

Blockchain Sandbox Demo: SLA Management

John Wilmes
Director of IoE Projects
jwilmes@tmforum.org

- A continuously growing list of records, called blocks, which are linked and secured using cryptography.
- Each block contains typically a hash pointer as a link to a previous block, a timestamp and transaction data.
- By design, blockchains are inherently resistant to modification of the data.
- Functionally, a blockchain can serve as "an open, distributed ledger that can record transactions between two parties efficiently and in a verifiable and permanent way."
- For use as a distributed ledger a blockchain is typically managed by a peer-to-peer network collectively adhering to a protocol for validating new blocks.
- Once recorded, the data in any given block cannot be altered retroactively without the alteration of all subsequent blocks and a collusion of the network majority.

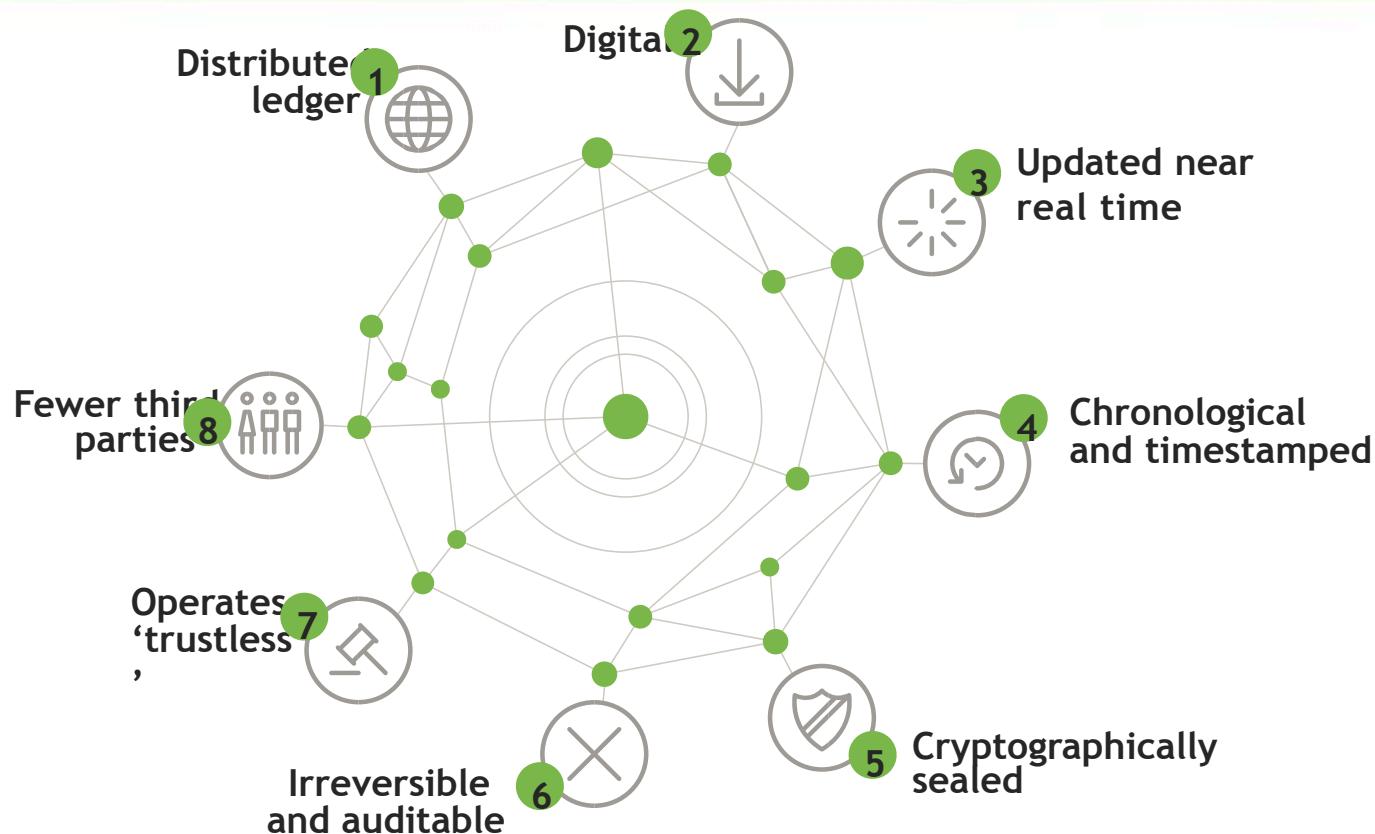
- Digital currencies (and digital assets in general) require a solution to the “double-spend problem” - the ability to spend a digital token twice
- Before Bitcoin, this required a single trusted third party, which is a single point of failure from both technical and trust viewpoints¹
- Bitcoin solves the problem with a distributed “chain” of “blocks” that contain verified, timestamped financial transactions
- Bitcoin blocks are hard to create, easy to verify, and globally visible - eliminating the need for a centralized authority
- Bitcoin’s blockchain is a public distributed ledger running on many nodes communicating over a peer-to-peer network protocol

