Welcome | Microsoft Cloud User Group

www.meetup.com/msftcloud

Razi Rais | Microsoft

www.linkedin.com/in/razirais

Meeting every month

Open to everyone!

Introduction to the Corda DLT Platform

Microsoft Cloud NYC User Group

October 2017

Tom Menner
Director, Solution Architect
tom.menner@r3.com
linkedin.com/in/tommenner

Agenda

- Overview
- Corda Features
- Corda Technical Details and Network Deployment
- Corda Roadmap
- Azure Integration

Evolution from Blockchains to Distributed Ledgers

2009

Bitcoin Release

Altcoin Rush

Meta Protocols, Colored Coins, and Tokenization

ethereum

Extends the concept in Bitcoin of '*decentralized storage* to '*decentralized computer*'

2015

R3 Consortium begins

Hyperledger Project

Towards blockchain inspired technologies – DLT

Corda open sourced

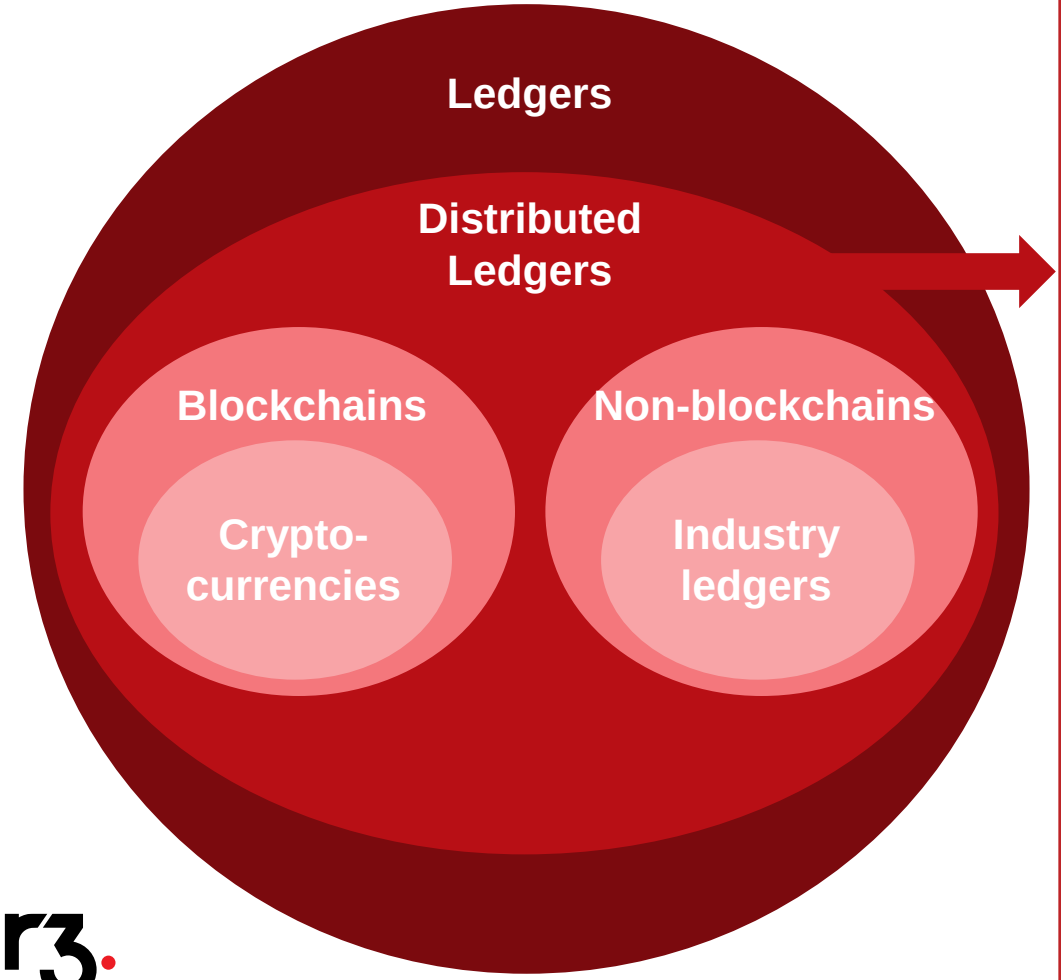
Today

Corda V1

Defining characteristic of a distributed ledger

Distributed ledgers are systems that enable parties who don’t fully trust each other to form and maintain consensus about the existence, status and evolution of a set of shared facts.

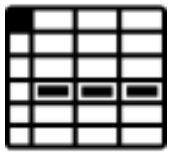
Distributed ledgers at a glance



Cryptography to ensure identity authentication for each transaction

Non-repudiation/immutability to preserve integrity of data and create an audit trail

Smart contracts for the automatic execution of business logic when certain criteria are met



Shared ledger so each participant sees the same view of the same data, updated in real time, subject to permissioning

Distributed consensus to ensure the state of the ledger represents the agreed-upon truth of all stakeholders

A unique shared ledger approach

Blockchain-inspired: takes best attributes from Bitcoin, Ethereum, and others.

Consensus: achieved at individual deal level, rather than system level. Supports a variety of consensus mechanisms.

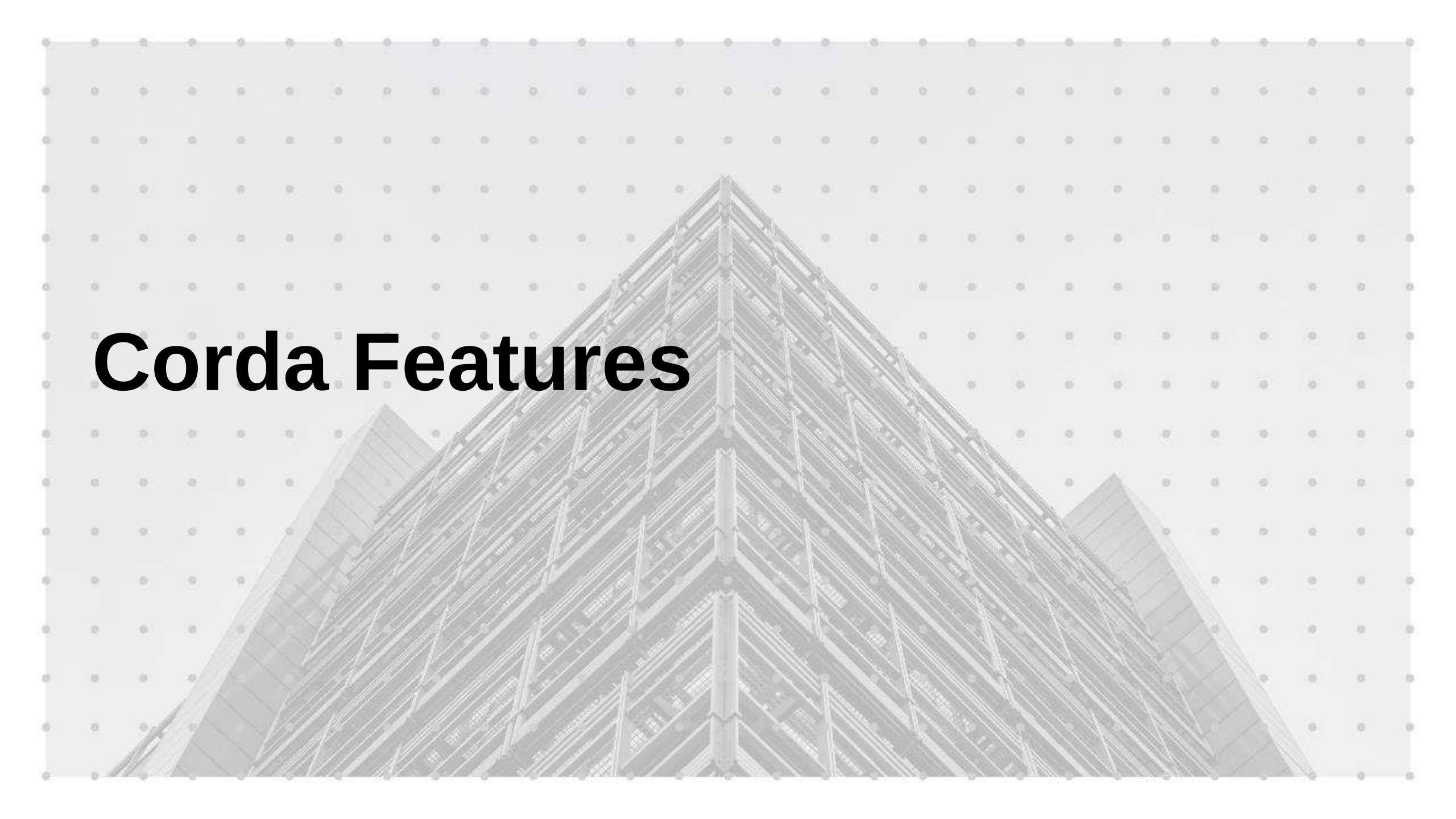
Enterprise grade: built specifically for financial markets.

