The Digital Asset Platform

Non-technical White Paper

Digital Asset

December, 2016

Executive Summary

This paper provides a high level overview of the architecture of the Digital Asset Platform, a common foundation on which financial services applications can be built. It provides the business rationale for our design decisions and is intended for a non-technical audience.

The Digital Asset Platform uses Distributed Ledger Technology to allow the mutualization of financial market infrastructure across distinct market participants. It does this while maintaining confidentiality and scalability, both vital for large, regulated markets. The DA Platform eliminates discrepancies between disparate but duplicative siloed data records, reducing the current errors, latency, risk, cost and capital requirements involved in processing financial transactions. Participants in the Platform share a single source of truth which provides continuous data integrity, any desired or mandated degree of transparency and the opportunity for rapid innovation.

We do this using Digital Asset Modeling Language (DAML); a powerful, intuitive, modular, distributed, privacy preserving domain specific language. This ensures consistent interpretation and application of business logic, and provides a real-time, auditable log of ordered evidences of events. These evidences are cryptographically linked to private trade data that is replicated selectively among only those parties entitled to view or interact with it. By combining a network-wide, replicated blockchain log and partially replicated reference data, each participant can create their subsection of the ledger with full confidence that it is consistent with that of other parties.

We provide a short historical context to the evolution of Distributed Ledger Technology and the requirements that have led us to our platform architecture, with a focus on confidentiality and scalability. We then break this architecture down into its primary components to describe the key business benefits. Next, we explain some of the roles that market entities play on the Distributed Ledger network, and how these roles may change over time to support incremental adoption. Finally, we describe how the DA Platform can be extended, either by building applications or by adding libraries defining market functionality.

Table of Contents

1.0 Introduction

1.1 Current Financial Infrastructure

Existing financial market infrastructure, which processes trillions of dollars of value daily, is necessarily highly resilient and reliable. Market infrastructure has evolved incrementally over years without consistent architectural design and is characterized by siloed data stores, maintained independently by each participant. The redundant storing of common information provides resilience, but gives rise to expensive and time-consuming reconciliation activities between siloed data stores as each market participant strives to ensure their books match those of their counterparties.

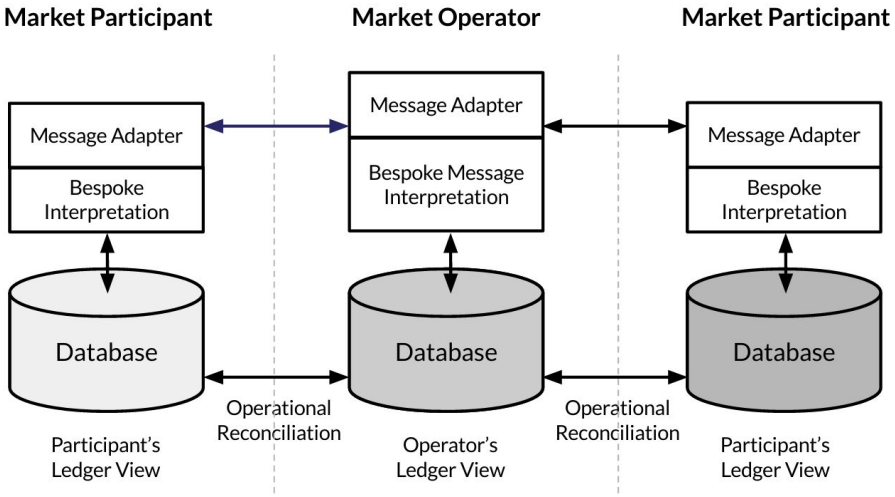

Figure 1. Today's siloed data causes messaging and reconciliation issues

Current infrastructure requires reconciliation because there are no other means of mutually agreeing on the status of important transactional data. This is accepted as a natural cost of multiple participants playing distinct roles in a market, but results in significant delays, operational cost and impact on capital.

Furthermore, the software that underpins market infrastructure has largely been in operation for decades, and is difficult and expensive to upgrade and adapt as both regulatory and market needs evolve. This has been highlighted in the aftermath of the global financial crisis as regulators demand greater market transparency and reporting, for which legacy systems were not designed.

Distributed Ledger Technology represents a generational opportunity to tackle the challenges of traditional market infrastructure and significantly reduces the need for reconciliation, thereby reducing errors, delays, risk, and capital inefficiencies. It enables more flexible market structures, increases the rate of innovation and allows for far more streamlined reporting.

1.2 Distributed Ledger Technologies

Distributed Ledger Technology is a convenient collective term that encapsulates a number of components. These include the use of blockchains, public key infrastructure and cryptographic signing, hash functions, modeling and automation of business logic, consensus algorithms, and others. The particular components required vary according to the problem being solved, such that Distributed Ledger implementations may make use of some or all of these technologies in differing combinations.

A Distributed Ledger is a record of transactions or other data which exists across multiple distinct entities in a network. The ledger can be wholly replicated across participants, or segments can be partially replicated across a subset of participants. In either case, the integrity of the data is ensured in order to allow each entity to rely on its veracity and to know that data they are entitled to view is consistent with that viewed by others entitled to view the same data. This makes the Distributed Ledger a common, authoritative prime record — a single source of truth — to which multiple entities can refer and with which they can securely interact.

Distributed Ledger Technologies have expanded beyond mere transaction registries to include other forms of data and encoded business logic, sometimes referred to as "smart contracts." This means that not only does the technology synchronize the record of who owns what, but also provides a common workflow for processing that data, ensuring that the results of agreements are processed in the same, mutually agreed manner.

1.2.1 Bitcoin

Bitcoin was the first notable example of a Distributed Ledger. It is a protocol, a blockchain network, and a unit of account (the cryptocurrency "bitcoin"). It is a globally replicated ledger, where each full node in the network validates transactions and stores a copy of the entire history of the ledger. Miners, incentivized by the reward of bitcoin creation, provide security and validity for the network by partaking in a computationally intensive guessing game that, while not easily solved, is easily verified once solved. The difficulty of this guessing game is automatically adjusted based on the rate at which it is being solved by miners, to achieve a nearly steady duration between each solution.

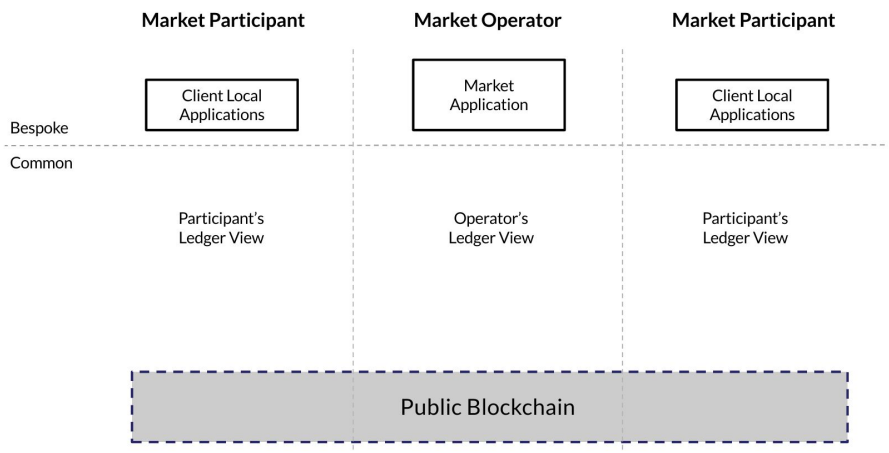

Figure 2. A common public blockchain: a shared, replicated ledger across all participants

Bitcoin's design decisions follow logically from its objective - to create a secure, public, peer-to-peer electronic cash system that avoids the need for intermediaries. It is a revolutionary and elegant solution to those unique and challenging requirements. However, different applications, such as wholesale securities clearing and settlement or derivatives processing call for a very different solution. Such markets differ materially in terms of transaction values, volumes, and throughput rates and also the highly regulated nature of both participants and activities. For such uses, the computational intensity of the Proof-of-Work consensus process used in Bitcoin makes it expensive due to energy consumption, and it is inherently too slow. Moreover, lack of confidentiality, scalability and settlement finality, and the pseudonymity of users and transaction processors make Bitcoin unsuitable and non-compliant for most regulated financial services applications.