¹ <https://en.wikipedia.org/wiki/Double-spending>

- Ethereum solves the double-spend problem in a way similar to Bitcoin, but more favorable to decentralized distribution of security
- Ethereum blocks contain accounts rather than transactions - its blockchain tracks “the transfer of value and information between accounts”² rather than focusing only on financial transactions
- Ethereum is programmable - cryptocurrency is just one use case
- Each Ethereum node runs a virtual machine that can execute code - every Ethereum node can execute the same instructions
- Ethereum is for “applications that automate direct interaction between peers or facilitate coordinated group action”²
- Ethereum blockchains can be public, semi-private, or private

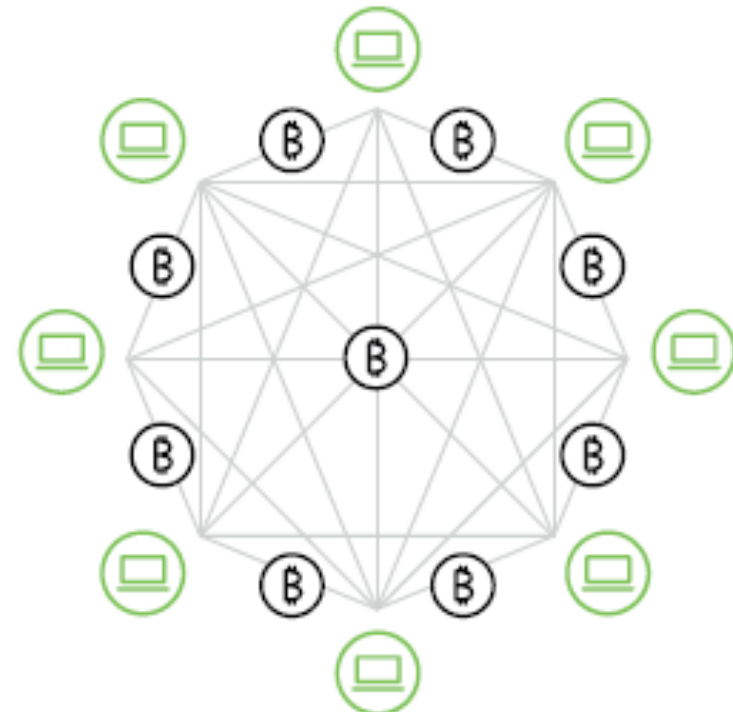
² <http://ethdocs.org/en/latest/introduction/what-is-ethereum.html>

- A smart contract stores and executes technical aspects of a business contract as software
- Typically implemented on a blockchain that supports code execution using a specialized smart contract language
- Often refers to, but is not (yet) in itself, a legal contract
 - The subject of a new collaboration between TM Forum and Stanford University
- Typically driven by external events but can call other smart contracts
- Applicable to many use cases that involve state changes over time with associated value
- Executes as coded, not necessarily as expected (e.g. successful theft of funds last year from DAO / Ethereum by people exploiting code defects)





Central database or clearinghouse



Blockchain





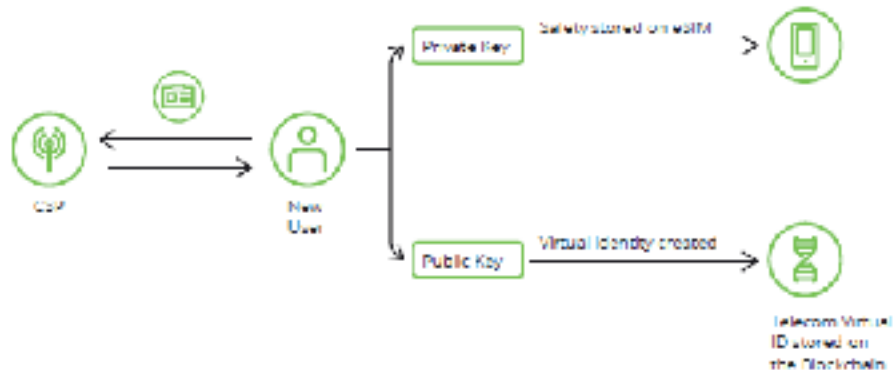
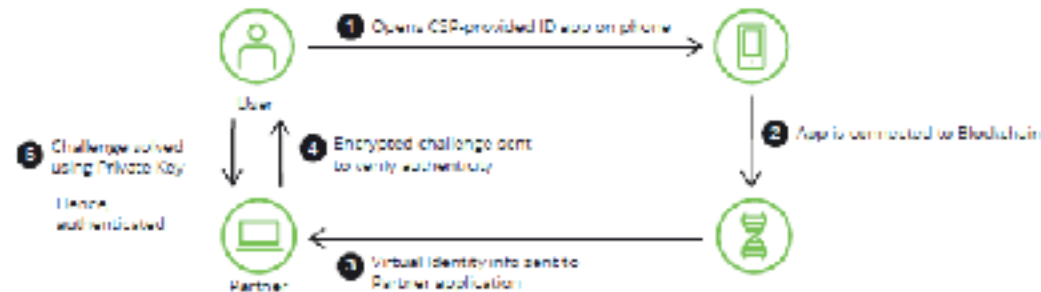
Current Core and VAS Opportunities		Upcoming Telecom Trends Opportunities	
<div>Fraud area</div> <div>Fraud Management</div> <div></div>	<div>Identity-as-a-Service</div> <div></div>	<div>5G Enablement</div> <div></div>	<div>IoT Connectivity</div> <div></div>
<div>Description</div> <div>Implement blockchain for data and value exchange within and between networks to reduce fraud</div>		<div>Platform to enable a new generation of access technology selection management, required for the realization of 5G network potential</div>	
<div>Benefits</div> <div><ul style="list-style-type: none">• Reduced losses due to fraud• Reduced costs for fraud detection applications</div>		<div><ul style="list-style-type: none">• Common platform to provide seamless connectivity</div>	
<div><ul style="list-style-type: none">• Decreased costs of implementing Identity Management• Additional revenue stream</div>		<div><ul style="list-style-type: none">• Common platform for IoT devices to communicate• Enable micropayments</div>	

