mandate on what terms should be reflected in the smart contract code, and how to account for potential alternative scenarios.

Review

Even if the programmer has a clear sense of the contractual terms, all parties should carefully review the source code for any errors and should ideally have a third-party professional review the code as well. Unless all parties are well versed in computer programming languages, there may be confusion or misrepresentations surrounding the code as actually written, which could undermine the

meeting of the minds required to form a legally binding contract and could lead to operational difficulties. This concern is exacerbated by the immutability of smart contracts. One way to ensure that the smart contract operates as intended is to run it in a cordoned-off environment, without real-world effect and to assess their functions.

Ancillary Agreements

Depending on the transaction, multiple third-parties may be involved with setting up the smart contract. For example, an outside programmer may code the smart contract, or an outside firm may establish the private blockchain on which the smart contract runs. Parties may rely on their traditional contracts with vendors for these functions, but they should ensure that those agreements include representations, warranties, and covenants that the code operates and will operate as expected, and does not and will not infringe any third-party's intellectual property rights. The parties should also include an obligation of the vendor to indemnify with respect to any party claims resulting from a breach of any of these representations, warranties, and covenants.

Governance

The parties should institute governance mechanisms to regulate and self-enforce remedies or troubleshoot performance. If a blockchain is a consortium blockchain, then governance issues could be generally addressed in a consortium agreement to which all members and participants are a party.

Kill Mechanism

As a fail-safe mechanism, parties may want to include in their smart contract a kill function that the parties can exercise if an issue warrants it. The parties should carefully consider the circumstances in which a party should be able to run that function (as, for example, the ability to activate the kill function could amount to a termination for convenience right). For information on termination for convenience, see Standard Clause, General Contract Clauses: Term and Termination: Drafting Note: Termination for Convenience (2-507-0812).

Amendment Procedure

Because the only solution for creating the effect of amending a blockchain smart contract is to deploy and use a new one instead, the parties should consider what to do when an amendment to the core terms of a smart contract is required. If the parties are using a hybrid smart contract approach, they can provide in the traditional contract a requirement to negotiate a replacement smart contract in good faith under specified conditions.

HYBRID SMART CONTRACTS

It is possible for standalone smart contracts to rise to the level of a legally binding contract under state laws, but uncertainty remains about their use and interpretation, in part due to a lack of standardization and specifically applicable jurisprudence. Until there is greater clarity around standalone smart contracts, parties should use hybrid smart contracts (smart contracts that are governed by or which implement provisions of a traditional contract) to handle complex legal issues with a traditional contract they know a court will enforce.

The traditional contract associated with the hybrid smart contract should specify key terms of the agreement including each of the following:

- Responsibility for coding and deployment, see Responsibility for Coding and Deploying the Smart Contract.
- Alignment between the traditional and smart contract, see Alignment Between the Traditional Contract and the Smart Contract.
- Precedence between the traditional and smart contract, see Precedence Between the Traditional Contract and the Smart Contract.
- Governing law, jurisdiction, and venue, see Governing Law, Jurisdiction, and Venue.
- Representations and warranties, see Representations and Warranties.
- Indemnity, see Indemnity.
- Insurance, see Insurance.
- Fallback mechanisms, see Fallback Mechanisms.
- Collateral issues, see Consideration of Collateral Issues.
- Force majeure, see Force Majeure.
- Remedies in case of technology failure or error, see Remedies in Case of Technology Failure or Error.

Responsibility for Coding and Deploying the Smart Contract

As an initial step, any smart contract should delineate, in a traditional contract, the roles of the parties regarding the coding and deployment of the smart contract. The parties should clearly define coding standards, review and testing procedures and how the parties will agree when the code is ready to deploy, as well as who will bear any transaction costs of running the smart contract on the applicable blockchain.

Alignment Between the Traditional Contract and the Smart Contract

The parties should give special attention to the interplay between the traditional contract and smart contract. The traditional contract should align with and, if possible, reference the code within the smart contract to avoid any confusion among courts in determining whether the smart contract embodies the terms of the written agreement. For information on integration of agreements and ancillary documents, see Standard Clause, General Contract Clauses: Entire Agreement (9-520-4139).

