



Digital Asset

Functional Modelling of Contractual Workflows

DAML and the DLT Architectural Style

Who am I

Andrae Muys (@etymon)

Engineer working at Digital Asset

Part of the Digital Asset team that in 2017, built the first existence proof that DLT can simultaneously meet the HA, DR, throughput, and functionality requirements of the CHES+ replacement system.

Currently working on the production version of CHES+, the replacement Clearing and Settlement system for ASX





Legal Requirements of Contractual Workflows

Legal Contracts

Legal requirements for formation and execution





Legal Contracts

Legal requirements for formation and execution

- Multiple autonomous parties
- Need to establish trust



Legal Contracts

Legal requirements for formation and execution

- Multiple autonomous parties
- Need to establish trust
- Certainty
- Offer and Acceptance



Legal Contracts

Legal requirements for formation and execution

- Multiple autonomous parties
- Need to establish trust
- Certainty
- Offer and Acceptance
- Obligations to perform
- Grants of rights



Legal Contracts

Legal requirements for formation and execution

- Multiple autonomous parties
- Need to establish trust
- Certainty
- Offer and Acceptance
- Obligations to perform
- Grants of rights
- Privity of contract
- Privacy and Monitoring



Legal Contracts

Legal requirements for formation and execution

- Multiple autonomous parties
 - Need to establish trust
 - Certainty
 - Offer and Acceptance
 - Obligations to perform
 - Grants of rights
 - Privity of contract
 - Privacy and Monitoring
- + 2 non-technical requirements: (Intent to be legally bound, and Consideration)



Technical Requirements of Contractual Workflows



Legal Contracts

Technical requirements for formation and execution

- Multiple autonomous parties
- Need to establish trust
- Certainty
- Offer and Acceptance
- Obligations and Rights
- Privity of contract
- Privacy
- Monitoring of performance



Legal Contracts

Technical requirements for formation and execution

- Multiple autonomous parties
Distributed consensus
- Need to establish trust
- Certainty
- Offer and Acceptance
- Obligations and Rights
- Privity of contract
- Privacy
- Monitoring of performance



Legal Contracts

Technical requirements for formation and execution

- Multiple autonomous parties
Distributed consensus
- Need to establish trust
Independent validation across autonomous trust boundaries
- Certainty
- Offer and Acceptance
- Obligations and Rights
- Privity of contract
- Privacy
- Monitoring of performance



Legal Contracts

Technical requirements for formation and execution

- Multiple autonomous parties
Distributed consensus
- Need to establish trust
Independent validation across autonomous trust boundaries
- Certainty
Deterministic execution
- Offer and Acceptance
- Obligations and Rights
- Privity of contract
- Privacy
- Monitoring of performance



Legal Contracts

Technical requirements for formation and execution

- Multiple autonomous parties
Distributed consensus
- Need to establish trust
Independent validation across autonomous trust boundaries
- Certainty
Deterministic execution
- Offer and Acceptance
Model of consent and authorization
- Obligations and Rights
- Privity of contract
- Privacy
- Monitoring of performance



Legal Contracts

Technical requirements for formation and execution

- Multiple autonomous parties
Distributed consensus
- Need to establish trust
Independent validation across autonomous trust boundaries
- Certainty
Deterministic execution
- Offer and Acceptance
Model of consent and authorization
- Obligations and Rights
Model of delegation and change
- Privity of contract
- Privacy
- Monitoring of performance



Legal Contracts

Technical requirements for formation and execution

- Multiple autonomous parties
Distributed consensus
- Need to establish trust
Independent validation across autonomous trust boundaries
- Certainty
Deterministic execution
- Offer and Acceptance
Model of consent and authorization
- Obligations and Rights
Model of delegation and change
- Privity of contract
Model of obligation
- Privacy
- Monitoring of performance



Legal Contracts

Technical requirements for formation and execution

- Multiple autonomous parties
Distributed consensus
- Need to establish trust
Independent validation across autonomous trust boundaries
- Certainty
Deterministic execution
- Offer and Acceptance
Model of consent and authorization
- Obligations and Rights
Model of delegation and change
- Privity of contract
Model of obligation
- Privacy
Perfect forward secrecy
- Monitoring of performance



Legal Contracts

Technical requirements for formation and execution

- Multiple autonomous parties
Distributed consensus
- Need to establish trust
Independent validation across autonomous trust boundaries
- Certainty
Deterministic execution
- Offer and Acceptance
Model of consent and authorization
- Obligations and Rights
Model of delegation and change
- Privity of contract
Model of obligation
- Privacy
Perfect forward secrecy
- Monitoring of performance
Guaranteed notification



Digital (smart) Legal Contracts (Requirements)

So, to support real world legal contracts, a smart contract language must be:

A distributed multi-party system that crosses autonomous trust boundaries, supporting deterministic evolution of consensus of belief that is independently validatable; supported by formal models of authorization (consent, delegation, and obligation).