Figure 4: Potential use cases of blockchain for a CSP

I. ID Creation



III. Authentication



- Blockchain stores identity transactions
- CSP provides Identity as a service to partners
- CSP creates a digital identity for subscriber
- Private key for identity is stored on eSIM
- CSP creates virtual identity using public key from digital identity, then adds digital signature using its own private key
- Pointer to virtual identity added to blockchain
- Copy of the ledger entry is sent to merchant
- e-commerce site takes public key from the virtual identity, encrypts a challenge and sends it to the subscriber app which decrypts it and responds
- The e-commerce site generates an e-commerce virtual identity which is stored in the ledger
- Next time the subscriber visits same e-commerce site, can be authenticated the same way
- Ledger holds subscriber's history / preferences
- e-commerce site can use related insights
- Subscriber can use the same virtual identity to login to a different e-commerce site the same way
- CSP virtual identity can be used to help create other virtual identities e.g. a travel virtual identity

Features that differentiate blockchain from other MMT enablers

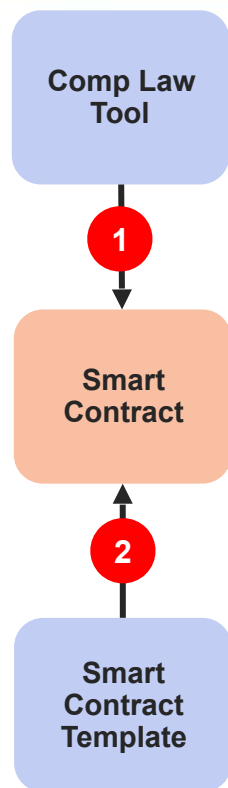


Feature	Differentiator
Works in a "trustless" environment	The distributed ledger performs the accounting and audit functions neutrally and reliably.
Distributed ledger requires no intermediaries	Transactions can be conducted without the need for third parties like clearinghouses.
Decentralized consensus ensures integrity	The negotiation protocol ensures that all participants see and approve a common view.
Immutability prevents tampering	It is infeasible for a bad actor to change the distributed record in all locations.
Near-real-time updates maintain consistency	Although transaction times vary, consistency across all copies of the ledger is quickly achieved.
Verifiability supports auditing	Any participant can go back to any transaction at any time and prove its integrity.
Timestamps & sequencing support chronology	The distributed ledger establishes the timing of transactions as well as their content.
Scripted transactions support smart contracts	Blockchain enables the execution of coded "contracts" which are themselves part of the irrevocable record.
Fractional currency for micro-transactions	The cryptocurrencies supported by blockchain allow extremely small financial transactions that are impractical or impossible with traditional settlement systems.

Blockchain feature	Privacy	Identity	Security	Asset	SLA	Digital rights
Works in a “trustless” environment	X	X	X	X	X	X
Distributed ledger requires no intermediaries	X	X	X	X	X	X
Decentralized consensus ensures integrity	X	X	X	X	X	X
Immutability prevents tampering	X	X	X	X	X	X
Near-real-time updates maintain consistency	X	X	X	X	X	X
Verifiability supports auditing	X	X	X	X	X	X
Timestamps & sequencing support chronology		X	X	X	X	X
Scripted transactions support smart contracts			X	X	X	X
Fractional currency for micro-transactions					X	X

- Service level agreements (SLAs) and related business processes have been exhaustively defined by the TM Forum and other organizations, but still incur risk of litigation over significant sums.
- Sources of risk include
 - Legal agreements that are discovered to be inconsistent, incomplete and/or incorrect
 - Processes that do not reflect legal agreements
 - Disagreements over the application of service level criteria and/or measurements
 - Long intervals between settlements leading to “bill shock”.
- The demo described here attempts to demonstrate the technical possibility of addressing the above sources of risk:
 - Use computational law tools to ensure the validity of the legal contract and its mapping to the smart contract - not demonstrated as of July 2017
 - Use a blockchain to provide a shared, immutable record of criteria, measurements and events as well as a platform for smart contracts - demonstrated
 - Use smart contracts running on a blockchain to automate the detection, assessment, documentation, and settlement of SLA violations - demonstrated
 - Use token currency supported by a blockchain to make frequent, small settlements using token micropayments - demonstrated