1.2.2 Ethereum

Ethereum is currently the second largest blockchain network, inspired by Bitcoin but designed to improve upon its functionally limited ability to program business logic. Ethereum addressed this with expanded smart contract capabilities — software applications that are shared and run on all validating nodes of the Ethereum network.

Ethereum supports a number of smart contract languages which allow agreements to be written in code that can be executed automatically by the network. These self-enforcing agreements independently control and automate the exchange of escrowed value according to predetermined rules based on predefined inputs. This is a notable feature, as all smart contracts on Ethereum have to be executed by multiple participants in the network, including and especially those not party to the contract. Thus, any third party can not only view all transactions, but can also know the exact terms of those contracts.

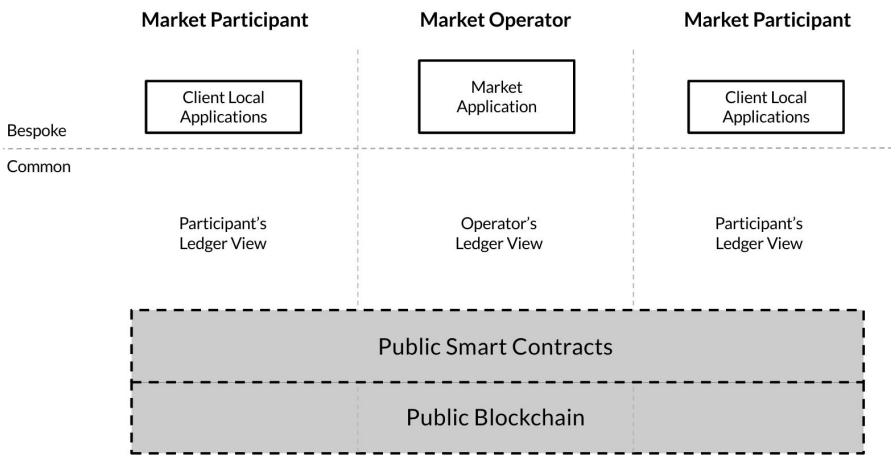

Figure 3. A public blockchain with censorship resistant smart contracts

As with Bitcoin, Ethereum is an ingenious solution designed to minimize the role of trust and enable actors unknown to each other to enter into software-enforced "contractual" agreements that are censorship resistant, meaning neither party, intermediary, nor even government can interfere with or prevent their execution. However, Ethereum smart contract languages are general purpose in nature and not designed with the safeguards and finitely predictable outcomes appropriate for financial market applications. One consequence of this is that they risk introducing potentially significant unintended consequences into self-executing contracts (as evidenced by the recent "DAO exploit"). This, combined with the lack of contractual privacy, means that Ethereum in its current evolution is also unsuitable for use in highly regulated financial services.

1.2.3 Other Distributed Ledgers

Some solutions attempt to address privacy by obfuscating the data associated with publicly visible transactions and identities. However, obfuscation leaves open the risk that identities can nonetheless be determined by post-facto analysis of the relationship between transactions. This approach is not suitable where regulations require customer confidentiality and data domicile restrictions apply.

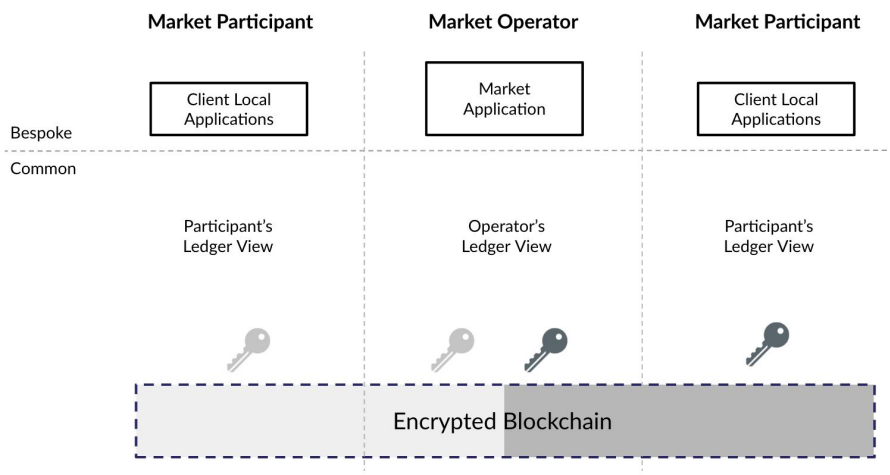

Figure 4. A globally replicated blockchain with encrypted data

Alternative confidentiality solutions use encryption to create what we refer to as "logically segregated" ledgers. In this case, transaction data is still replicated to all participants, but only those who are entitled to view the data can decrypt it. However, such solutions are vulnerable to the requirement of "forward secrecy" — ensuring that historical data remains confidential even in the event of a subsequent compromise of encryption, due for example to advances in quantum computing. Even assuming an impenetrable encryption algorithm, this approach is generally not compliant with data domicile restrictions as data is still stored on non-entitled entities' servers, even if in encrypted form.

Another evolving cryptographic approach to maintaining privacy uses Zero Knowledge Proofs (ZKP). In ZKP systems, the transaction contains a proof of its validity without containing any of the confidential data itself, which can be revealed separately. While promising, implementations are as yet immature and unproven in production environments.

Reflecting the requirements of both customers and their regulators, it is Digital Asset's position that confidential data should never be stored by a party not entitled to view that information, even if obfuscated or encrypted. As such, any potential solution designed for financial institutions must physically segregate confidential data. One solution requires an interconnected network of physically segregated, low-population ledgers.

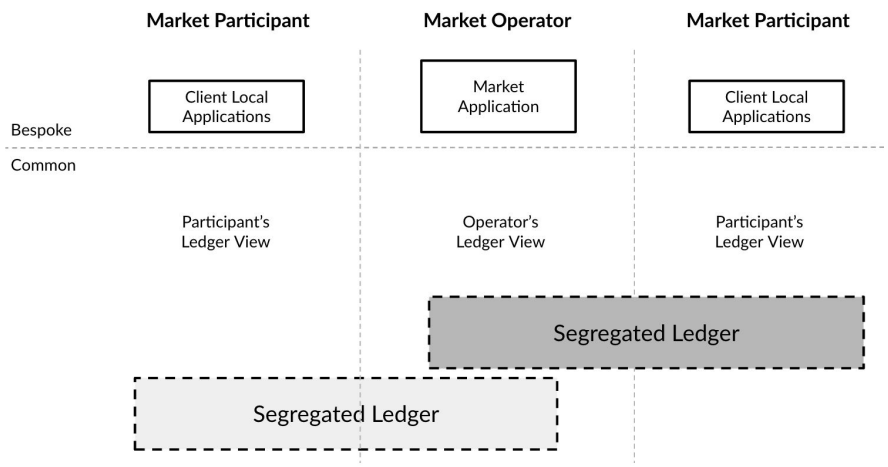

Figure 5. A Distributed Ledger with physically segregated ledgers

Such physically segregated ledgers achieve confidentiality, but introduce their own difficulties: A network of segregated ledgers lacks a global arbiter of the truth, and the system cannot guarantee network participants integrity of the complete set of relevant transactions. This approach proliferates independent bilateral and multilateral ledgers, leading to reconciliation issues similar to those that exist today.