Regulator-focused: design directly enables regulatory/supervisor observer nodes.

Data privacy: transactions info propagated only to relevant nodes.

Smart contract: strong link between legal prose and smart contract code.

Easy integration: reuse existing developer skills and make integration with bank systems easy and safe. Query and join the ledger to existing DBs with SQL, and code contracts in modern, standard languages like Java.



Corda Features

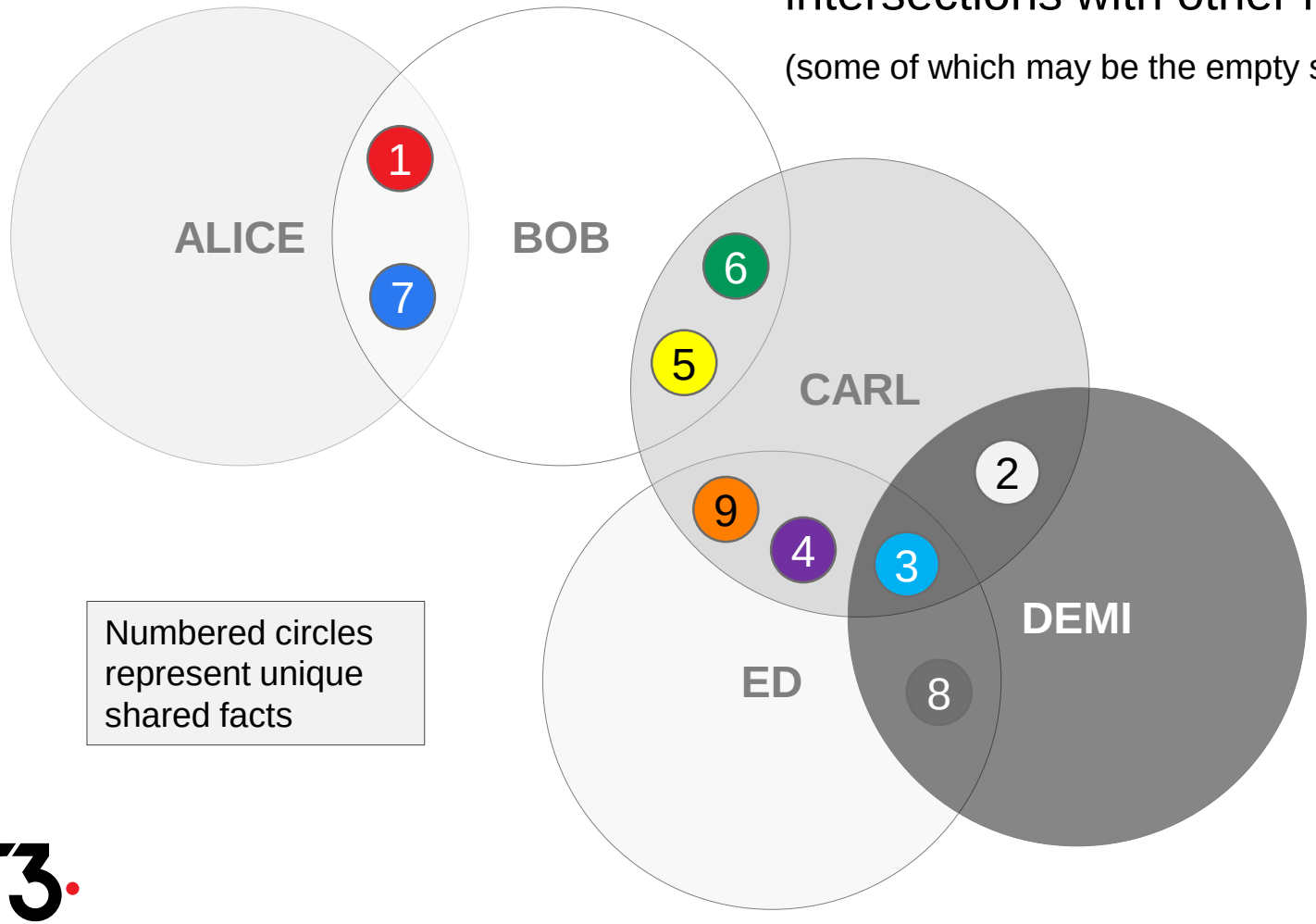
Corda salient features

- No **blockchains**, no **mining**; instead a **permissioned network**
- No **broadcast**: all communication is **point-to-point**
    - We reject the notion that data should be broadcast to all participants – or to cumbersome, predefined groups
    - Message senders need to know the identity of recipients
    - Data is shared on a **need-to-know basis** and peers only see what they need to see
    - Not sending is preferable to sending and encrypting
- Unspent Transaction Output (**UTXO**) for recording states (like Bitcoin)
- Platform is **JVM-based**, written in Kotlin (can use Java, Clojure, etc)
- Supports **industry-standard** protocols: AMQP, JDBC, PKIX, etc
- No cryptocurrency but can represent digital cash

The Corda Ledger

The ledger from each peer's point of view is the union of all intersections with other network peers

(some of which may be the empty set)



Numbered circles represent unique shared facts

ALICE = { 1 7 }

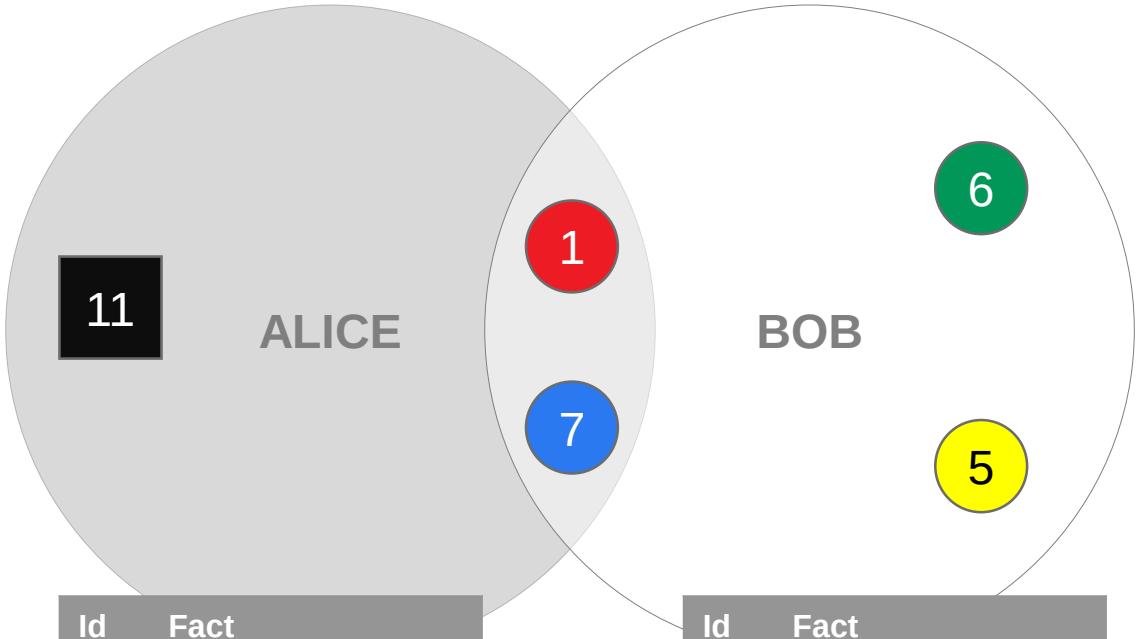
BOB = { 1 7 6 5 }

CARL = { 9 4 6 5 2 3 }

DEMI= { 2 3 8 }

ED= { 9 4 8 3 }

Anatomy of a bilateral ledger

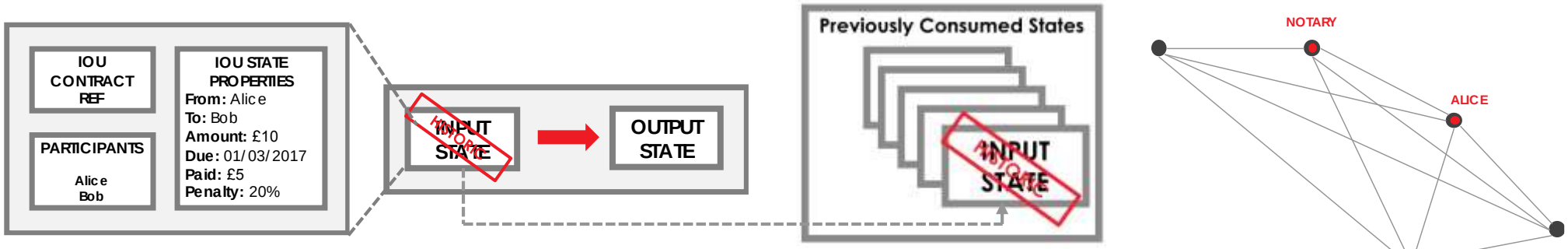


Id	Fact
1	“Much consensus”
7	“So bilateral”
11	“Wow ledger”

Id	Fact
1	“Much consensus”
6	“Very fact”
7	“So bilateral”
5	“amaze network”

- There is no “central ledger”
- Each network peer maintains a separate **vault** of facts (akin to rows in a DB table)
- All peers to a shared fact store identical copies
- Not all on-ledger facts have to be shared with other peers
  - The black square **“11”** is an example of a on-ledger fact not shared with any peers
- **Immutable:** easy to do analysis on a static snapshot of the data and reason about the contents
- **No accounts:** easy to apply transactions in parallel
- **Transaction ordering:** impossible to mis-order transactions due to reliance on hash functions to identify previous states
- **Consensus:** conflict is the double spend problem
- **Auditability:** full history of all activity is recorded

Corda: Key Concepts



State Object

States are immutable objects that represent (shared) facts such as a financial agreement or contract at a specific point in time

Transaction

Transactions consume input states and create output states.