- Open source Ubuntu Linux
- Open source Ethereum blockchain
- Open source Ethereum blockchain explorer
- Open source smart oracle
- Open source code generator consuming the swagger.json file defining the TM Forum SLA Management API
- Open source MongoDB database
- Running instance of the SLA Management API and its underlying logic and storage using the above generated NodeJS code over MongoDB
- Simple in-house HTML5 app as an SLA management dashboard



1. Legal staff use a template to create a contract representing a service level agreement (SLA), validate the contract using a computational law tool, and distribute it for electronic signature.
2. Technical staff use the validated electronic contract and a smart contract template to create and deploy a smart contract that will manage the execution of the previously defined SLA.

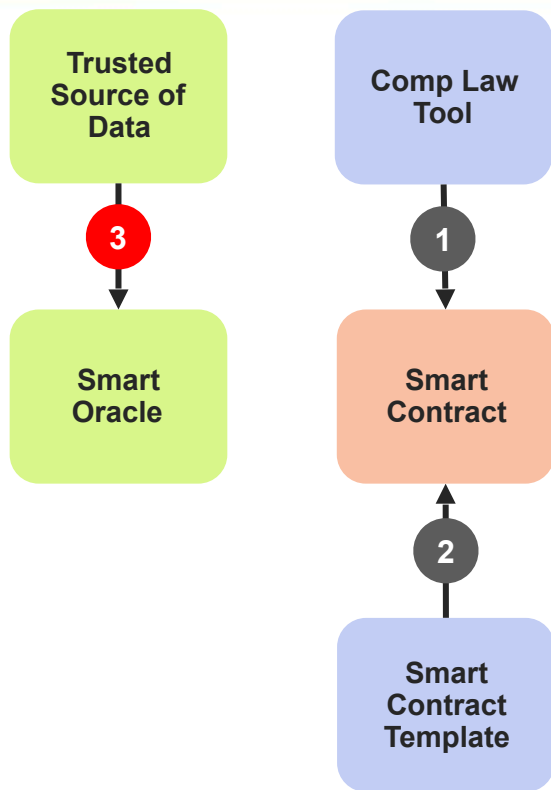
Notes

- We used a simple Word template to create the simulated legal document, but have not yet applied a computational law tool. This aspect of the demo remains to be demonstrated successfully.
- There is a complex but definable relationship between a legal contract and a smart contract, described in the proceedings of the three smart contract template summits held during the past year by the R3 fintech consortium (see appendix 1).
- Domain-specific languages for legal contract evaluation and smart contract generation are being developed by R3, Stanford University and other organizations (see appendix 1) but were not available as of July 2017.
- In the use case of an SLA, the legal contract and smart contract must include rules that define an SLA violation. Because we are using the TM Forum SLA Management API, we simply adopted its rule paradigm of metrics, units, reference values, operators, tolerances and consequences for that purpose.