1.3 Digital Asset Platform Solution

The Digital Asset Platform has been designed to maintain the same confidentiality guarantee as physically segregated ledgers but also to allow for the same data integrity assurances of typical blockchain solutions.

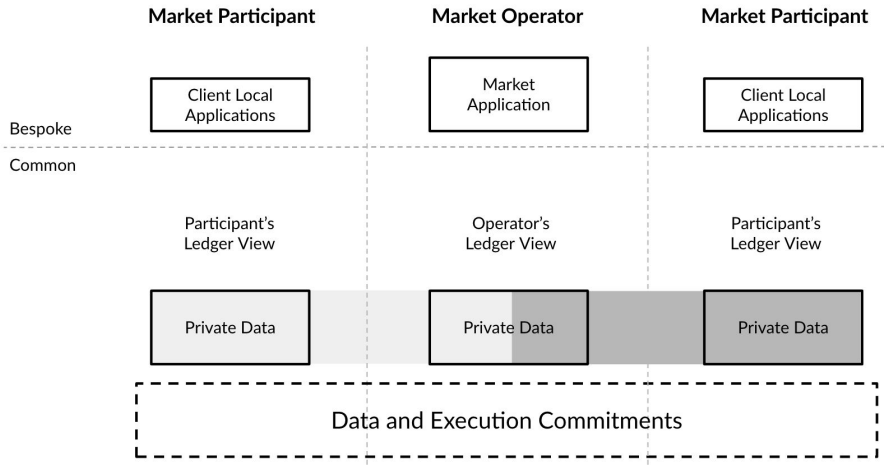

Figure 6. Synchronizing private data across a distributed network

This is achieved by the parties involved physically segregating and storing locally confidential contractual information, and sharing a globally replicated

log of only fingerprints, or "hashes", of the sensitive data and execution commitments. These hashes are one-way cryptographic functions which can be proven to accurately match a party's data but do not contain any information about the confidential data itself nor the parties involved.

2.0 Digital Asset Platform Architecture Overview

In its simplest form, a solution built with the DA Platform consists of three layers: the Application layer, the Business Logic layer, and the Distributed Ledger layer. Each layer has its own communication channels, as pictured below.

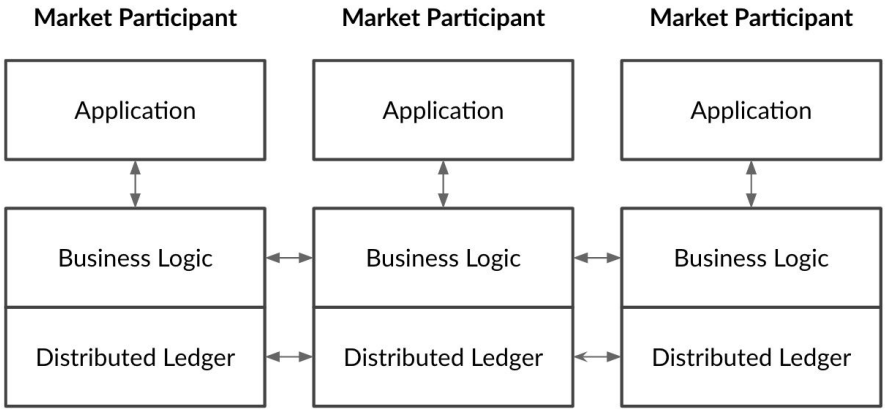

Figure 7. Simplified architectural layers and communications channels

The Business Logic layer and Distributed Ledger layer are processed by the DA Platform, and the Application layer consists of custom software interfacing with the DA Platform and other systems.

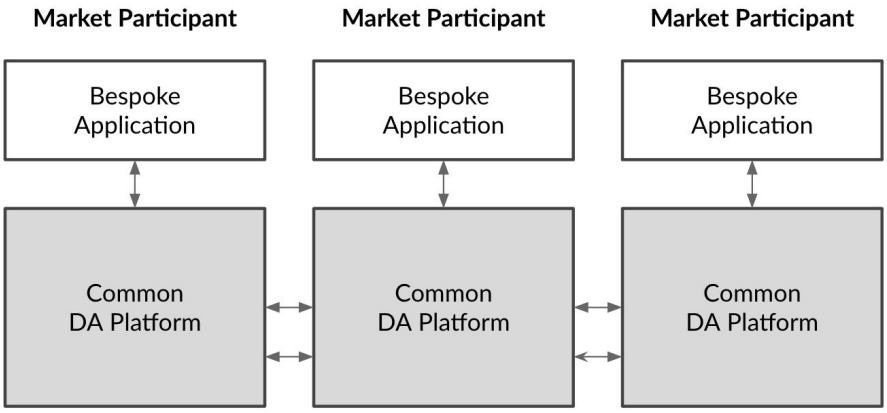

Figure 8. Applications are bespoke to the participant, but the Distributed Ledger Layer and Business Logic Layer of the DA Platform are common to all

2.1 Distributed Ledger Layer

The Distributed Ledger is a permissioned ledger, meaning it is a ledger accessible (for reading or for writing) only by known and pre-approved parties. This differs from a permissionless ledger, like Bitcoin and Ethereum, where anyone can read or write to the blockchain. The Distributed Ledger in the DA Platform is comprised of two subcomponents: the Global Synchronization Log ("GSL") and the Private Contract Store ("PCS") .

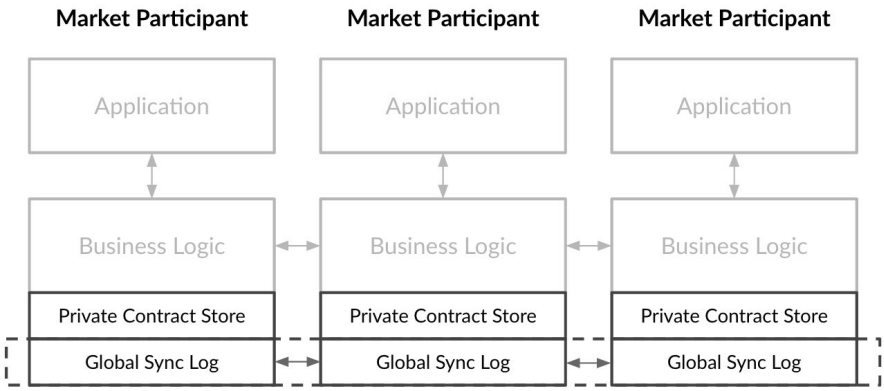

Figure 9. Including the two primary components of the Distributed Ledger Layer

2.1.1 Private Contract Store

Each Participant has its own PCS, which contains all validated contracts to which the participant is a party. In this context, the term "contracts" refers to business logic, including transaction parameters, rights and obligations, reflecting the encoded terms of legal agreements by which participants are bound.

The PCS is stored locally and only contains those contractual agreements that the participant is entitled to store and view. The PCS is a durable store of the codified contractual relations between parties. It does not process the executable business logic itself, which is performed at the Business Logic Layer.