The newly created output states replace the input states which are marked as historic.

Consensus

Parties reach consensus on the evolution of a shared fact. This is done by testing the validity (by way of contract code) and uniqueness (by way of the notary) of the transaction.

Flows

Flows are light-weight processes used to coordinate interactions required for peers to reach consensus about shared facts.

Transactions

- Any peer may create a **transaction proposal**
- Transaction proposals are uncommitted by default
- Before a transaction proposal is committed it must first be digitally **signed** and then verified and by all required peers on a need-to-know basis
- Once a transaction is committed it marks the input state references as **historic** and creates new output states reflecting an updated ledger

Flows

- With Corda, peers communicate on a **point to point** basis
  - Most distributed ledger platforms use message broadcasting and gossip networks to share data
- To communicate, peers must specify message recipients
- Recall that to commit a transaction, multiple peers are often required to sign and verify it
- To commit a transaction proposal, a workflow or **“flow”** of messaging, signing, verifying, among other things, is required
- Peers on a Corda network may have thousands of counter-parties and hundreds of thousands of concurrent flows

Two types of consensus

Peers reach consensus over transactions in two ways

In Corda, **verification consensus** involves peers reaching certainty that a transaction:

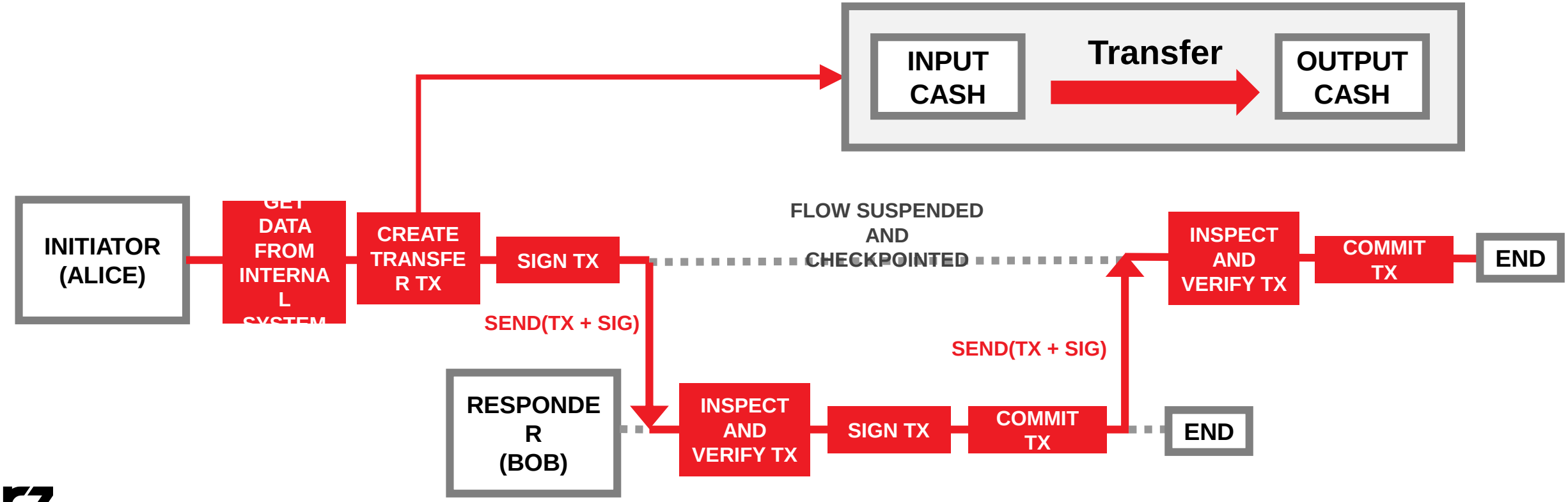
- is **signed by all required peers** listed in the commands of a flow;
- and **satisfies the constraints** defined by the contracts pointed to by the input and output states.

However there is an additional step required…

The "double spend" problem for on-ledger issued assets can be mitigated with **uniqueness consensus**

- Uniqueness consensus is provided by **notary services**
- When a state is issued on-ledger it is **assigned** a notary service
- The assigned notary ensures the state is **not used as an input to a transaction more than once** for the duration of the state's lifecycle on-ledger
- The point of transaction **finality is reached** when the specified **notary service signs** the transaction

Example: Cash Transfer Flow



Corda Technical Details and Network Deployment

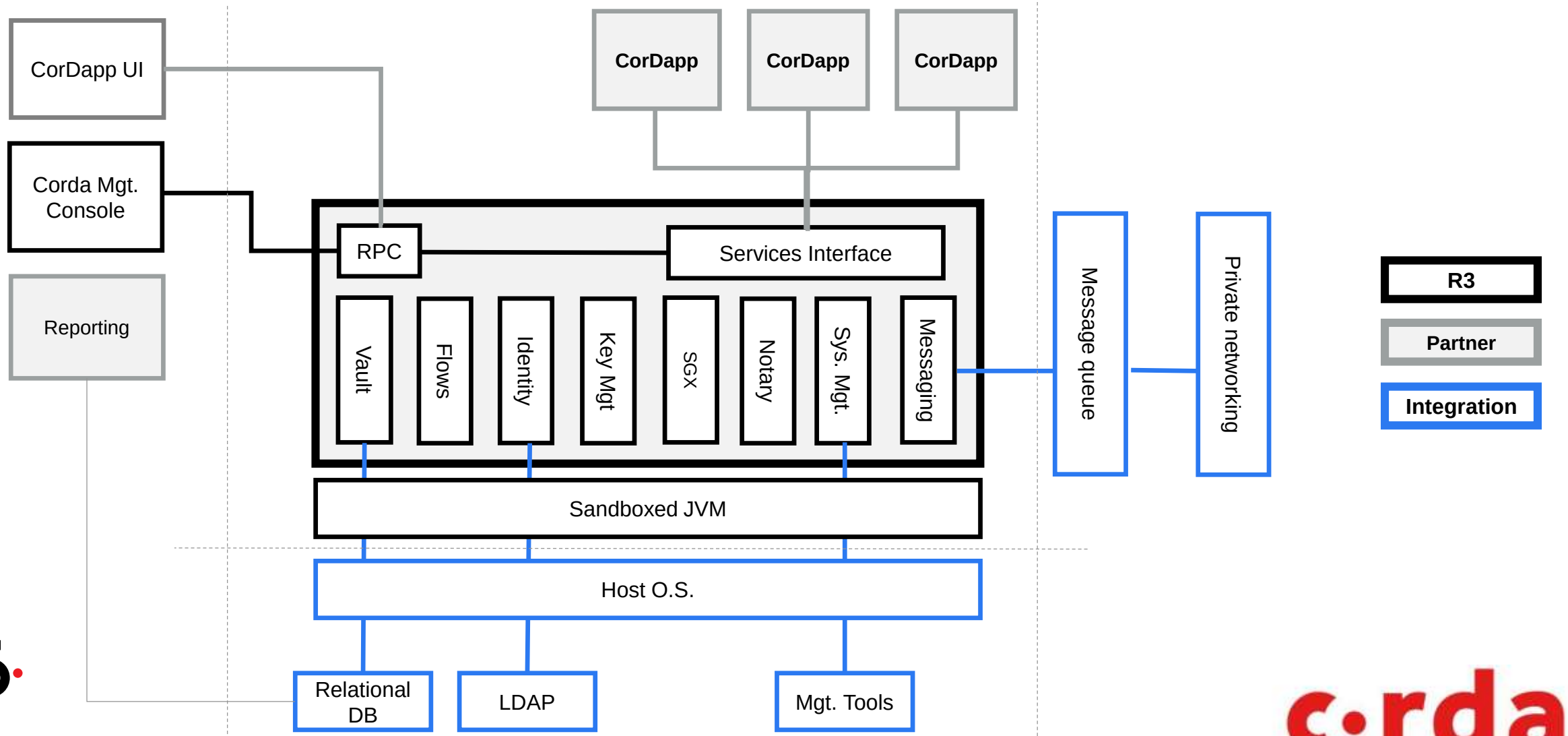
Network overview

What is a Corda Network?