*External
Systems*

*Adjacent
Tools*

*Ethereum
Blockchain*



*External
Systems*

*Adjacent
Tools*

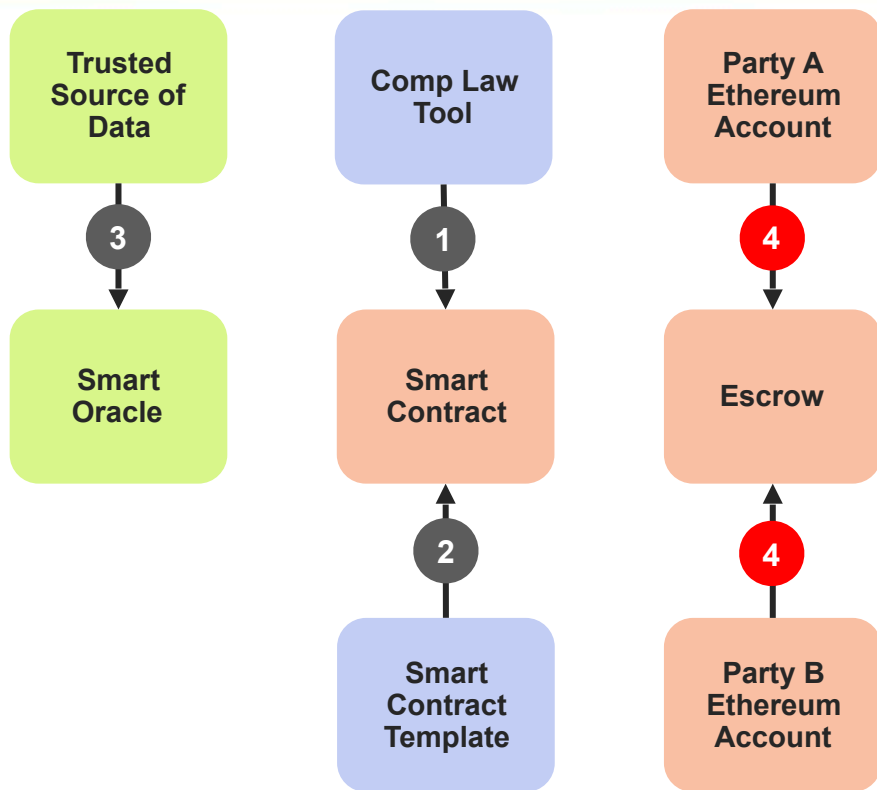
*Ethereum
Blockchain*

3. The SLA covers the availability of a Web site as reported by a smart oracle. Pingdom, a widely trusted company for this purpose, monitors the Web site. The smart oracle uses Pingdom as a trusted source of data.

Notes

- Pingdom is one of several site monitors available in both free and paid plans (see appendix 1). We used it because it offers both push and pull notification, allowing experimentation with several styles of smart oracle / smart contract interaction.
- Ethereum smart contracts need smart oracles if they are to make use of external data because, as a fundamental design choice, they cannot make API calls outside the blockchain, and a typical trusted data source is not able to make ABI (application binary interface) calls. Smart oracles bridge that gap¹, and may also bridge between multiple token currencies.
- The choice between push notification (trusted data source sends data asynchronously and/or periodically), pull notification (trusted data source provides continuous access to data), or a hybrid, affects the details of setup and operation of the smart oracle and the smart contract. The demo at Action Week 2017 Lisbon used pull notification. The current demo uses push notification.
- Deployed Ethereum smart contracts cannot be modified, although somewhat complex arrangements can be made to componentize a contract into replaceable parts, also affecting setup and operation.

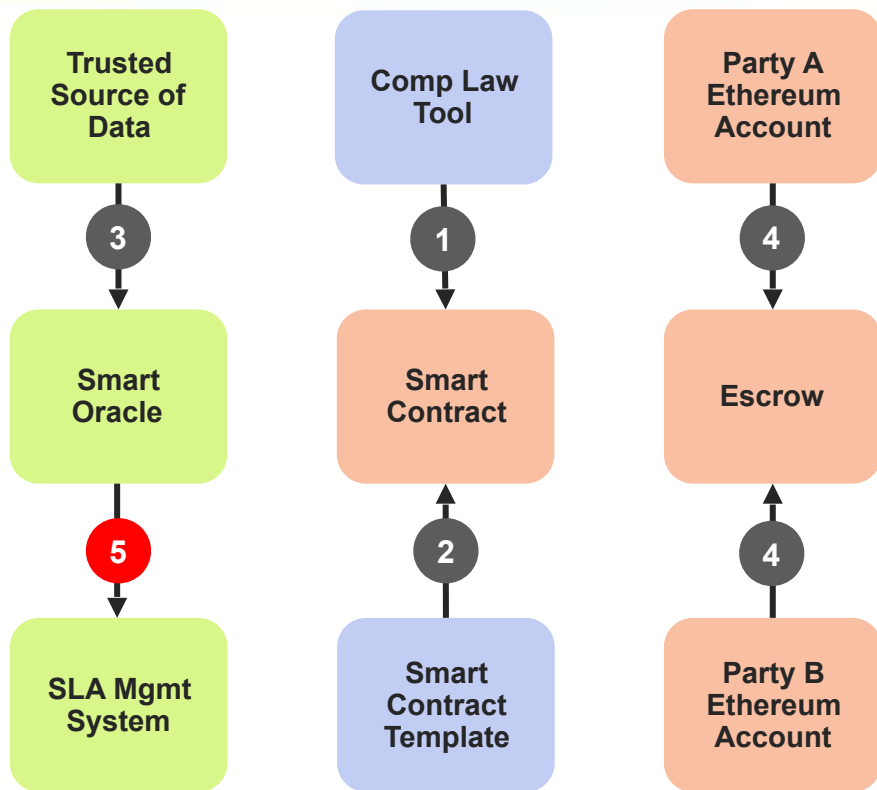
¹ ironically adding trusted data sources to a trustless environment



4. The two parties to the SLA each place in escrow an amount of ether sufficient for worst-case fulfillment of their contractual obligation. They then initiate the smart oracle and smart contract.

Notes:

- One party is compensated for successful performance (the Web site is up) and the other is compensated for SLA violation (the Web site is down).
- We used an escrow model because it was easy to implement in a private test mode blockchain, and allows immediate and final settlement (it might take 20-30 seconds in the public Ethereum chain). A real-world scenario would need to use the public Ethereum blockchain as the simplest way to implement an escrow of Ether.
- The escrow model requires all parties to use the same token currency, or to add oracles or inter-ledger protocols to handle currency conversion.
- The previous demo at Action Week 2017 Lisbon used direct payments across two token currencies, facilitated by a smart oracle (one party used Bitcoin and the other used Ether).



5. The smart oracle reports the smart contract initiation to the SLA management system using the TM Forum SLA Management API.