It is important to note that since the PCS contains a historical record of all executable contracts (both active and inactive) pertaining to a participant, this segment of the Distributed Ledger cannot be constructed from the contents of the PCS alone. To do this, contracts within the PCS must be paired with corresponding active evidences in the GSL.

2.1.2 Global Synchronization Log

The GSL, as detailed in our previous white paper[1], provides the same integrity and transparency guarantees found in shared, replicated ledgers to a distributed network of physically segregated transaction data. The GSL further uses a blockchain as a privacy-preserving uniqueness service with a built-in notification mechanism.

The GSL is a log of commitments and notifications that guarantees the integrity and auditability of the distributed data stores to contract stakeholders. The GSL establishes a common and complete set of valid transactions that, when combined with the corresponding private contract data in the PCS, comprises the Distributed Ledger. The GSL is a communication layer designed to deliver network-wide integrity guarantees of transaction commitments and notifications.

The GSL serves three primary functions:

1. To serve as the arbiter of relative order between dependent transactions;
2. To ensure uniqueness of mutually exclusive events and maintain the state of the ledger data. This state is derived from the stream of transactions; and
3. To serve as an assured notification mechanism. Any stakeholder affected by a state change of data or contracts must be notified, or, more precisely, must have the assurance that it will be made aware that this state change has occurred.

2.2 Business Logic Layer

Business logic is primarily written in Digital Asset Modeling Language ("DAML") and is composed of two main segments: DAML Libraries, containing the business logic rules for distinct use cases or functions, and the DAML Execution Engine, for processing and verifying these rules.

[1] Digital Asset: *The Global Synchronization Log*
http://digitalasset.com/press/updates-to-open-source-community-work-and-whitepapers.html

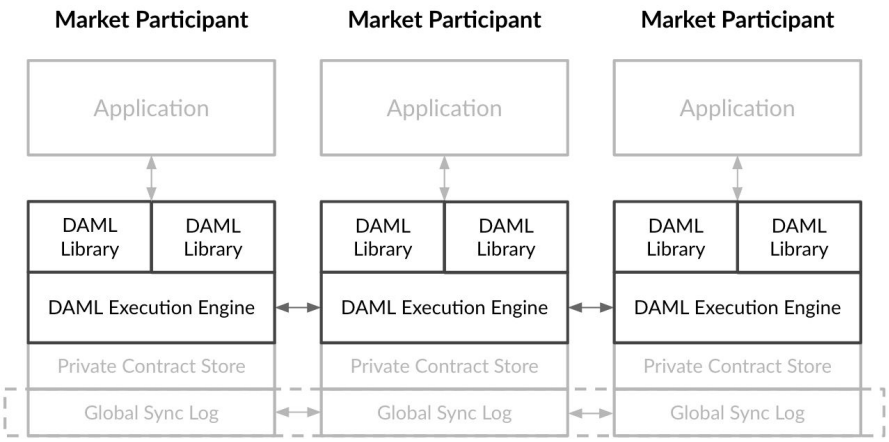

Figure 10. The two components of the Business Logic Layer

2.2.1 Digital Asset Modeling Language

DAML is the language in which logic and behavior for the DA Platform are written. It easily enables market functionality to be added to extend the platform's capabilities. DAML enforces the rules of the market through software and is used to model contractual rights and obligations in executable code within the boundaries of those market rules.

The promise of automation

Much of the current enthusiasm for Distributed Ledger Technology in financial services centers on the potential benefits that may be realized through combining a synchronized ledger between participants in a market with the automation of complex, multi-party workflows acting upon the ledger.

As introduced in the earlier section on Ethereum, the shared snippets of code used to automate these workflows are known as "smart contracts" and the variety of programming languages used for writing them as "smart contract languages".

The automation needs and constraints within financial markets

Transactional activity within modern regulated financial markets is built upon a shared hierarchical foundation; namely:

- A base layer of national legal systems or agreed international standards providing security of legal contract between parties and predictable mechanisms for dispute resolution.
- A layer above of agreed market rules (or rulebooks), published and enforced by central market operators or through bilateral trading agreements between organizations.
- A top layer of standard form legal agreements embodying common activities and trades within these markets.

Any proposed automation within financial markets must accurately reflect this hierarchy for benefits to be realized. Modern markets evolve continuously and a key requirement for any language used for automation is that it must be readily adaptive and facilitate rapid development of new functionality.

Smart contract languages each make key design choices that significantly impact their suitability for highly regulated financial markets. When designing DAML, we did so with several overarching goals: legal certainty, distributed execution, privacy, analyzability and ease of use.

Legal certainty

A natural tension point, evidenced in the term "smart contract" itself, is the relationship between the intent of the parties to an agreement and the code implemented to automate it.

Some proponents of smart contracts have argued for the position that "the code is the agreement". This is to say that smart contract code is immutable and that any outcome of the code itself is necessarily what the parties intended. It follows that there should be no recourse to dispute resolution through existing legal systems, centuries of common law legal precedent, nor reference to standard legal prose contract terms (even when an outcome results from a clear bug in the smart contract code).

Given the requirements of financial markets noted above, Digital Asset takes the view that code used for automation must remain subservient to legal systems and dispute resolution, market rules and commonly understood legal prose. Consequently, we designed DAML explicitly to enable the implementation of this hierarchy — delivering the benefit of automated workflows, while ensuring parties continue to have the certainty afforded to them today by the common legal foundation they all share.

Building a framework of market rules with DAML is important because it provides certainty to all participants of the allowable actions in any given scenario and it does so without compromising confidentiality.

DAML is also used to capture the terms and conditions of individual agreements between parties associated with transactional activities. DAML contracts provide absolute certainty of obligations between parties at all times throughout the asset lifecycle. This dramatically simplifies complex actions like buyer protection processes and significantly reduces the risks associated with participant failure.

Distributed Execution

The DA Platform is a use-case agnostic, distributed execution environment capable of processing any financial services application. By having a clear separation between platform capabilities and modeling functionality, software deployments of the DA Platform are identical across all participants and markets. This makes it significantly easier to maintain, support and extend functionality.

Just as the Distributed Ledger allows institutions to replace independent data stores with a shared ledger, DAML allows those institutions to agree on all possible updates to the state of that ledger with common workflows. Throughout the lifecycle of a contract, every stakeholder can view and execute choices in the contract, allowing each entity independently to verify updates to the ledger and automate business processes across institutions.

Developers using DAML do not expressly need to code for a distributed execution environment because the language has been designed for this purpose. For example, DAML assumes no centralized maintenance of state and has no mutable state, allowing for efficient and concurrent processing and analysis.. It has no locking mechanisms which would adversely impact performance and it does not assume the ability to query data of which one is not a stakeholder.

Privacy

Physically segregating data in each participant's PCS solves the challenge of maintaining privacy in a distributed system, but only if it can be determined which data should be shared with them in the first place. Interpreting the logic of which parties are entitled to view which data is typically a manual or error-prone process. DAML automatically identifies all stakeholders to complex financial workflows, including those that may be affected in the future. Because DAML can identify affected parties, it readily facilitates notification on a need-to-know only basis.

Market-wide rules, encoded in DAML, are shared with all market participants, but the actions taken by participants are private. Actions taken are guaranteed to adhere to the market rules. For example, the rules by which valid netting can occur are shared as DAML code, but the engine that calculates this netting, and hence the algorithm by which it is performed, remains private.

Analyzability

At a high level, programming languages can be characterized along a spectrum from general and complex to restricted and simple.