It should also simultaneously guarantee *perfect forward secrecy*, and *notification* of relevant events.

Oh, and if you're modeling financial contracts: it needs to scale to billions of individual contracts, and millions of contract events per minute.



Architectural Style supporting Contractual Workflows



Distributed Multi-Party

Consensus of Belief
Deterministic Certainty
Core Architecture Style (CQRS)
Trust Boundaries
Validatable
Modeling Consent
Modeling Delegation
Modeling Privity
Stepwise Determinism
Privacy
Notifications
Final Architecture Style





Distributed Multi-Party

Consensus of Belief

Deterministic Certainty
Core Architecture Style (CQRS)
Trust Boundaries
Validatable
Modeling Consent
Modeling Delegation
Modeling Privity
Stepwise Determinism
Privacy
Notifications
Final Architecture Style



Synchronization Layer

S



Distributed Multi-Party
Consensus of Belief

Deterministic Certainty

Core Architecture Style (CQRS)

Trust Boundaries

Validatable

Modeling Consent

Modeling Delegation

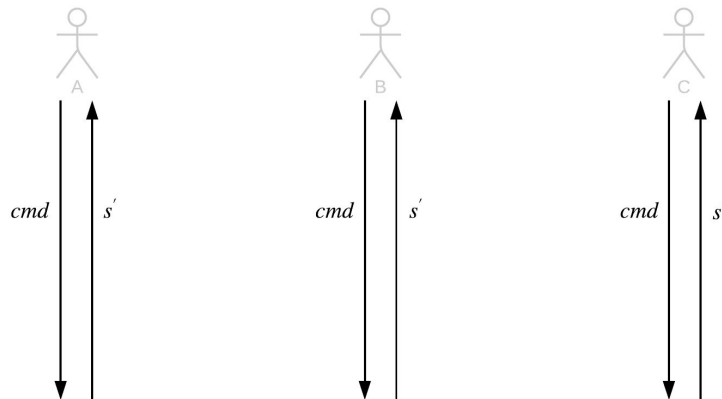
Modeling Privity

Stepwise Determinism

Privacy

Notifications

Final Architecture Style



Synchronization Layer

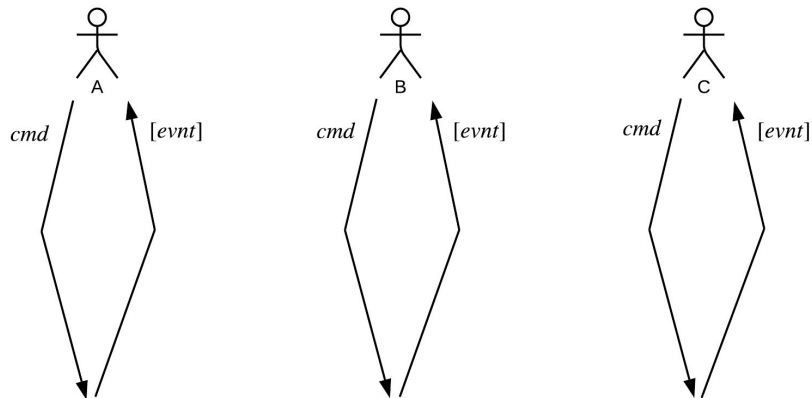
$$^S Eval_{state} : s \rightarrow cmd \rightarrow s'$$



Distributed Multi-Party
Consensus of Belief
Deterministic Certainty

Core Architecture Style

Trust Boundaries
Validatable
Modeling Consent
Modeling Delegation
Modeling Privity
Stepwise Determinism
Privacy
Notifications
Final Architecture Style