A Corda network is comprised of:

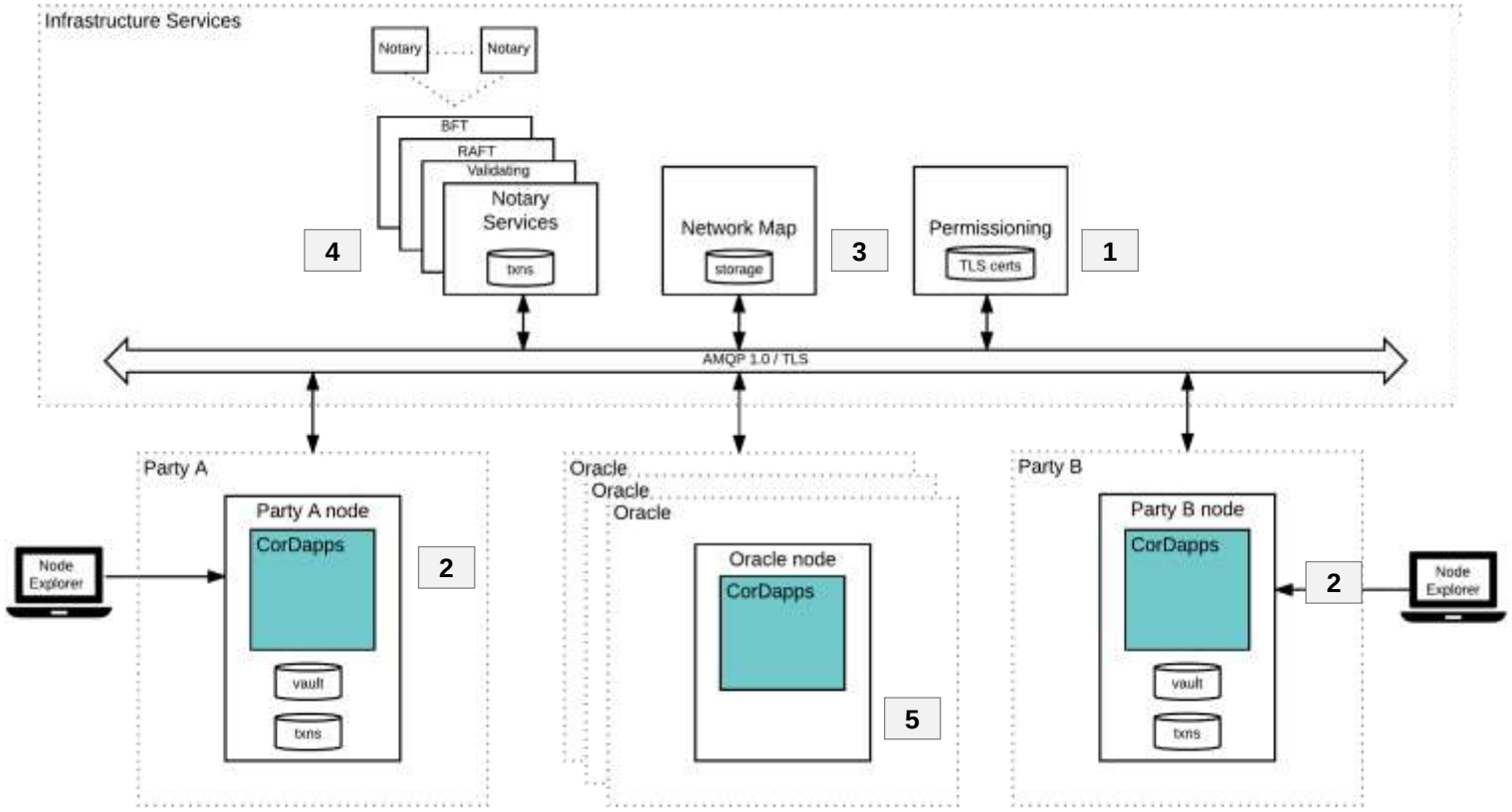
- A **doorman**
- Two or more Corda **nodes**
- A **network map** service
- One or more **notary** services
- Zero or more **oracles**

Corda node architecture



Corda Network: Detailed Overview

A Corda Network includes a 1) doorman ("permissioning service"), 2) two or more Corda Nodes, 3) a network map service, 4) one or more notary nodes and 5) zero or more oracles



1 **Doorman**: Enforces rules regarding the information nodes must provide before being admitted to the network. If satisfied, node's identity is certified with a root-authority-signed TLS certificate.

2 **Nodes**: JVM run-time with a unique network identity running Corda with two interfaces: network layer (interacting with other nodes) and RPC (interacting with node's owner)

- **Network Map Service:** Publishes IP addresses through which

3 all nodes can be reached along with certificates and services provided by node

- **Notary**: Attest uniqueness, and possibility the validity, of ledger

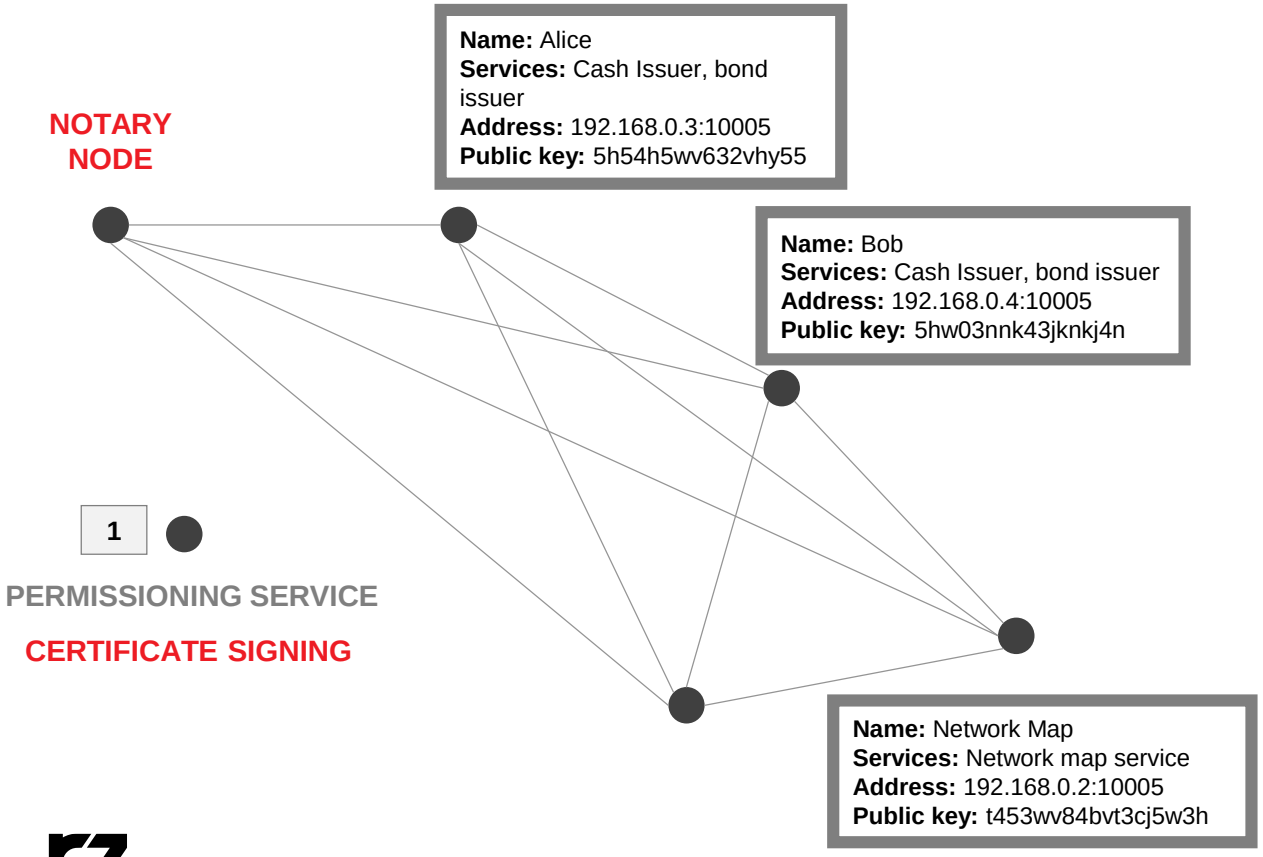
4 updates.