Highly general, or Turing Complete, languages (e.g., Java or Ethereum's Solidity) empower developers to leverage rich toolsets and solve any computable problem. However, coded solutions quickly become complex and difficult to interpret — even for experts. Developing new functionality, identifying, isolating and correcting bugs, and predicting all possible business outcomes for participants can be extremely challenging. Such languages introduce significant risks if used for automation in a financial markets context.

When automating financial markets with smart contracts, the standards for quality assurance must be very high. When designing DAML and its tooling, we therefore drew upon the rich knowledge of the formal methods and theorem-proving community to create a domain specific language for financial markets. DAML offers an intentionally restricted toolset tailored specifically for unambiguously specifying the terms of financial agreements within a common legal framework. Deliberately not Turing Complete, DAML's design enables three critical features:

1. Predictable behavior of the contract code;
2. Finite analyzability of all possible outcomes for all participants to codified agreements; and
3. A high degree of readability to both developers and business people.

The property of analyzability allows participants to pre-authorize future choices or workflows with full confidence of what they are committing to. A subtle consequence of this, unlike other implementations, it is not required for every affected party to a transaction to approve each ledger update or acknowledge that they were notified, unless their consent is actually required. This is not merely convenient; it ensures that if one party fails to approve a ledger update due to, for example, system outage, there is no halt of a business process. This avoids interdependency of one institution's service on another's availability. It also removes the possibility of neglecting to authorize obligations in order to gain a commercial advantage. For example, a party owning an option must be able to exercise without needing the authorization of the option seller at the time of exercise.

These features greatly broaden the appeal and opportunity of DAML for automation across financial markets.

Ease of Use

DAML offers the readability of a functional programming language, and incorporates integrated safeguards in the development environment. This gives developers instant feedback to confirm the correct interpretation of their intent and so it is faster, easier, and safer to develop financial applications in DAML than in general purpose languages. The language itself also has built in safeguards, for example, a contract cannot be written that obligates a party without that party's explicit authorization.

To enable rapid development, DAML has also been designed to be modular and allow reuse of common building blocks across use-cases. Each contract can reference other contracts and contract templates can be packaged into reusable collections, called DAML Libraries.

2.2.2 DAML Libraries

DAML Libraries ("Libraries") are collections of reusable templates for use-case-specific business logic and workflows. Examples include templates that define the characteristics of an asset class (e.g., equities, FX, or fixed income), a post-trade process (e.g., clearing and settlement), or a functional grouping (e.g., identity management). Libraries can be loaded into the DA Platform and contracts can be created from their templates for use across multiple parties. Each DAML Library automatically exposes the relevant Application Programming Interface ("API") for its functionality through the Platform API. The DA Platform is designed to be deployed across a wide variety of financial services applications and is capable of loading and executing multiple DAML Libraries in its Execution Engine.

2.2.3 DAML Execution Engine

The DAML Execution Engine ("DAMLe") combines supplied parameters and market templates to create the corresponding contracts, thereby converting actions triggered at the Application layer into events at the Distributed Ledger layer. Every Network Participant operates a DAMLe and is able independently to re-execute and verify the contracts committed to the ledger. This ensures that the results of a command's interpretation are consistent with the results of any other DAMLe executing or validating the same choice.

2.3 Application Layer

Applications are participant- or operator-specific configurations and integrations to existing systems, such as reporting, invoicing or netting services. Applications may be developed for use on the DA Platform by third parties. Applications enable users of the solutions to develop new services quickly and easily by leveraging the power of the DA Platform through the Platform API.

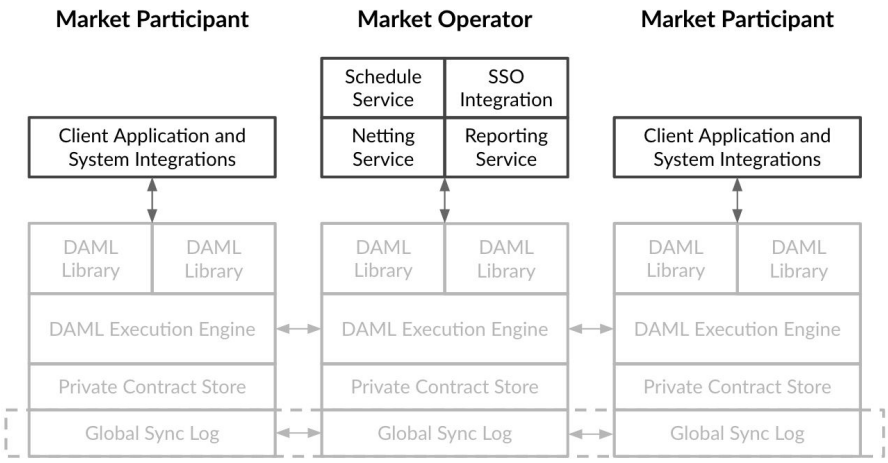

Figure 11. Example services of a Market Operator Application

It is important to note that Applications are distinct from Libraries. All participants in a market have identical copies of the Libraries but do not all use identical Applications.

3.0 Roles

In any Digital Asset deployment, there are two primary roles: that of Operator and Participant. An Operator has network-wide responsibilities such as maintaining the GSL, creating and distributing the digital rulebook of its domain, and validating a superset of data in the network. A Participant is an entity that has been granted permission to join the network by the Operator(s), and is given the opportunity to control and maintain its own local instance on the larger network. This entails reading and validating all transactions it is a party to, locally storing both private and globally replicated information, and cryptographically authorizing its transactions and updates to the ledger.

Either of these roles have the option to delegate operations to a third party participating in the network. Granular levels of delegation exist — a Participant may choose to operate directly as an authorizing party on some actions in the system while contracting with third parties to perform others. Doing so allows for a flexible migration of Participants into the network and tiered levels of control and disclosure.

Any distributed system must allow for institutions to play multiple roles. To allow roles to change over time, and, most importantly, to allow for the possibility that not all market participants will adopt the technology at the same time, each institution's role in the DA Platform is easily migratable and flexible, and entities electing different roles may co-exist. A subtle consequence of this flexibility is that the often-cited "network effect" required for the successful adoption of DLT is materially mitigated.

3.1 Network Participants

Network Participants are entities that have chosen to manage their own Digital Asset Platform Instance. Network Participant deployments are directly connected to a shared GSL. Any network that has two or more Network Participants constitutes a distributed network.

3.1.1 Actions

Each Network Participant can perform one or more of the following actions, depending on their role in the network:

Distributed Reading

Reading is the action of monitoring and receiving information from a ledger and re-executing the logic that produced the events in order to interpret and validate the information. Participants may be entitled to view all information (in the case of a regulator or market operator), or view partial information that pertains to them (in the case of a market participant). All participants perform reading and verify the integrity of the ledger through re-executing the DAML commands they are party to.

Distributed Signing

Signing is the action of authenticating and authorizing transactions using private keys, requiring participants to maintain their own key security. In order to be a party that directly signs their own transactions, a participant must also perform the action of reading to verify the transaction or agreement they are authorizing.

Distributed Writing

Writing is the action of committing evidences of data found in the PCS to the GSL. In scenarios with multiple distrusting Operators, a BFT Consensus Algorithm is required to reach a consistent state of the ledger for all parties. Consensus Algorithms remain an ongoing area of research and impact both throughput capacity and scalability. For centralized market structures, such as those with a CSD or CCP, distributed writing is not necessary to achieve the benefits of a Distributed Ledger. Even for bilateral or OTC market structures, it is likely that the appointment of a trusted market utility as a transaction validator may be a less risky and more performant alternative to a multi-Operator solution in the near term. Transitioning from a currently centralized infrastructure to a multi-Operator network relying upon a

Consensus Algorithm potentially expands vulnerability by increasing the opportunities for network exploitation and would likely be more challenging for regulators to approve in the near term.

