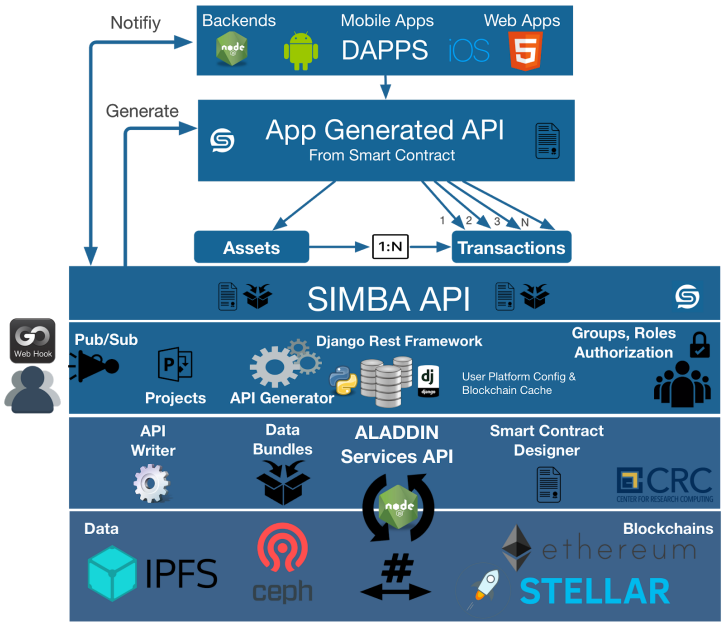
What is Simba?

A Blockchain is a distributed ledger in which an immutable, and therefore non-repudiable, record of transactions or events can be stored permanently and verifiably, without the need for a central authority. Blockchains allow digital information to be distributed, but not copied, meaning each individual transaction can only have one owner. This is enabled through the use of wallets, containing (multiple) special asymmetric cryptosystem keys that are used to sign each transaction, making all transactions identifiable and authentic. Blockchains that employ the use of Smart contracts (programmable pieces of code) can ensure certain transactions are only performed when specified conditions are met without the need for a human to manually intervene --- smart contracts have been described as “cryptographic ‘boxes’ that contain value and only unlock if certain conditions are met.”[1]. Smart contracts are also deployed and written as a transaction onto the Blockchain, which also makes them immutable and signed by the user that deployed them to ensure the authenticity of the code is not compromised.

From an application standpoint however, this underlying tooling is not enough to build production applications from. There are at least two other aspects that are needed: often, applications need to store multiple files, documents or datasets alongside their blockchain transactions, which are simply too large and expensive to store on-chain, hence off-chain support is needed; and private blockchain access control layers are needed to provide the ability to add authorization permissions to the on-chain and off-chain data that is shared, allowing a user to decide with whom they share the data and transactions with, and how.

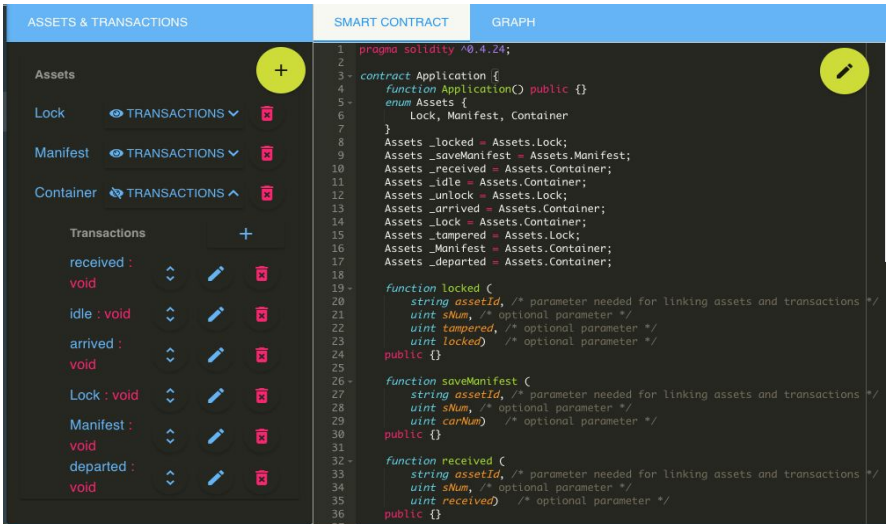
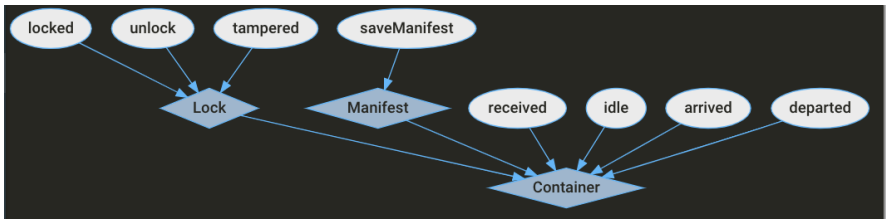
SimbaChain addresses these needed features and more, the architure of the system is shown to the right. It has a core underlying goal of providing tooling for developers to make it easy to build and deploy blockchain systems across multiple blockchains and data stores. It provides a design tool for generating a business process model for tracking assets, which auto-generates a smart contract and corresponding API for integration with external applications. The resulting API provides REST based access to smart contract methods that transact on the ledger and provides access control using groups for reading or



[1] https://github.com/ethereum/yellowpaper

writing to such resources. All data smart contract API endpoints allow multiple data files to be attached and stored in an off-chain data store, such as IPFS or Ceph. The diagram to the right expands on this to illustrates the core features of the system for each level of the development Blockchain stack.

	Blockchain	Simba
Access Control	Private - external tools, public - none	Permissioned Data & Blockchain in API
Wallet	Keys Sign Transactions	Fully Integrated Wallets (Browser/ APIs)
Data Store	External Data, Hashcode On-chain	Automated Off-chained Data
Smart Contract	Non Reputable Code & Automation	Write Smart Contacts With Zero Code
Blockchain	Non Reputable Transactions	App Interface to Multiple Blockchains

For the **Blockchain** layer, Simba provides a generic API to multiple Blockchain systems, thus the system does not have a dependency on a single Blockchain or DLT implementation. Currently, the platform supports Ethereum and Stellar but several more are on the roadmap. Furthermore, although SimbaChain is proprietary, it will soon release a code generation capability that auto generates source code that binds to the specific Blockchain system and Data Store the user has configured. This makes it possible for developers to export the system into an enterprise environment or to use SimbaChain to create open source APIs for public use.

Smart Contracts provide the interface, and business logic, to what is written on the Blockchain and what rules need to be satisfied for this write operation to take place. In SIMBA Chain, Smart Contracts are automatically generated from conceptual models that define the Assets and Transactions that transact on those Assets. Such models are specified using the Simba's Web App's UI shown on the right. A user simply uses the GUI to add Asset or Transactions along with their methods and parameters, and SimbaChain automatically generates the smart contract for the platform they select (in this case, Solidity code for Ethereum). It also generates a graph of the relationships for the model as shown just below. The resulting smart contract once deployed on