- **Oracles**: Well-known service that signs transactions if they

5 state a fact and that fact is considered to be trust

A Corda Network

A Corda network is an authenticated peer-to-peer network of nodes where each node is a Java Virtual Machine run-time environment hosting Corda services and executing applications known as CorDapps



- A Corda network is a **fully connected** graph
- **No global broadcast** or gossip network
- Communication occurs on a **point-to point basis** only
- Peers communicate using **AMQP/1.0** over **TLS**
- **Network map service** publishes list of peers
- Graph edges represent the **potential** to communicate, not persistent connections
- Think **Email** and **SMTP**

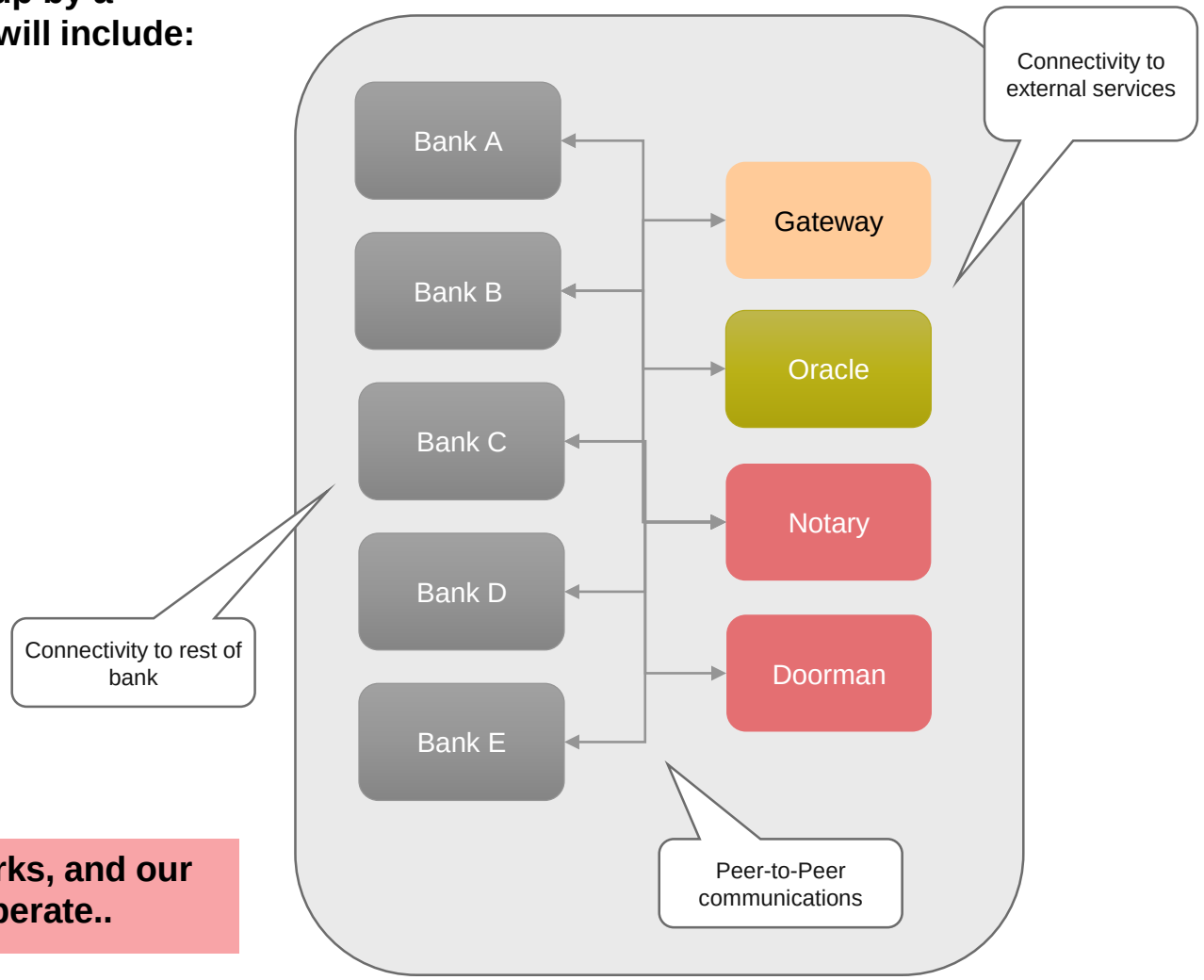
What makes a Business Network?

We expect that single business networks will typically be set up by a consortium of banks and a system delivery partner, and they will include:

- A ledger agreement / set of rules
- An operating entity
- The specific ledger application for this ledger (CorDapp)
- Common network parameters that allow Nodes to transact

A network will comprise a number of Corda Nodes:

- Bank nodes
- A Doorman Node
- At least one Notary node
- Oracles
- Messaging Gateways (e.g. SWIFT)

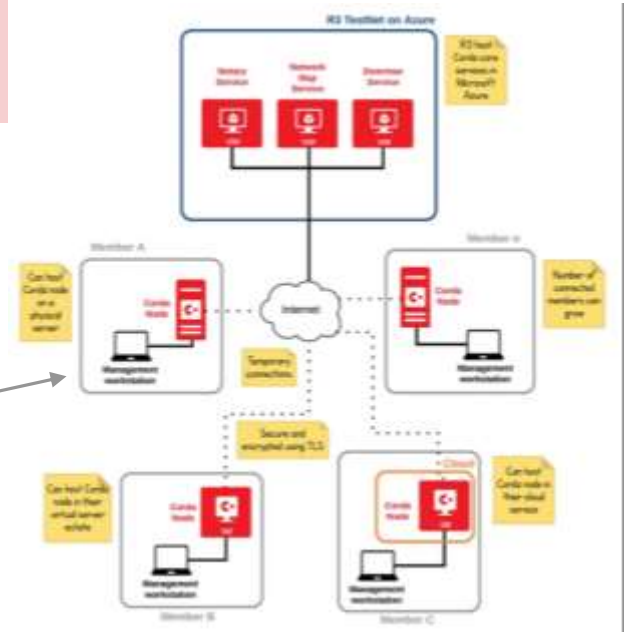


But of course we want many of these business networks, and our primary objective is for them to be able to interoperate..

TestNet and ‘R3Net’

- Projects
- Demos
- Partners
- Development and Test Focus

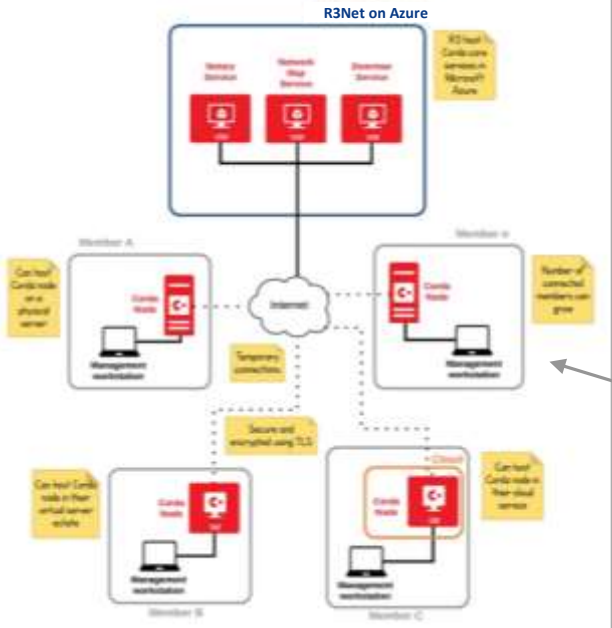
TestNet



Connectivity to Bank test environments

Change management challenge

‘R3Net’