Synchronization Layer

$$S \quad Eval_{cmd} : s \rightarrow cmd \rightarrow [evnt]$$

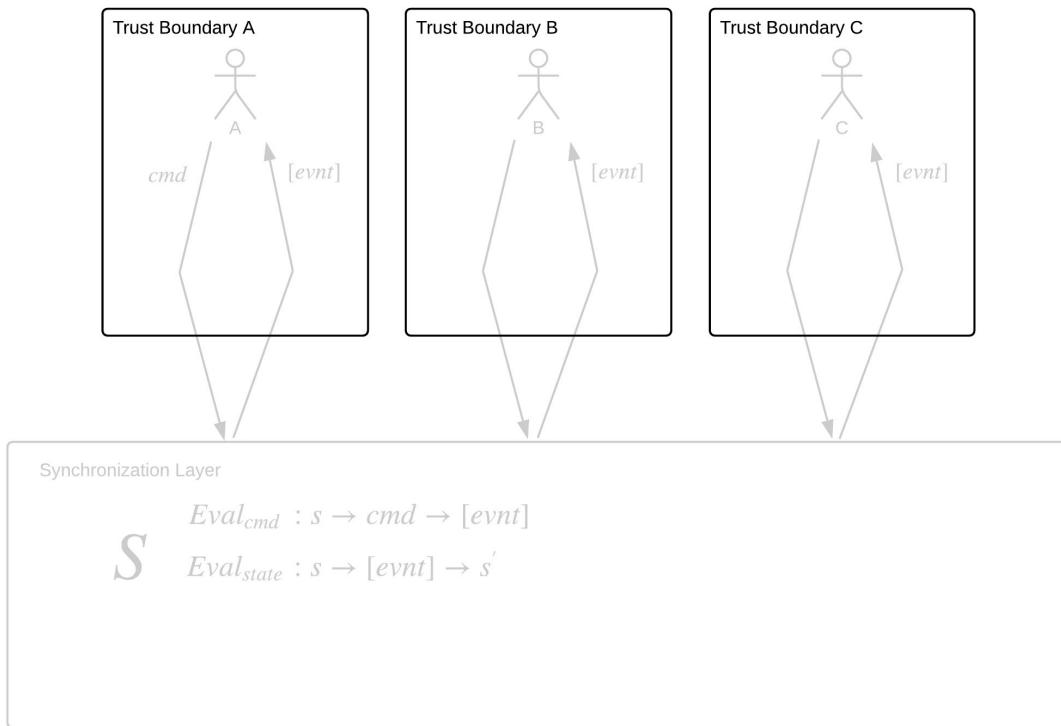
$$Eval_{state} : s \rightarrow [evnt] \rightarrow s'$$



Distributed Multi-Party
Consensus of Belief
Deterministic Certainty
Core Architecture Style (CQRS)

Trust Boundaries

Validatable
Modeling Consent
Modeling Delegation
Modeling Privacy
Stepwise Determinism
Privacy
Notifications
Final Architecture Style

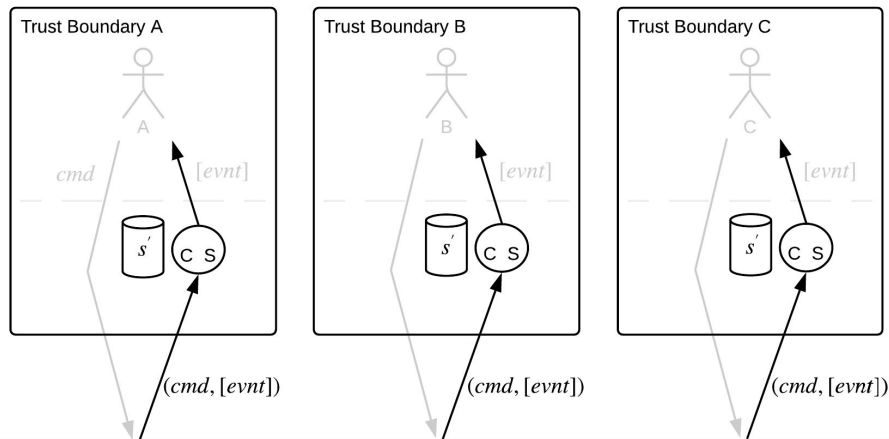




Distributed Multi-Party
Consensus of Belief
Deterministic Certainty
Core Architecture Style (CQRS)
Trust Boundaries

Validatable

Modeling Consent
Modeling Delegation
Modeling Privity
Stepwise Determinism
Privacy
Notifications
Final Architecture Style



Synchronization Layer

$$S \quad Eval_{cmd} : s \rightarrow cmd \rightarrow (cmd, [evnt])$$

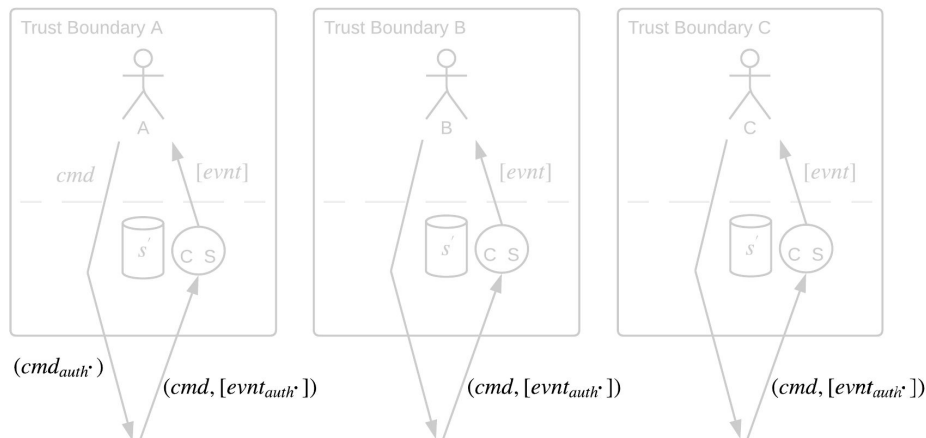
$$Eval_{state} : s \rightarrow [evnt] \rightarrow s'$$