3.1.2 Roles

Network Participants can act in one of two roles:

Operators

Operators codify and distribute the rules of the market. Only Operators can participate in Writing and there may be one or many Operators in the same network. Each Operator can run multiple Write Instances for added resiliency.

The most obvious example of a Operator is a centralized market infrastructure provider that is responsible for processing transactions. In this scenario, they may be the only Operator in the network, at least initially. In a single Operator configuration, Network Participants cannot prevent fraud or error on the part of the Operator, but they can detect it independently as they can authorize and authenticate their own transactions and validate the ledger. This affords them the opportunity to rely on the Distributed Ledger for their own internal books and records, and hence to eliminate reconciliation requirements against other Participants.

For markets in which there is no existing central entity today, some participants in the market can adopt the role of Operators and partake in Distributed Writing. Unless relying on legal constructs or trust, networks with multiple mutually distrusting Operators must rely upon a BFT Consensus Algorithm to ensure resilience against malicious actors. These complex algorithms allow Operators to come to agreement over the validity and order of transactions that are committed to the ledger. This prevents a faulty, malicious, or compromised Operator from censoring the network or committing invalid transactions.

Participants

Participants partake in Distributed Reading and Distributed Signing. Participants can see and validate their transactions evidenced on the GSL by verifying both the transaction outputs and their own signatures.

In this manner, Participants act as more than just passive consumers of data from Operators because they play the role of independently auditing and validating the integrity of the ledger. Any data alleged to pertain to them, but not already stored locally, can be requested directly from an Operator or other transaction stakeholders. They can then independently validate that the data is

correct, creating a sophisticated, self-validating, yet privacy-preserving Distributed Ledger network.

There is a specialized type of Participant whose role it is to verify specific events and notifications in the network. This is an optional role that can be performed by existing neutral institutions or regulators to ensure the network is functioning as expected and to monitor certain actions.

3.2 Indirect Participants

We recognize that the adoption of DLT will likely not happen across all entities in a market at the same time. In fact, some entities for whom the benefits of independent validation are outweighed by the cost of operational change, may decide never to become Participants. The DA Platform allows entities to continue to interface with the market in the same way as they do today by being Indirect Participants, which interact through existing messaging protocols without reading a GSL, maintaining a PCS or maintaining their own private keys. They may opt to become a Network Participant at a later stage, allowing for a phased-in deployment that does not require all market participants to join the network on day one.

For Indirect Participants to engage in the market, they must delegate effectuation of their actions to a Network Participant. This may be directly with an Operator, as they do today with a CCP, or through a Network Participant that offers market access as a service (e.g., a post-trade service provider or dealer).

3.3 Trade Flow Example

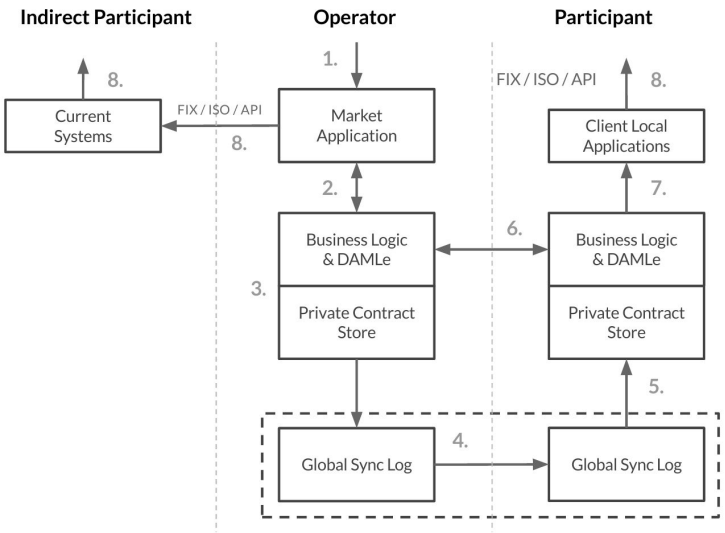

Figure 12. Example showing the flow of data between a Operator, a Network Participant and an Indirect Participant

The example in Figure 12 shows a post-trade process between an Indirect Participant (operating using existing messaging protocols only) and a Network Participant (operating a full instance of the DA Platform software) through an Operator, in this example an exchange.

1. An instructional message is sent to the Market Application from an execution venue.
2. A DAML command is sent to the DAML Execution Engine of the Operator.
3. The Operator then processes the command, instantiates the relevant contracts, persists them to its Private Contract Store, and inserts a hashed evidence of the transaction onto the Global Synchronization Log.
4. This evidence is then broadcast on the GSL to all Network Participants.
5. The Participant(s) affected by this event is(are) privately notified.
6. The Participant then requests receipt of the relevant contracts from the Operator (or any other party to the trade), verifies that the contracts are correct against the hash in the GSL, executes the contents of the contracts in its DAMLe to validate the contract business logic independently, and persists the contracts to its PCS.
7. A DAML Event is sent to the Participant's local Application.
8. The Participant is then notified by an ISO or FIX message or through the API. The Indirect Participant is also notified in the same way by the Operator.

4.0 Network Topology and Deployment Options

The DA Platform architecture has been designed to be operated in multiple deployment configurations. Here we provide three illustrative examples of different deployments or rollout phases, however there could be more or less depending on market requirements.

4.1 Fully Centralized Solution

The DA Platform can be implemented as a centralized solution, replacing existing functionality but without requiring Distributed Reading, Signing or Writing. As the first step in an adoption of a Distributed Ledger solution, this requires only the Operator to adopt the technology, with Indirect Participants communicating via messaging at the Application layer, as they do today. This benefits the Operator by replacing aging, inflexible and non-transparent systems, readies the Operator's infrastructure for future adoption of a distributed system, imposes minimal impact on customers, but offers none of the efficiency benefits of independent ledger validation to the Operator's customers.

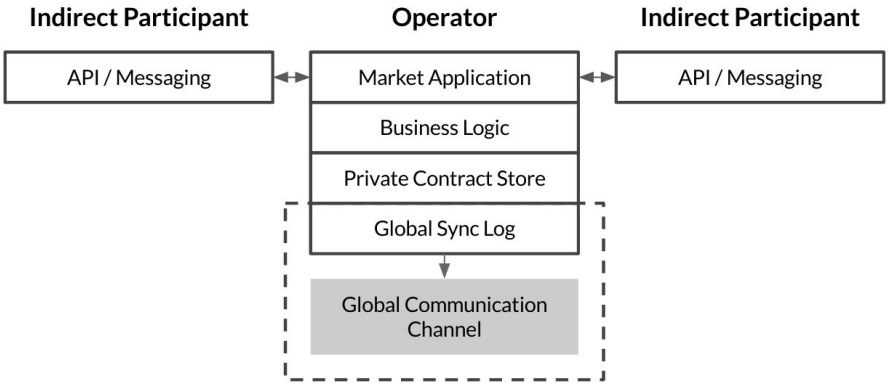

Figure 13. Example fully centralized network topology

In this simplified example there are one Operator and two Indirect Participants. The Operator is the only writer to the GSL and has been delegated the right to sign all transactions on behalf of all participants. There are no Distributed Readers, although there does remain the option to permit a regulator read or audit rights.