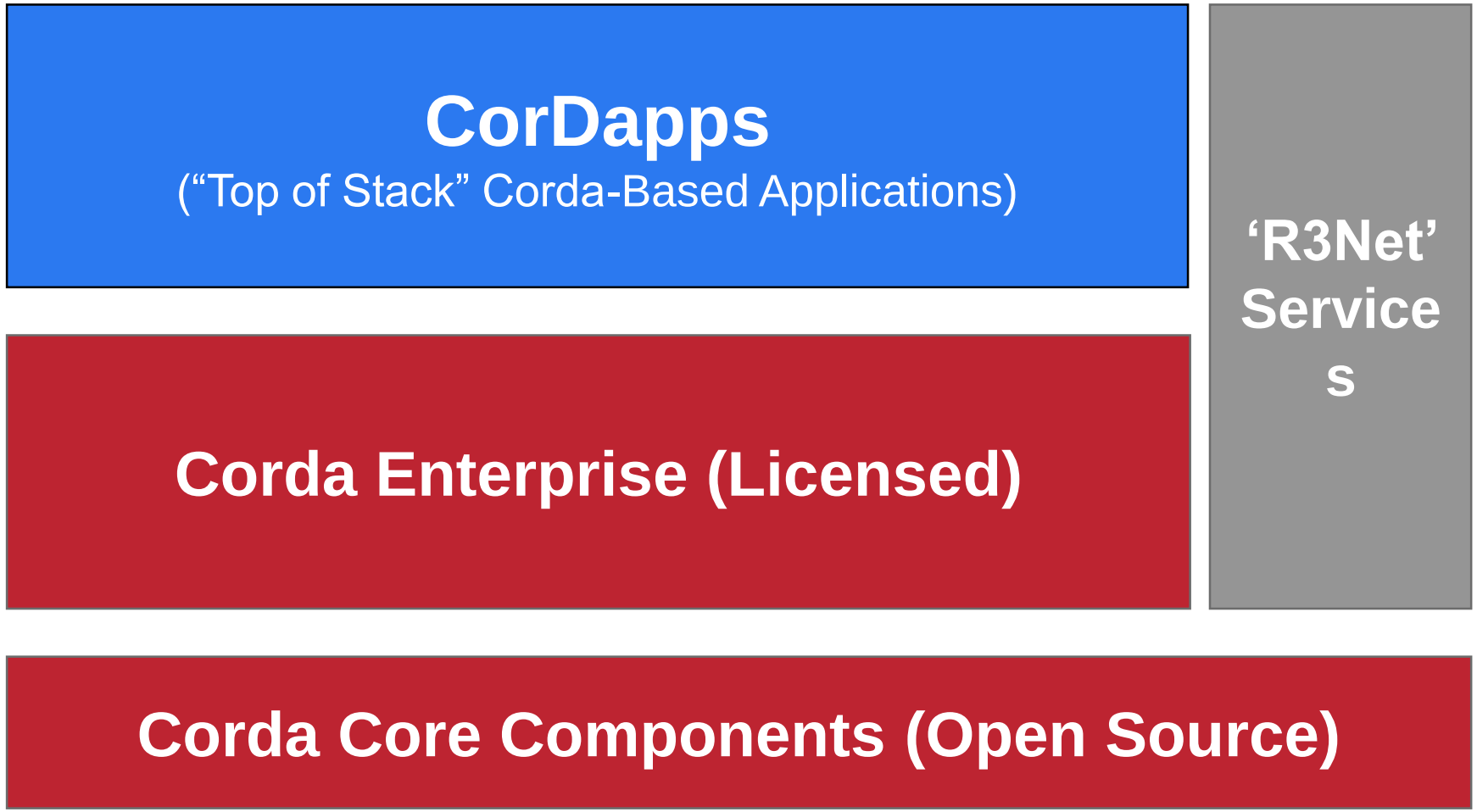
- Production Applications
- Live business
- Permissioned FIs only

Connectivity to Bank Production environments

Bank production environments are highly controlled and protected. There are many change management hurdles for banks to negotiate before they can connect Corda nodes to their production environments

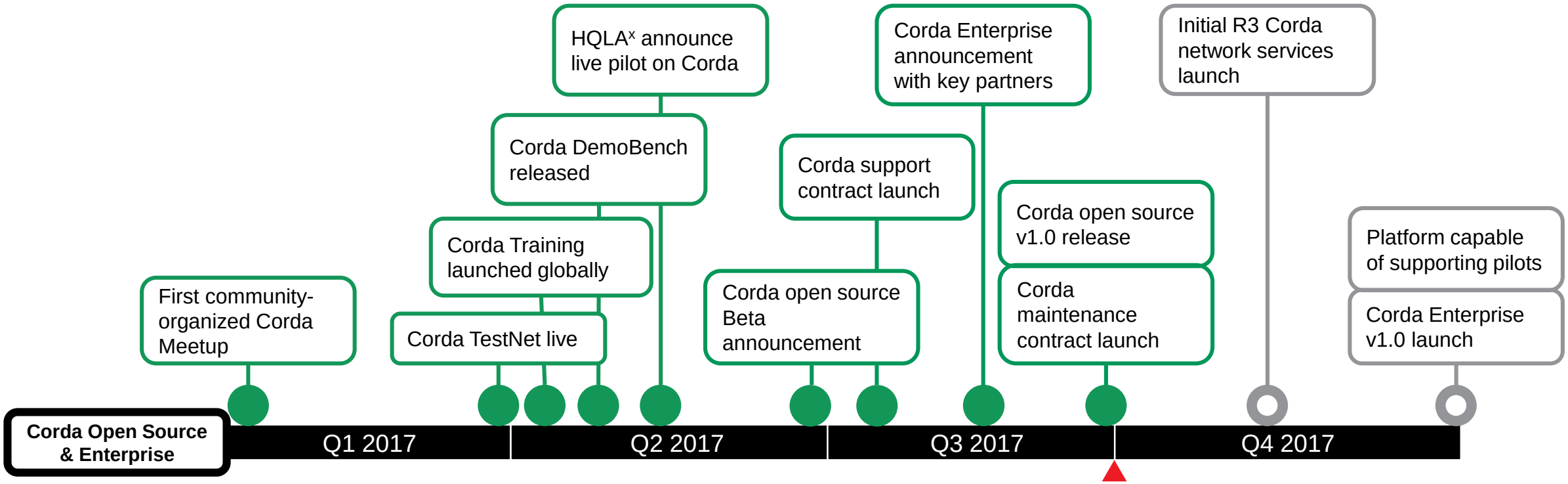
Corda Roadmap

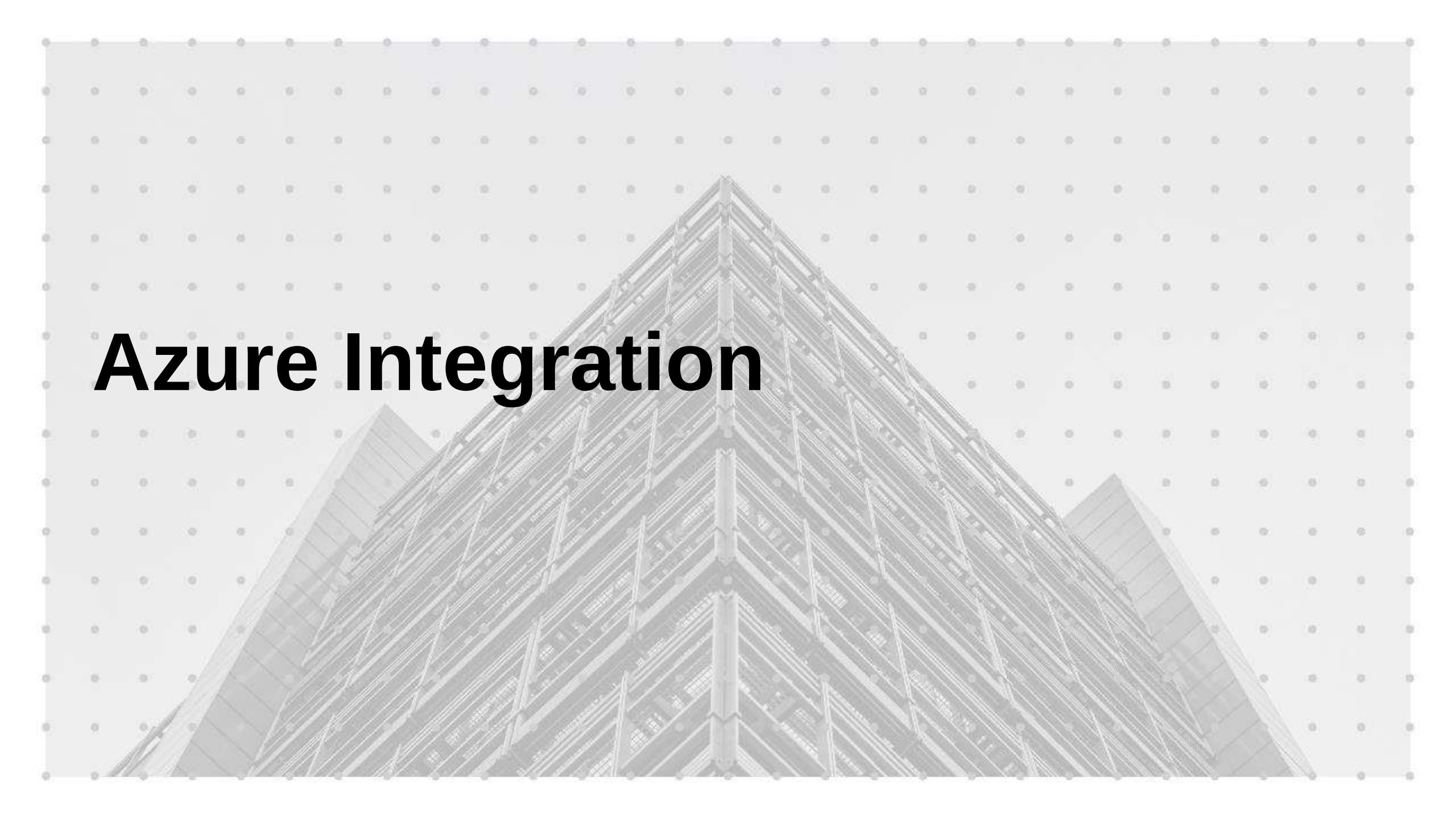
R3 Platform Vision



Corda Timeline 2017

As the largest financial services DLT consortium, R3 is uniquely positioned to enable collaboration and the development of new DLT-based products and services for the financial services industry.