Distributed Multi-Party
Consensus of Belief
Deterministic Certainty
Core Architecture Style (CQRS)
Trust Boundaries
Validatable

Modeling Consent

Modeling Delegation
Modeling Privity
Stepwise Determinism
Privacy
Notifications
Final Architecture Style



Synchronization Layer

S

$$Eval_{cmd} : s \rightarrow cmd_{auth} \rightarrow (cmd, [evnt_{auth}])$$

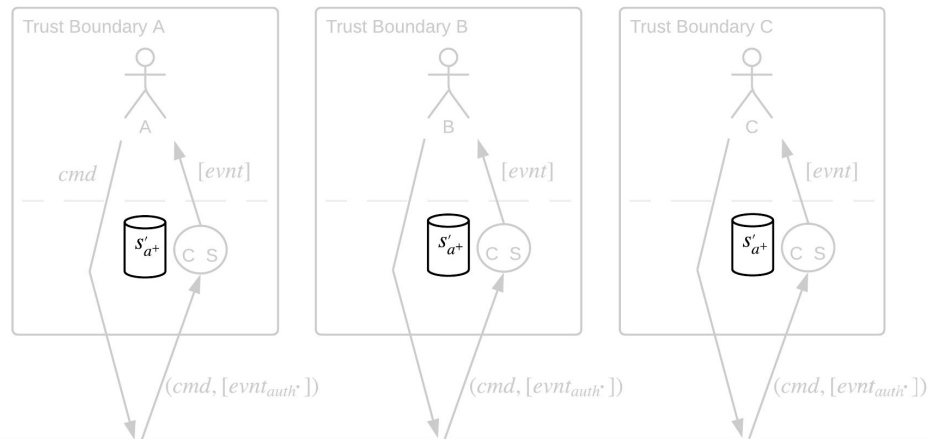
$$Eval_{state} : s \rightarrow [evnt_{auth}] \rightarrow s'$$



Distributed Multi-Party
Consensus of Belief
Deterministic Certainty
Core Architecture Style (CQRS)
Trust Boundaries
Validatable
Modeling Consent

Modeling Delegation

Modeling Privity
Stepwise Determinism
Privacy
Notifications
Final Architecture Style



Synchronization Layer

$$S_{auth^+} \quad Eval_{cmd} : s_{auth^*} \rightarrow cmd_{auth^*} \rightarrow (cmd, [evnt_{auth^+}])$$

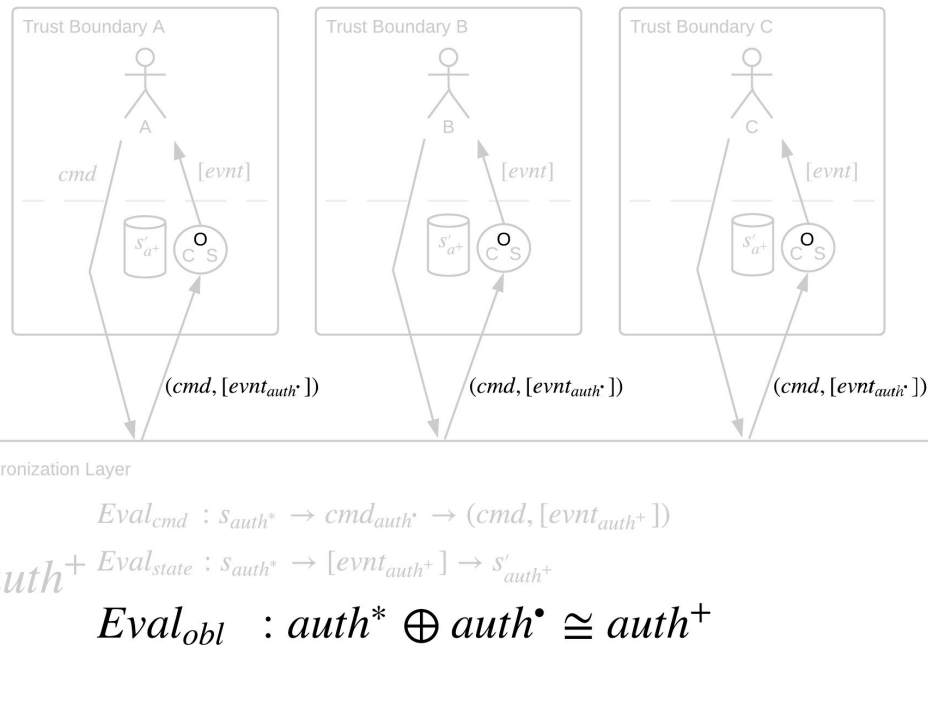
$$Eval_{state} : s_{auth^*} \rightarrow [evnt_{auth^+}] \rightarrow s'_{auth^+}$$