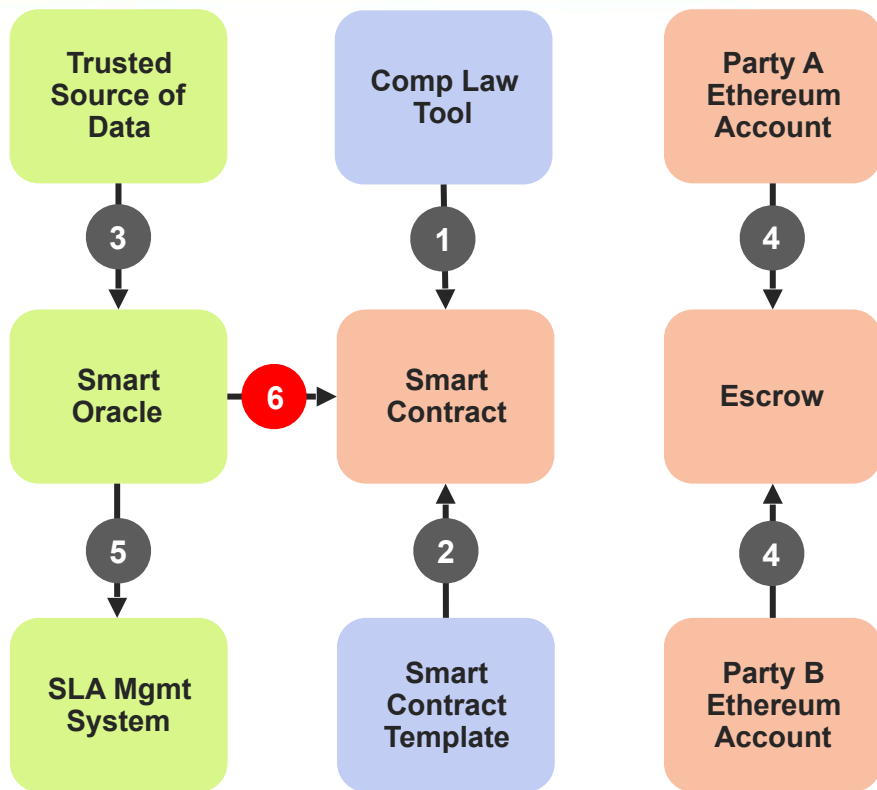
Notes:

- This step emphasizes the significant role of the smart oracle in bridging an information gap between systems. The smart contract is unable to communicate with the SLA management system, and the SLA management system (in this simple implementation at least) is passive with respect to the creation of SLAs and SLA violations.
- In a real-world scenario each party might have its own SLA management system, so the oracle would have to update all of them, introducing opportunities for desynchronization if one system was temporarily unavailable. That could be resolved by recording a “master” copy of all relevant data in the blockchain itself, but that also opens up a number of questions about system roles.
- We deliberately tailored the simulated legal agreement and smart contract to use the data elements already defined in the SLA Management API, reasoning that they reflect existing best practice as well as TM Forum technical guidelines.
- In a real-world scenario the amount and complexity of data defining an SLA and its violations would be considerably greater than in our simple demo scenario. Those factors might require breaking down the violation rules across multiple smart contracts or subcontracts. They might also require expanding the data content of the API itself.

*External
Systems*

*Adjacent
Tools*

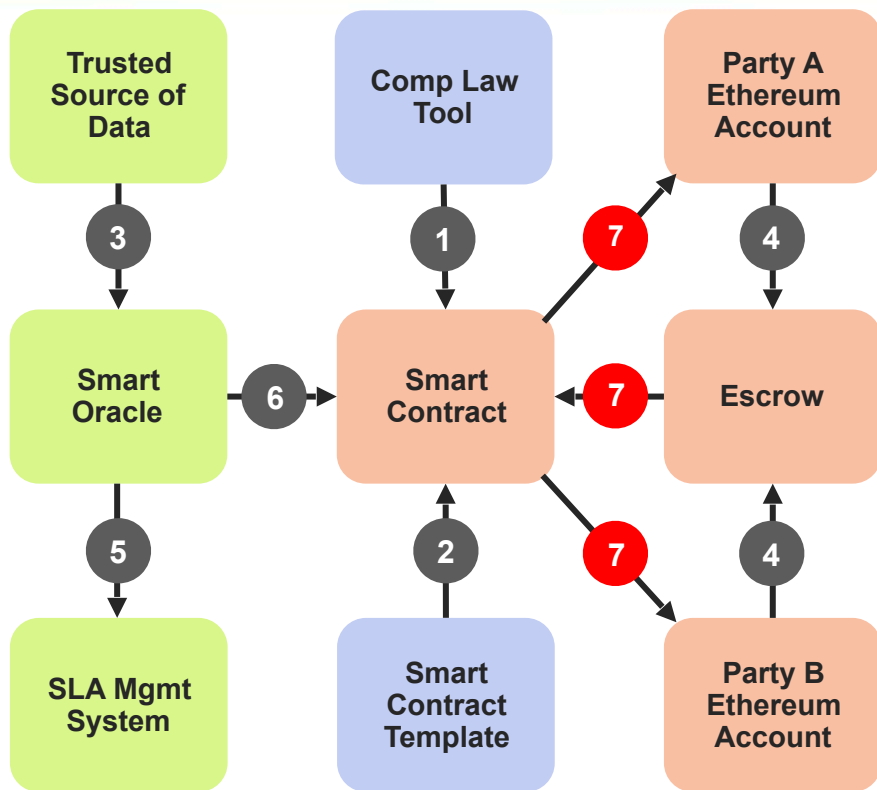
*Ethereum
Blockchain*



6. The smart oracle sends the smart contract a message at previously agreed intervals. The message contains the current up/down state of the Web site.