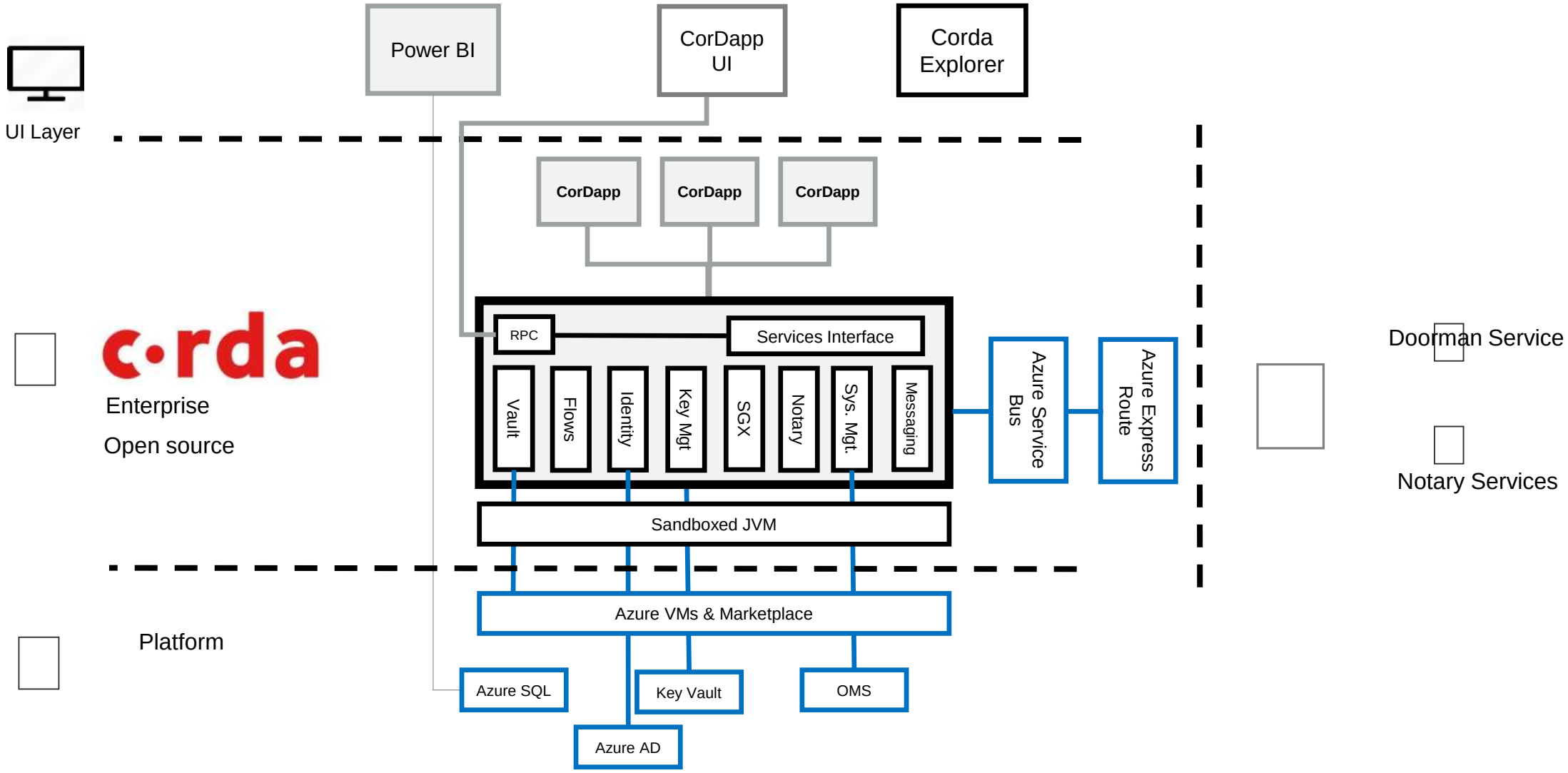


Azure Integration

Partners provide powerful deployment options

- **System integrators** for development and implementation capabilities and experience, domain expertise, and successful project management to completion
- **Microsoft** and R3 have partnered to not only provide Corda on Azure, but Corda is adding capabilities to take advantage of rich features Azure is developing to support blockchain platforms
- **Intel** has partnered with R3 to make Corda capabilities availability within Intel's Software Guard Extensions (SGX) hardware security module
- **HPE** is building Tandem Non-Stop Servers with Corda installed for rapid Blockchain-out-of-a-Box solutions

Azure Proposed Integration



Microsoft Blockchain Vision and Strategy

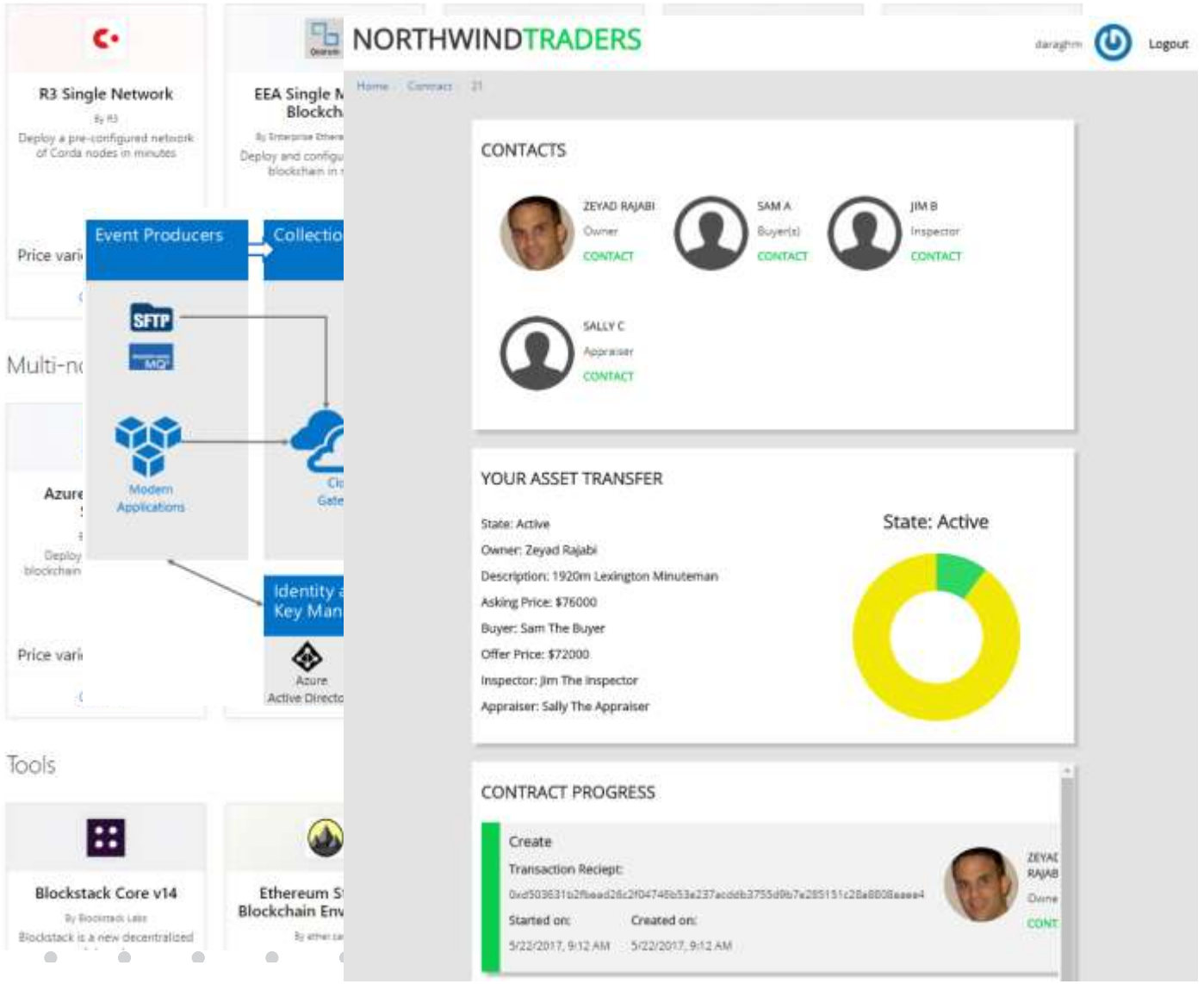
Accelerate Blockchain Deployment

Rapidly Deploy End to End Solutions (Lexington)