Distributed Multi-Party
 Consensus of Belief
 Deterministic Certainty
 Core Architecture Style (CQRS)
 Trust Boundaries
 Validatable
 Modeling Consent
 Modeling Delegation

Modeling Privity

Stepwise Determinism
 Privacy
 Notifications
 Final Architecture Style

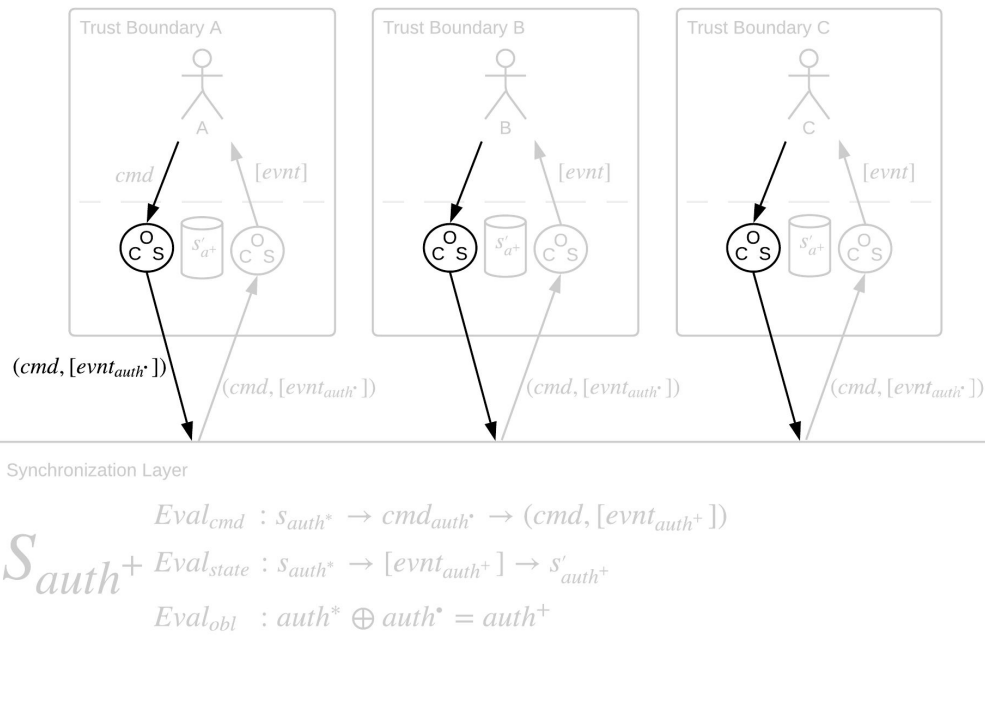


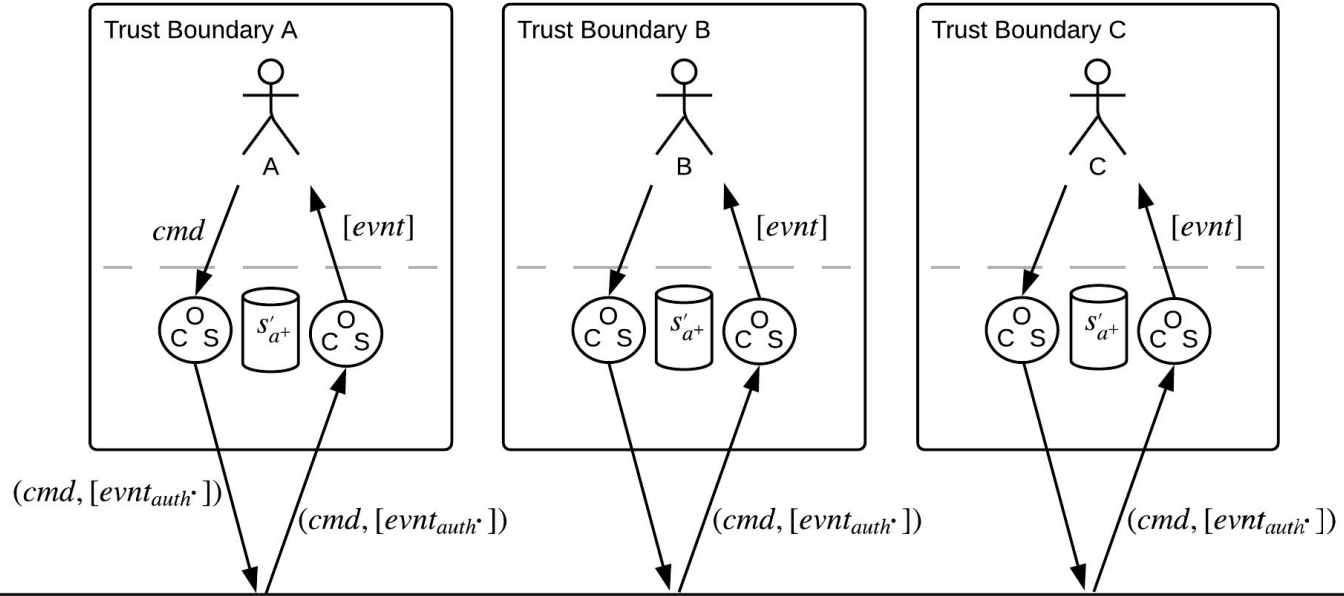


Distributed Multi-Party
 Consensus of Belief
 Deterministic Certainty
 Core Architecture Style (CQRS)
 Trust Boundaries
 Validatable
 Modeling Consent
 Modeling Delegation
 Modeling Privity

Stepwise Determinism

Privacy