The Operator and regulators benefit from the efficiencies of encoding business logic in DAML and the way data is structured to record rights and obligations. Market transparency is greatly increased, potential outcomes are simpler to

analyze and predict, and scenarios can be simulated. Importantly though, reconciliation is not eliminated: Indirect Participants must continue to operate their own books and records and reconcile as they do today

4.2 Distributed Ledger Solution with Multiple Untrusting Participants

A Distributed Ledger solution involves two or more Network Participants running instances of the Digital Asset Platform. These parties are able to authorize transactions cryptographically, read the GSL to detect the absence of transactions that they expect to see, and re-execute business logic using DAMLe to validate transactions that affect them. Consequently, they can rely on the Distributed Ledger as the authoritative, independently validated record, and they can start to reduce external reconciliation efforts accordingly.

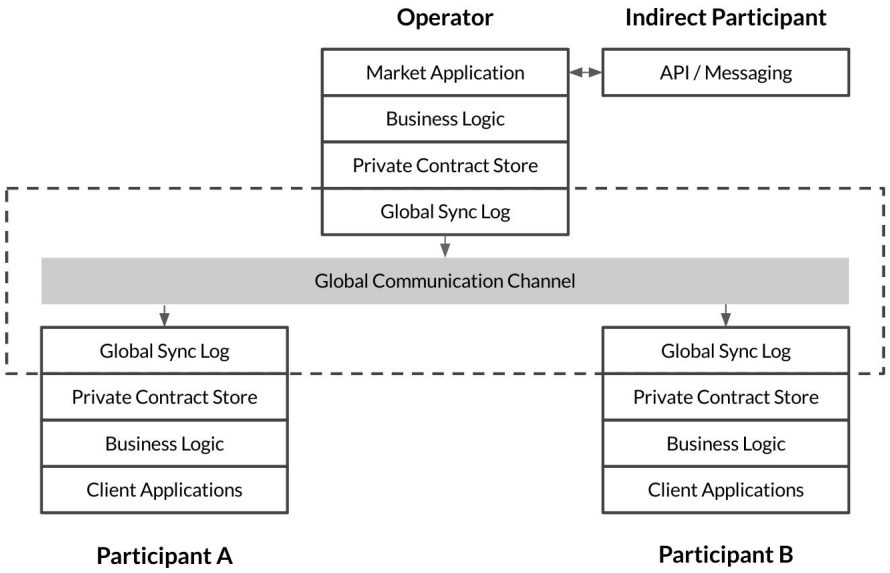

Figure 14. Example network topology with single Operator

In the example above, two entities have been permissioned to be Participants, each receiving identical copies of the GSL from the Operator and running their own Platform Instance. For simplicity, the diagram does not show communications at the Business Logic layer, which are across a direct, private channel between each Participant and the Operator. As in the Fully Centralized Solution, entities can remain as Indirect Participants and co-exist with Participants if they do not wish to transition to the new technology.

In a scenario in which there is already a central market infrastructure provider, this deployment option makes for an ideal second rollout phase, preserving the

roles of each entity in the market and allowing parties to adopt the technology on differing timelines.

For simplicity, Figure 14 shows each entity running a single Platform Instance. In reality, each Participant can run multiple Platform Instances for resilience, higher performance through parallelization, and disaster recovery. For example, an Operator will likely run multiple Operator Instances, in the same manner in which they operate multiple servers in a traditional internally distributed system or database today.

Each of the Operator Instances administered by an Operator must come to agreement on the sequence of events that will be written to the GSL and so must use a Fault Tolerant Consensus Algorithm. Note that this is not what is usually referred to as "consensus" in the Distributed Ledger industry, which generally uses that term as shorthand for Byzantine Fault Tolerant Consensus ("BFT" Consensus). BFT Consensus is required to ensure network resilience (up to a threshold) in the presence of multiple untrusting Operators of the ledger, and is both technically more challenging and more detrimental to performance.

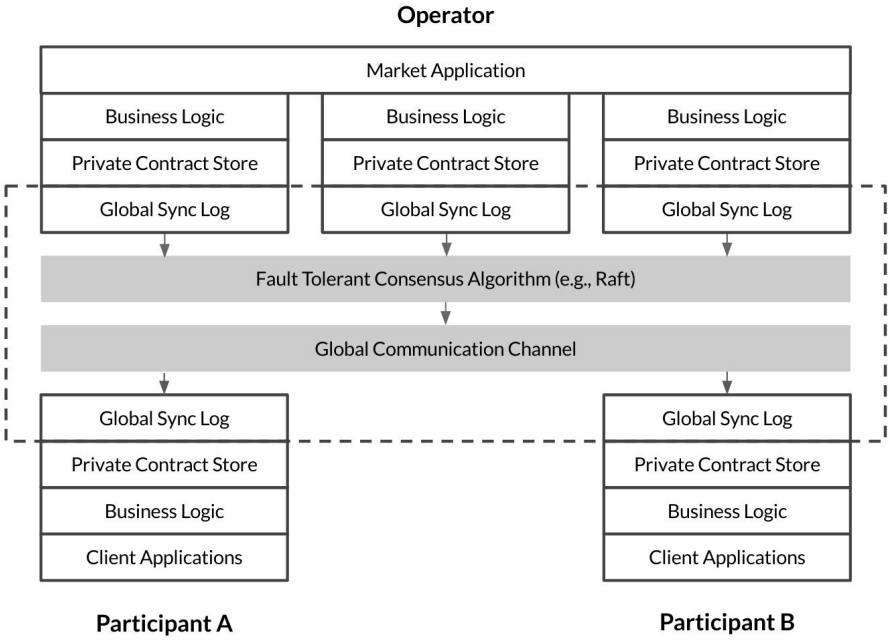

Figure 15. Example with single Operator operating multiple Operator Instances

In this illustration, the Operator is running multiple Operator Instances. The Indirect Participant has been removed for simplicity.

4.3 Distributed Ledger Solution with Multiple Untrusting Operators

Many markets do not have existing centralized infrastructure, and the creation of a new entity may be impractical or undesirable. Even in markets with central entities, it may be preferable to have multiple Operators in the same network, for example, to promote cross-market liquidity or collateralization. It is worth noting that interoperability of distinct Distributed Ledgers is a worthy alternative to this approach, whose benefits may outweigh the technical costs of expanding to multiple Operators. This is an important topic for exploration beyond the scope of this paper.

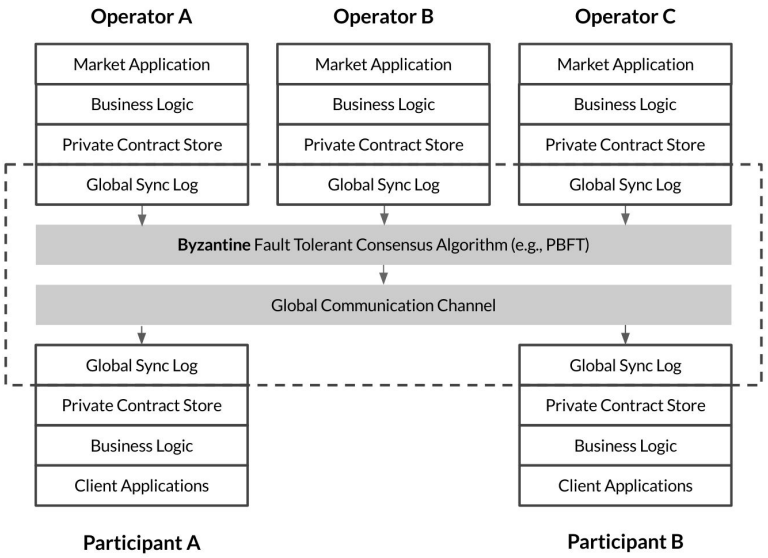

Figure 16. Example with multiple untrusting Operators

Figure 16 illustrates three Operators running one Platform Instance each, rather than one Operator running three Platform Instances, as in Figure 15. This requires a BFT Consensus Algorithm, which permits a network to withstand a degree of malicious behavior, whether deliberate or through external or internal compromise.