Enterprise Integration Capability (Coco Framework)

Rapidly Deploy Blockchain Infrastructure (Bletchley)

Azure Platform / On Premise / Hybrid



For more information

Code and documentation

corda.net – code download, Demobench

docs.corda.net – documentation

github.com/corda/corda

Help

cordaledger.slack.net

stackoverflow.com/questions/tagged/corda

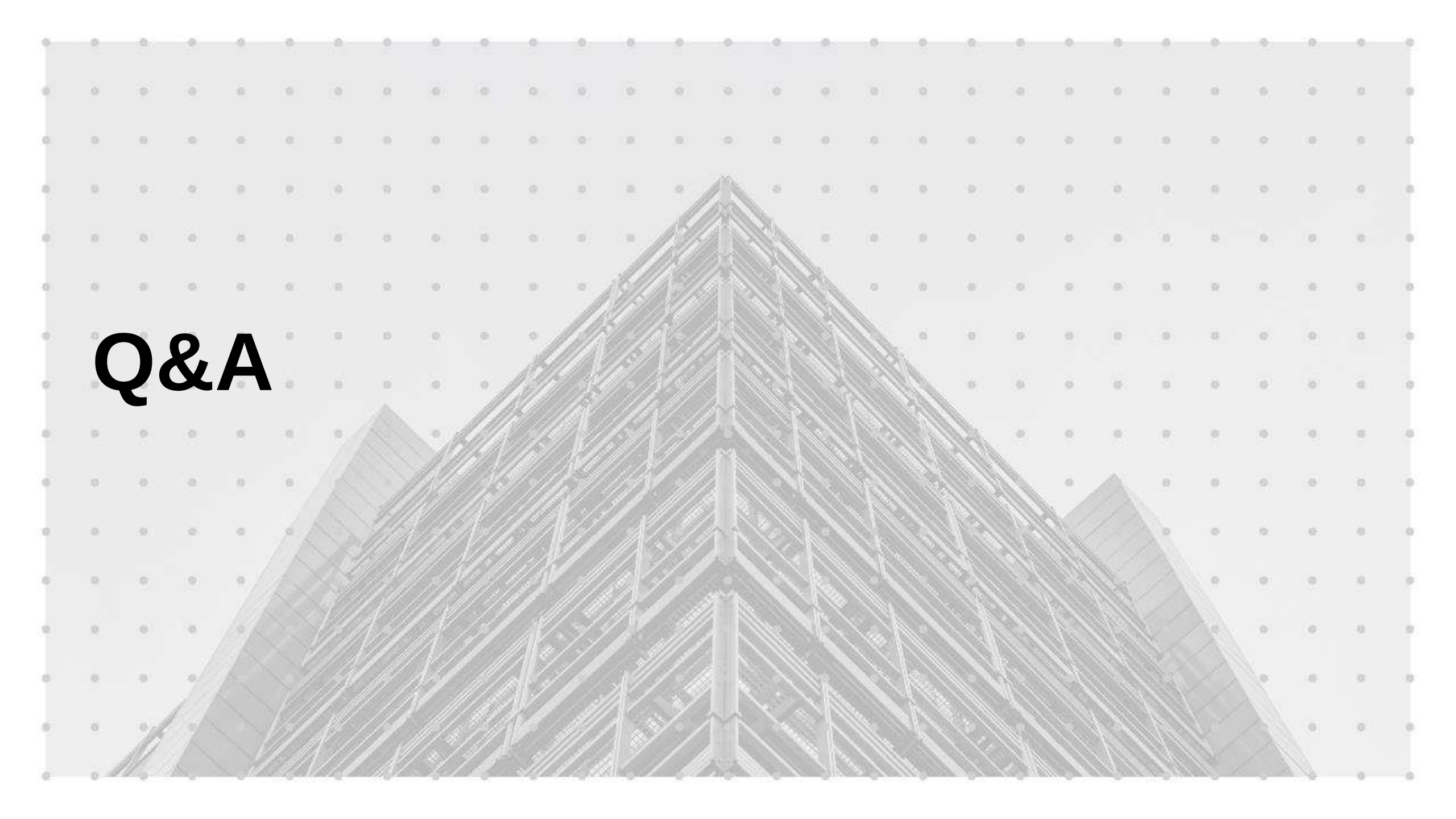
Social

www.meetup.com/New-York-Corda-Meetup/

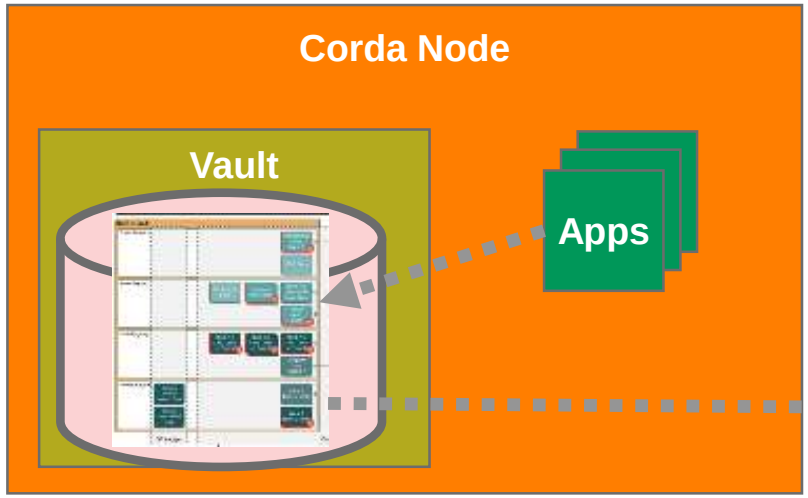
Tom

tom.menner@r3.com

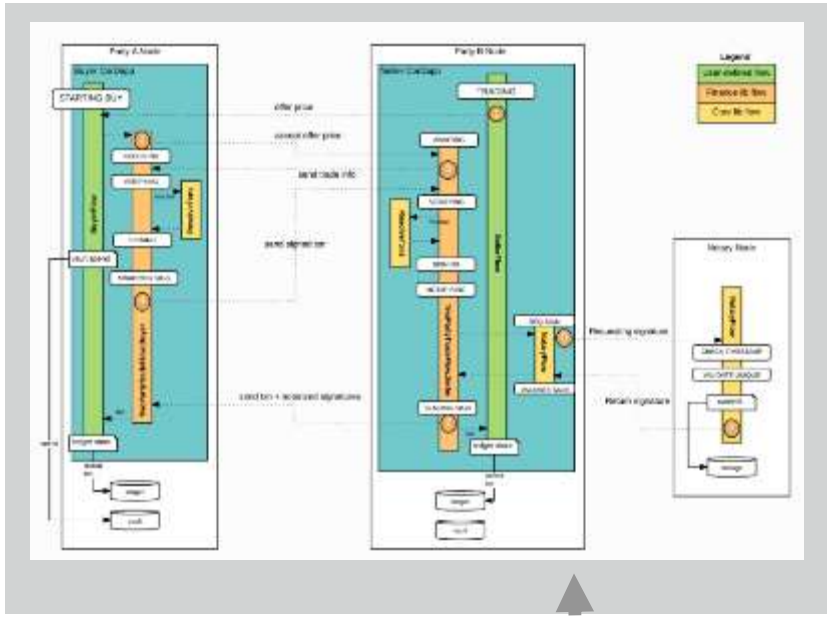
Q&A



Appendix: Corda component architecture



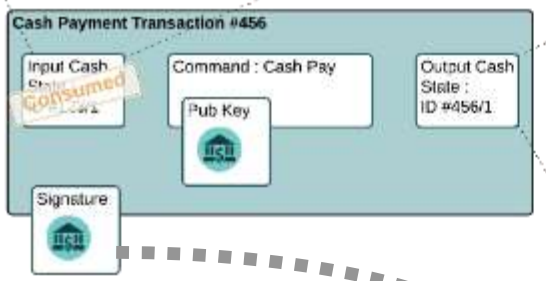
State Object creates properties / fields for the given financial agreement



Contract Code:

Verify that:
Transfer (whole):
- Rule 1 { code }

Transfer (partial):
- Rule 1 { code }
- Rule 2 { code }



Transactions enable transition between states

Flows defined for specific data sent to each node/identity – signatures on transactions