 Notifications
 Final Architecture Style





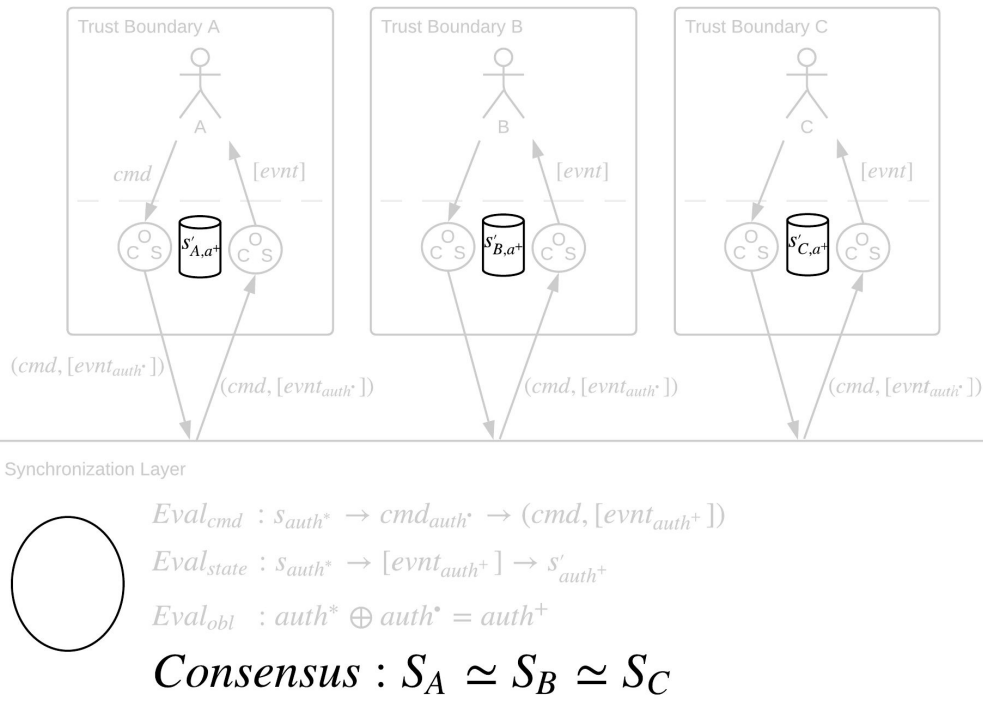
$$\begin{aligned}
 & Eval_{cmd} : s_{auth}^* \rightarrow cmd_{auth}^* \rightarrow (cmd, [evnt_{auth}^+]) \\
 & S_{auth}^+ Eval_{state} : s_{auth}^* \rightarrow [evnt_{auth}^+] \rightarrow s'_{auth}^+ \\
 & Eval_{obl} : auth^* \oplus auth^* = auth^+
 \end{aligned}$$



Distributed Multi-Party
 Consensus of Belief
 Deterministic Certainty
 Core Architecture Style (CQRS)
 Trust Boundaries
 Validatable
 Modeling Consent
 Modeling Delegation
 Modeling Privity
 Stepwise Determinism