Notes:

- In the previous demo at Action Week 2017 Lisbon we found that the timing of notifications from the trusted data source, the compilation and dissemination of information by the smart oracle, and the latency of the smart contract's blockchain introduced the possibility unexpected results. Given the same starting conditions, three runs of the demo on three nights produced three different results.
- Part of the problem in Lisbon was that we were attempting to calculate a running average of uptime in the smart contract itself. In this demo we took a much simpler approach based solely on the uptime during a short interval.
- Each iteration of an Ethereum smart contract has an inherent cost, so that there is a strong incentive to minimize the frequency of iterations. On the other hand, the need to support micropayments requires more frequent iterations. We were not exposed to real per-iteration costs in our test mode blockchain, but this could be an important tradeoff in a real-world scenario.



7. The smart contract makes incremental micropayments from the escrow to one of the parties at each predefined interval, based on the reported state of the Web site.

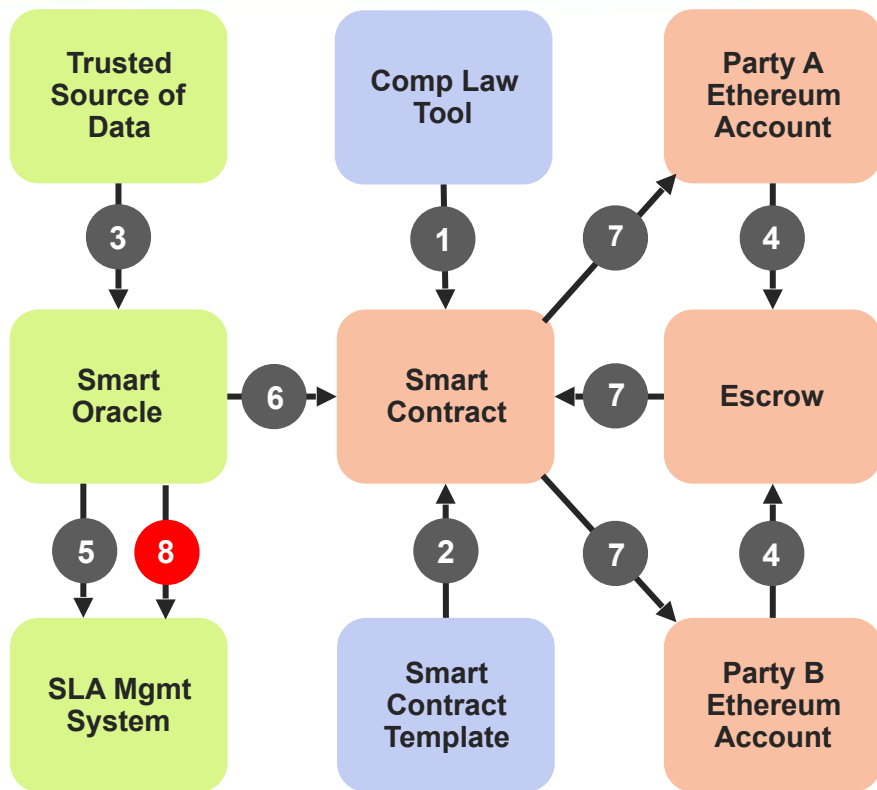
Notes:

- The use of escrow facilitates one of the objectives of the demo, which was to make frequent, small settlements using token micropayments.
- Settlements could also be made by direct transactions, but that approach would introduce the additional risk that a token wallet might at some point contain insufficient funds.
- In a real-world scenario, the amount placed in escrow by the parties would likely be asymmetrical, depending on the relative amounts of compensation for performance versus compensation for SLA violation.
- The current demo does not address the additional complexity of multi-party (three or more parties) SLAs and related violations.