Precedence Between the Traditional Contract and the Smart Contract

If there is a conflict between the terms of the traditional contract and the smart contract, it should have clarity as to which terms control. Courts may organically view smart contracts in a hybrid smart contract structure to be subordinate to the traditional contract, but some jurisdictions may view all documentation of the transaction equally and scrutinize the intent of the parties. Best practice would be to specify in the traditional contract that its terms will control over any conflicting term or outcome of the smart contract. For more information on order of precedence, see Standard Clause, General Contract Clauses: Entire Agreement: Drafting Note: Ancillary Documents (9-520-4139).

Governing Law, Jurisdiction, and Venue

Specifying governing law, jurisdiction, and venue in a traditional contract will bypass the issue of deciphering which jurisdiction applies to judicial interpretation of smart contracts, which could become complicated given the borderless nature of blockchain. However, despite the contractual clauses, depending on the nature of the claim and the geographic locations of the parties, it is possible that the specified courts may not be able to actually exercise jurisdiction over a defendant. For more information on governing law, jurisdiction, and venue, see Practice Note, Choice of Law and Choice of Forum: Key Issues (7-509-6876).

Representations and Warranties

Representations and warranties will be critical to allocating risk surrounding the creation and deployment of a smart contract. A party to a smart contract could potentially claim that they did not understand the nature of the transaction embodied in a smart contract given the technical nature of the coding. A representation providing that each party has or will have fully reviewed the terms of the smart contract using qualified programmers and understands them at the time the smart contract is deployed will bolster the conclusion that a valid, enforceable smart contract has been formed. For more information on representations and warranties, see Standard Clause, General Contract Clauses: Representations and Warranties (2-519-9438).

Indemnity

In addition to typical indemnity clauses, the parties should include in the governing traditional contract provisions to address smart contracts specifically. For example, indemnities protecting the parties that did not code the smart contract with respect to intellectual property infringement by the smart contract and damages resulting from improper operation of the smart contract or from other errors in the smart contract. Mutual or more nuanced indemnities may be required where both parties to a transaction collaborate in drafting, reviewing, and testing the smart contract. For more information on indemnity, see Practice Note, Indemnification Clauses in Commercial Contracts (5-517-4808).

Insurance

Although it remains to be seen how insurers will cover liabilities resulting specifically from smart contract failures, such as faulty oracle performance or coding mistakes, insurance will likely play a role in assessing the use of smart contracts. Parties entering hybrid smart contracts should determine whether their existing insurance covers smart contracts and related issues. For more information on insurance coverage, see Practice Note, Insurance Policies and Coverage: Overview (9-505-0561).

Fallback Mechanisms

Blockchains, when coded properly, have built-in integrity, but external data sources on which a smart contract relies may not provide the same protections. If parties are relying on timely fulfillment of obligations coded into a smart contract tied to an oracle, and that oracle lapses, parties should consider referencing a back-up oracle or provide for off-blockchain remedial measures to compensate for shortfalls. Parties may also consider linking a back-up payment account to distribute funds under the smart contract if the smart contract or primary distribution account lacks the requisite amount of funds. For more information on remedies, see Damages for Breach of Commercial Contracts Checklist (5-555-7166) and Practice Note, Contracts: Equitable Remedies (0-519-3197).

Consideration of Collateral Issues

Drafting a traditional contract provides parties with considerable flexibility in spelling out what situations should lead to a termination of the smart contract, particularly regarding collateral performance issues that would otherwise be undetected by the smart contract. This may include failure to receive a key deliverable or failure of an objective milestone to be reached. In this case, parties may provide that the occurrence of a collateral issue will allow for the initiation of the kill function in the smart contract. For more information on contract termination, see Standard Clause, General Contract Clauses: Term and Termination (2-507-0812).

Force Majeure

Force majeure clauses are often-overlooked but integral parts of all contracts, and they should be given considerable attention for smart contracts; in effect, force majeure provisions relieve a party of their contractual obligations when performance is prevented due to forces beyond the party's control. The provision might consider:

- Unforeseeable events in the smart contract environment.
- Who might be responsible if the smart contract is penetrated by an unauthorized third-party.
- Whether the smart contract should contemplate a reversion to a traditional contract system in the event of technology failure.