Privacy

Notifications
 Final Architecture Style

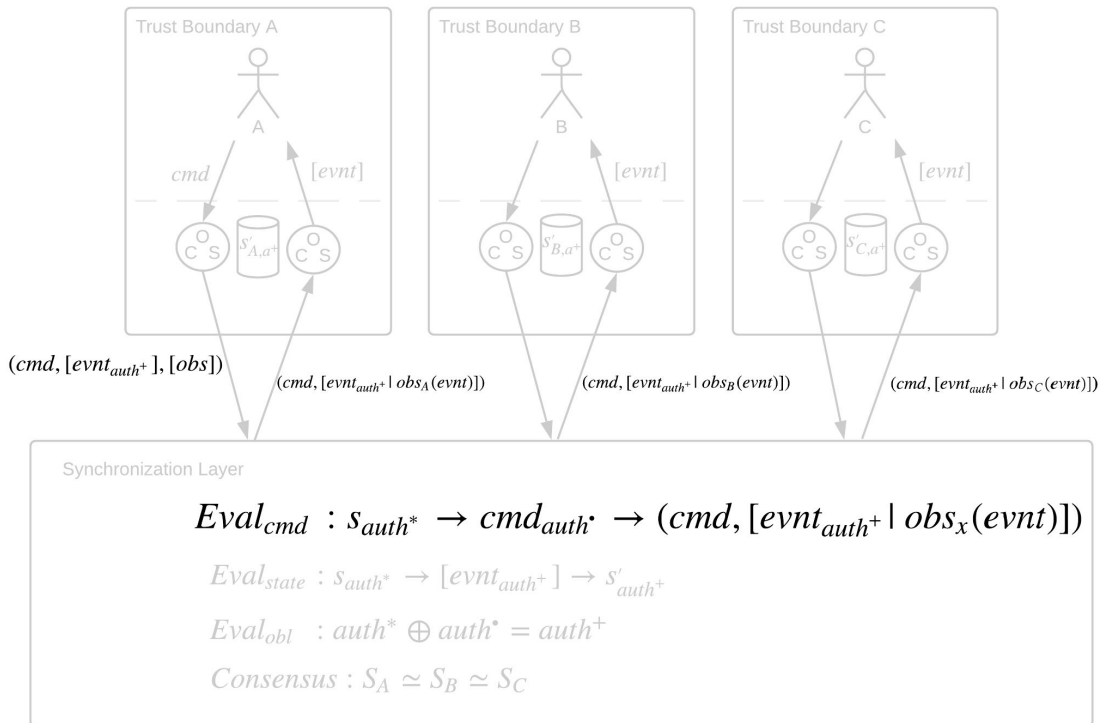


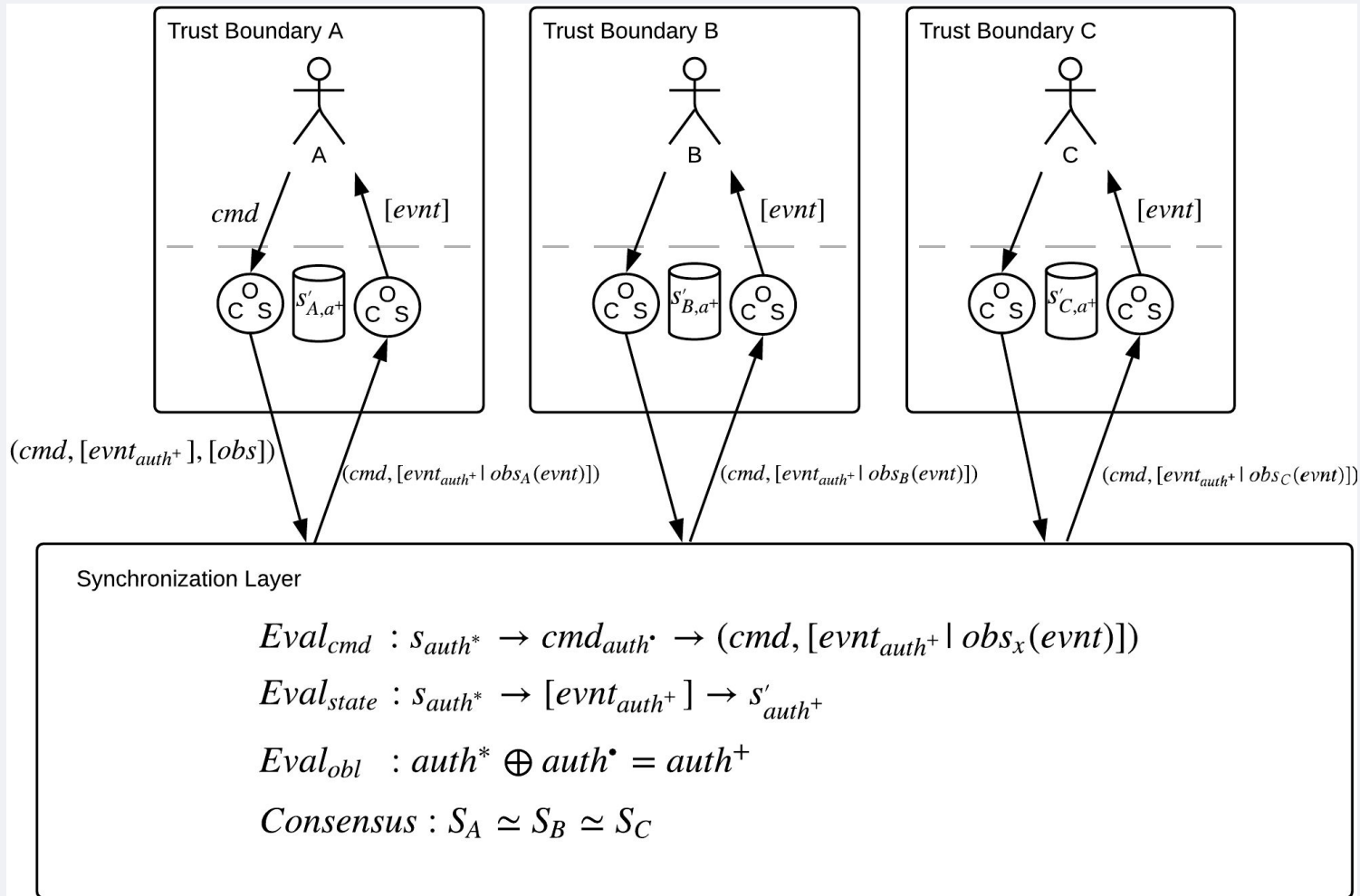


Distributed Multi-Party
 Consensus of Belief
 Deterministic Certainty
 Core Architecture Style (CQRS)
 Trust Boundaries
 Validatable
 Modeling Consent
 Modeling Delegation
 Modeling Privity
 Stepwise Determinism
 Privacy

Notifications

Final Architecture Style







CardMarket.daml



DAML (Digital Asset Modeling Language)

- Pure-Functional Haskell dialect
- Formal Operational Semantics that provides
 - Technology-neutral ledger model
 - Privacy and Disclosure
 - Authorization and Consent
 - Completely deterministic evaluation
- Digital Asset SDK provides
 - DAML IDE support (Visual Studio plugin)
 - DAML/DLT sandbox suitable for testing/debugging/PoC-dev
 - DAML compiler to intermediate canonical form
 - DAML interpreter
- Open-Source (<https://github.com/digital-asset/daml>)

Synchronization/Consensus Layer

- Distributed Consensus via
 - Total ordering on Commands
 - All multi-party coordination is via the Transaction Log
- Predicated Transactions (Commands + Expected Events)
 - Commands (Create + Exercise)
 - Events (Created + Archived)
 - Aggregated State := events \Rightarrow filter (valid) \Rightarrow fold (created - archived)



DAML (Modelling)

- We model Rights to act (ie. “Gates” or “Choices”) NOT data