External
Systems

Adjacent
Tools

Ethereum
Blockchain



8. The smart oracle reports violations to the SLA management system using the TM Forum SLA Management API.

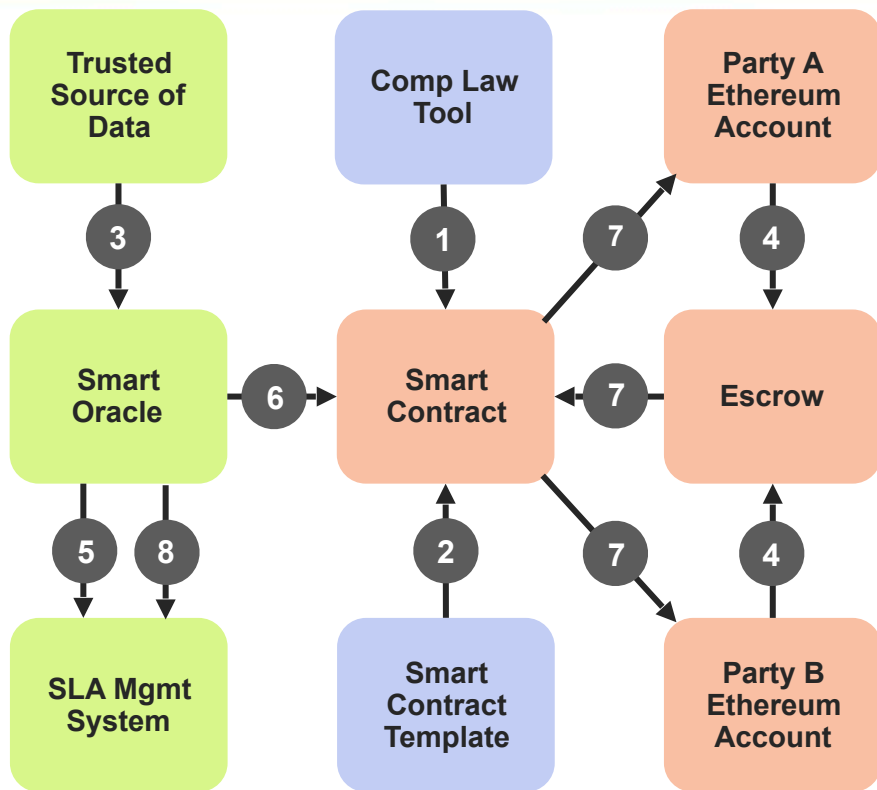
Notes:

- Again the smart oracle is bridging an information gap between systems. As mentioned earlier, a real-world scenario might require recording violation data in the blockchain itself as well as in the SLA management system(s). However, storing that data in the public Ethereum chain would involve additional cost and latency. For those reasons and others, it might be preferable to use a private blockchain for that purpose, which would also allow the use of a different (non-Ethereum) blockchain if desired.
- The strategy and tradeoffs related to the quantity and frequency of data collection, and the resulting maximum frequency with which violations can be reported, are affected by multiple factors, several of which were illustrated in previous steps. Notable in this step, which along with steps 6 and 7 represent the long-term operation of the smart contract, is the overlapping yet complementary scope and functionality of the smart contract and the SLA management system.
- An area for future investigation might be the use of a blockchain as the storage medium for the SLA management system, without necessarily requiring any change to the API itself. That approach might obviate the need for storage in the public blockchain, while retaining its advantages.

External
Systems

Adjacent
Tools

Ethereum
Blockchain



1. Legal staff use a template to create a contract representing a service level agreement (SLA), validate the contract using a computational law tool, and distribute it for electronic signature.
2. Technical staff use the validated electronic contract and a smart contract template to create a smart contract that will manage the execution of the previously defined SLA.
3. The SLA covers the availability of a Web site as reported by a smart oracle. Pingdom, a widely trusted company for this purpose, monitors the Web site. The smart oracle uses the Pingdom API as a trusted source of data.
4. The two parties to the SLA each place in escrow an amount of ether sufficient for worst-case fulfillment of their contractual obligation. They then initiate the smart oracle and smart contract.
5. The smart oracle reports the smart contract initiation to the SLA management system using the TM Forum SLA Management API.
6. The smart oracle sends the smart contract a message at previously agreed intervals. The message contains the current up/down state of the Web site.
7. The smart contract makes incremental micropayments from the escrow to one of the parties at each predefined interval, based on the reported state of the Web site.
8. The smart oracle reports violations to the SLA management system using the TM Forum SLA Management API.

- This demo used a private blockchain that could be accessed by (business) members of an ecosystem or consortium, rather than permissionless blockchains like the publicly accessible Bitcoin and Ethereum.
- In such a scenario, transaction cost and transaction time are far lower than with a public blockchain.
- However, that also meant that we were dealing with "play money" - a private blockchain is faster and cheaper, but has no direct connection to "real" token currency that can be exchanged on the open market for fiat currency like dollars or euros. So the actual SLA settlement would have had to map the "play money" transactions to "real money" outside the blockchain, thus losing some of the advantages of smart contracts that deal directly and securely in a cashable currency.
- There are several ways, all more complex than the original demo, to deal with the real money / play money problem while still preserving most of the advantages of a private blockchain, including the use of inter-ledger protocols, and the use of smart contracts in a private blockchain to trigger smart contracts in a public blockchain.
- We will demonstrate at least one of those techniques in an SLA management context at Action Week Vancouver.

- R3 smart contract template summit slides



- Contract definition and evaluation languages

- ❑ <http://clacklang.org/>

- ❑ <http://compk.stanford.edu/>

- List of free site monitors

- ❑ <https://topalternatives.com/testing-your-websites-speed-and-performance/>