For more information on force majeure, see Practice Note, Force Majeure Clauses: Key Issues (5-524-2181).

Remedies in Case of Technology Failure or Error

Due to blockchains' self-executing and immutable nature, parties have few options on a blockchain when problems arise other than:

- Following through with the smart contract.
- Killing or self-destructing the smart contract (if provided for).
- If the smart contract is one that will run when "called" by a master smart contract and that master smart contract was programmed with the ability to change the smart contract to which it makes calls, using that functionality to redirect those calls to an alternate smart contract before a call is triggered.
- Agreeing to no longer abide by the smart contract and separately remedying any effect of the smart contract running.

Best practice would be to use a governing traditional contract to clearly delineate how a problem with a smart contract deployed under the traditional contract will be remedied. For example, the parties may agree that a party must return digital assets erroneously transferred to it if such an error occurs.

For clarity, parties should include in the traditional contract a general statement that, despite any results of a smart contract running, the smart contract's purpose is limited to carrying out the intent of the traditional contract and, if running the smart contract results in an inconsistent result, then the parties will cooperate in good faith to carry out the intent of the parties embodied in the traditional contract.

STANDALONE SMART CONTRACTS

Best practices for drafting standalone smart contracts largely depend on the risk and size of the transaction. The greater the risk, the more consideration that needs to be given to protections. Drafters should review each of the points set out above for hybrid smart contracts and consider how they can be reflected within the code of the smart contract. In addition, standalone smart contract programmers should pay special attention to:

- Function, see Function.
- Risk allocation, see Risk Allocation.
- Accuracy, see Accuracy.
- Recourse, see Recourse.
- Force Majeure, see Force Majeure.

Function

The parties should consider whether they want their smart contract to rise to the level of a legally enforceable contract. There may be circumstances in which the rule of "code is law" in which the code simply runs and the result is, as a matter of fact, the definitive result, may be preferable. If the parties determine that they want the smart contract to be legally enforceable, special consideration should be given to ensuring that the common law elements of offer, acceptance, and consideration are met, along with any other requirements associated with the content of the transaction or the jurisdiction that parties are anticipating will govern the smart contract.

Risk Allocation

It is rare that all parties to a traditional contract share the same amount of risk in the event of default, and the same applies to smart contracts. Parties should evaluate the amount of risk they bear in the event of an error or malfunction in performance, because traditional protections like representations, covenants, and indemnity may be difficult to code into the smart contract. While blockchains, when coded properly, have inherent integrity, parties should remain cognizant of any potential points of failure introduced into the smart contract mechanism (for example, oracles). For more information on risk allocation, see Practice Note, Risk Allocation in Commercial Contracts (4-519-5496).

Accuracy

Accuracy is vital for any smart contract, but especially for standalone smart contracts that the parties wish to be legally enforceable. The parties should ensure that the smart contract code accurately reflects the understanding of all parties, including, but not limited to:

- Triggering events.
- Kill or self-destruct provisions.
- Provisions that call to, or otherwise interrelate with, other smart contracts, oracles, or data sources.

Without a traditional contract supporting the smart contract, courts or anyone reviewing the code will have little basis for determining the true intent of the parties beyond the code. The parties should implement procedures for both parties' technical teams to review the code to confirm accuracy. If the smart contract's code does not operate as expected the parties may be unable to remedy the issue without significant expense and additional transactions to correct the immutable result of the running of the smart contract and, if a party refuses to cooperate in the remediation, the other party may be without legal recourse.

Recourse

Without a traditional contract to guide courts on the intentions of the parties, courts may struggle to derive the expectations of the parties and to provide sufficient recourse. Care should be taken to build into the smart contract as much indication of intent as possible.

Force Majeure

Parties should consider drafting force majeure concepts into their written code. Such code could be designed to recognize specified force majeure events (for example, through oracles), and might provide for the suspension of performance on the detection of a force majeure event.

For more information on force majeure, see Practice Note, Force Majeure Clauses: Key Issues (5-524-2181).