As mentioned above, transitioning to a network with multiple Operators makes approval from associated regulatory bodies more challenging and potentially expands security vulnerabilities. Nonetheless, there are some incremental advantages beyond the connection of multiple Operators or supporting a market without an existing central infrastructure. These include protections against fraudulent activities being evidenced in the GSL, (as opposed to the automatic detection of these activities), and potentially higher resiliency and availability through expanding the network across multiple entities.

5.0 Extensibility and Interoperability

Technologies implemented by financial institutions must generally undergo extensive review and require expensive integration. Once deployed, these technologies can remain in place for years or even decades. As such, they need to stand the test of time and should not be functionally limited in the services they can perform nor markets they can support. Extensibility of this technology is important, therefore the DA Platform serves not only to replace existing legacy software, but allows for the consolidation of technologies and the possibility of opening up new business models with faster time-to-market.

5.1 Application Programming Interface

The DA Platform has a modern, language agnostic API that conforms to Web standards allowing it to be easily extensible through the development of Applications to run on the DA Platform.

This enables ease of adoption, accelerating innovation by institutions and their clients and also easing integration by aligning with standards that reduce the cost of integration with legacy systems.

5.2 Open Source Technology

Digital Asset firmly believes that using and contributing software back to the open source community fuels innovation, lowers costs, allows for critical inspection of the source code, and provides enormous benefits to end users and the industry as a whole, including our clients. Our objective in supporting open source work has been, and will continue to be, to promote standardization, to encourage interoperability across distinct platforms and ledgers, and hence to drive adoption.

Digital Asset has demonstrated its commitment to open, industry standards as a founding premier member of Hyperledger at the Linux Foundation. We contributed the Hyperledger name and tens of thousands of lines of code, and have been an active contributor and maintainer since, with our CEO and CMO elected as the first Chairs of the Governing Board and Marketing Committee.

Digital Asset also intends to open source DAML. Work is ongoing to ensure DAML has the appropriate functionality, documentation, and developer toolkit for use outside Digital Asset. Once that work is complete, we will release the language specifications publicly, followed by an appropriate Developer Studio — including software, static analysis capabilities, and other tools. By making DAML more widely available, we intend to enable clients, partners, and other vendors to develop, modify, and extend DAML Libraries for use with the DA

Platform or other platforms, fostering a vibrant ecosystem of vendors and solutions.

6.0 Conclusion

Digital Asset believes that Distributed Ledger Technology offers significant opportunities to reduce risk, capital and costs.

Many early DLT solutions evolved outside of wholesale finance in response to very different design constraints and are therefore structurally inappropriate for application to regulated markets.

The Digital Asset Platform, with its unique approach to providing a distributed network while protecting privacy, constitutes a sophisticated solution to tackling the requirements of financial institutions in adopting this new technology. Coupled with the power of DAML, an easy to use, distributed, privacy-preserving and modular modeling language, the Digital Asset Platform provides a common foundation for financial innovation.

By developing our software specifically for wholesale, regulated financial markets we deliver the following key benefits:

Continuous data integrity

Users of this technology will be able to know, with certainty, that they and others involved in a given transaction are all working from identical copies of the same data and business logic. This is achieved with total confidentiality, so that sensitive business information is not shared with any party not entitled to view it, whether encrypted or not. Ultimately, this results in reduced operational risk and costs due to the elimination of errors and reconciliation requirements.

Increased market transparency

Participants and regulators with legal rights to see data will be able to make meaningful, real-time queries against a shared, irrevocable, single source of truth. Participants will be able to analyze with certainty how the contracts they are party to will behave under a variety of scenarios. Risk is immediately reduced, with potential reductions in capital requirements.

Accelerated financial application innovation

Institutions will be able to leverage their systems and data to offer new services which were previously prohibitively expensive. Due to the inherently analyzable and modular nature of DAML Libraries, new applications can be securely and rapidly developed and deployed.

Glossary

Consensus Algorithm

A mechanism required for achieving agreement on a consistent view of the state of a ledger in the event of distributed writing. The consensus mechanism prevents censorship, adjudicates among conflicting versions of independent untrusted Operators and tolerates Operator failure(s). If it is a requirement to prevent the network from halting in the event of malicious behavior by a minority of Operatros (whether purposeful or due to compromise), the consensus mechanism must be Byzantine Fault Tolerant (BFT). BFT Consensus algorithms are a source of latency and an ongoing area of research.

Distributed

A condition involving two or more distinct parties acting in any of three capacities (signer, writer, reader) in a ledger.

Distributed Ledger

A ledger, or data store, which is automatically kept in sync across multiple entities.

Distributed Ledger Network

The Distributed Ledger is stored, updated, and validated across the Distributed Ledger Network, which consists of all Network Participants.

Distributed Reading

Reading is the action of monitoring and receiving information from a ledger and re-executing the logic that produced the events in order to interpret and validate the information. Participants may be entitled to view all information (in the case of a regulator or market operator), or view partial information that pertains to them (in the case of a market participant). All participants perform reading and verify the integrity of the ledger through re-executing the DAML commands they are party to.

All Network Participants engage in Distributed Reading. Indirect Participants delegate this act to a Network Participant to read on their behalf.

Distributed Signing

Signing is the action of authenticating and authorizing transactions using private keys, requiring the user to maintain key security. Network Participants can engage in signing. Indirect Participants delegate this act to a Network Participant to act on their behalf.

Distributed Writing

Writing is the action of committing data to a ledger, requiring a Consensus Algorithm to reach a consistent state of the ledger for all parties. Only Operators running Operator Instances are able to commit events to the GSL.

Indirect Participant

A participant which has delegated actions on the Distributed Ledger to a third party.

Ledger

Ledger is generally shorthand for Distributed Ledger. In some cases, such as the initial phase of a phased deployment, the Ledger may not yet be Distributed.

Network

A network is all of the Platform Instances that interact with the same Distributed Ledger. Shorthand for "Distributed Ledger network."

Network Participant

A Network Participant is a participant in the Distributed Ledger that operates a Digital Asset Platform Instance. Network Participants are Operators or Participants.

Operator

A participant who operates the rules of the market and governs access to the Distributed Ledger.

Participant

A participant which is not an Operator but has access to the Distributed Ledger and can partake in Distributed Reading and Distributed Signing.

Permissioned Ledger

A Distributed Ledger accessible (for reading or for writing) only by known and approved parties.

Platform Instance

A Platform Instance is one server running the Digital Asset Platform stack. Each participant can run multiple Platform Instances for resilience purposes, and instances can be permissioned to be Participant Instances or Operator Instances.

Participant Instance

A Participant Instance performs the function of Distributed Reading. It is an instance of the Digital Asset Platform and is identical to an Operator Instance, with the exception that it is not permissioned to write to the GSL. By operating a Participant Instance, a Network Participant can rely on their local data being valid and consistent with that of all counterparties and entities entitled to view the same data.

Operator Instance

An Operator Instance performs the function of Distributed Writing. It is an instance of the Digital Asset Platform and is identical to a Participant Instance, with the exception that it is also permissioned to write to the GSL. A Network Participant that runs an Operator Instance is referred to as an Operator. There can be one or many Operators in the same Distributed Ledger network.