- A digital ledger is not a database — we aren’t using CRUD
Rather, it is a distributed transaction log of choices made by parties

(this is intension vs. extension)

Not: "This state exists" or "Party A possesses token T"

Rather: "Party A chose act M, now Party B now has right to chose act N"

ie. "A makes offer to B":

Not: "Party B is now in possession of an 'Offer' from Party A"

Rather: "Party A chose to make offer", now

“Party B now has the choice of 'Accept Offer' or 'Reject Offer'”



DAML (Proving facts/beliefs)

Each contract on the ledger represents a formal proof that the signatories to the contract have agreed that a particular fact is true.

or more formally: an observer of a contract has a justified basis for belief that the signatories reached a consensus of belief: that a predicate holds for the lifetime of the contract





CardTrader.daml



DAML Resources

DA Website: <https://digitalasset.com/>

DAML: <https://daml.com/>

DAML API Hoogle: <https://hoogle.daml.com/>

DAML source code: <https://github.com/digital-asset/daml>

Further Reading: <https://medium.com/daml-driven>

Public Slack: <https://damldriven.slack.com>



Andrae Muys
e: andrae.muys@digitalasset.com
t: @etymon

Questions

For details on the CHESSE Replacement:

<https://www.asx.com.au/services/chess-replacement.htm>

<mailto:CHESSEReplacement@asx.com.au>