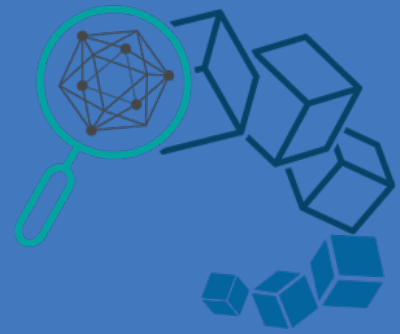
HYPERLEDGER

BLOCKCHAIN TECHNOLOGIES FOR BUSINESS



Hyperledger Fabric Architecture and Design

Baohua Yang

Dec, 2017

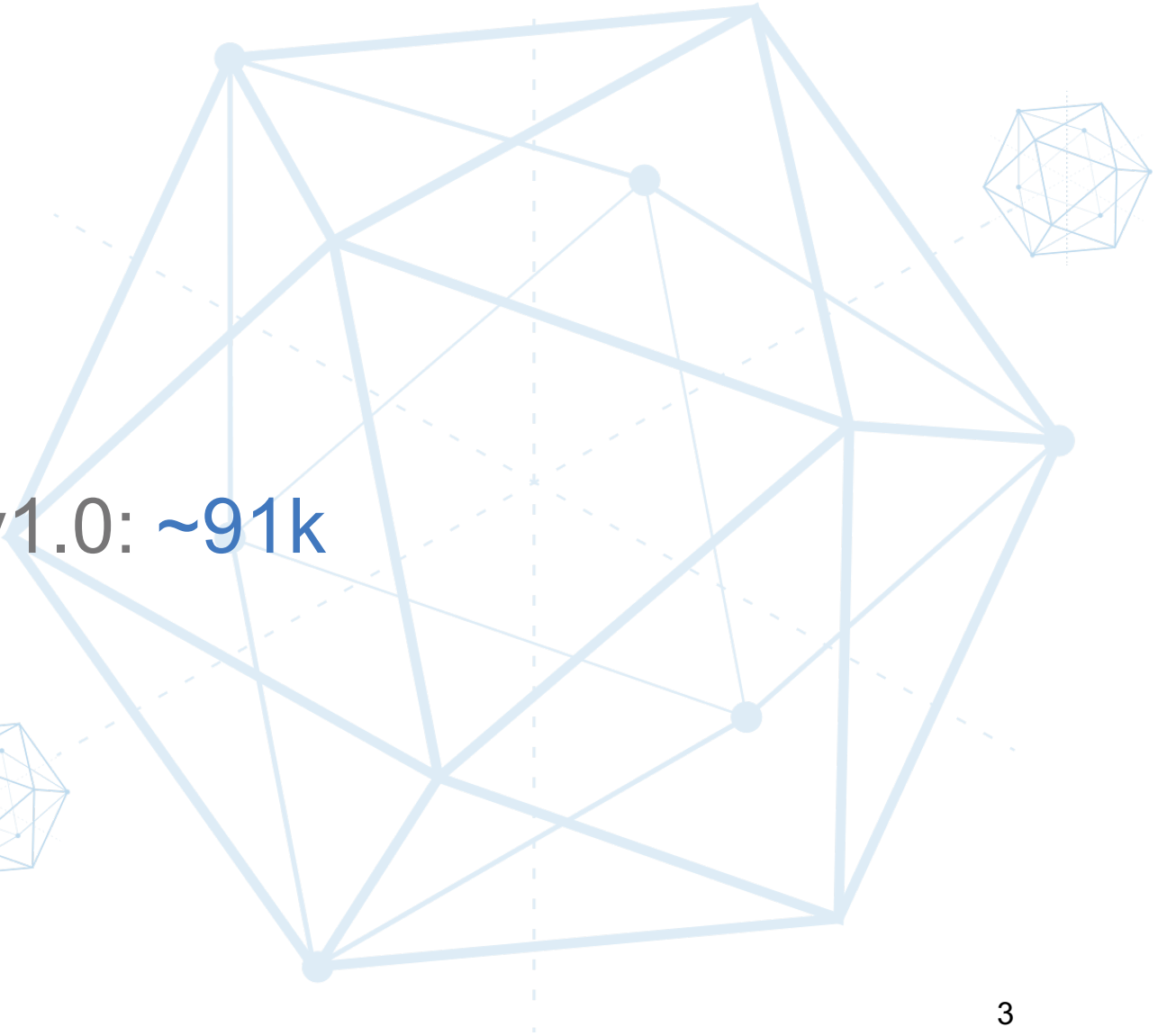
About Me

- **Interested Areas**
  - Fintech, Cloud and Analytics
- **Technical Leader**
  - Senior Researcher/Architect in IBM, Oracle
- **Open-Source Contributor**
  - Hyperledger, OpenStack, OpenDaylight, etc.
- **Hyperledger Developer**
  - Core designer & committer of Fabric, Cello, sdk etc.
  - Hyperledger Technical Steering Committee (TSC) Member
  - Hyperledger Technical Working Group China Chair

Hyperledger Fabric

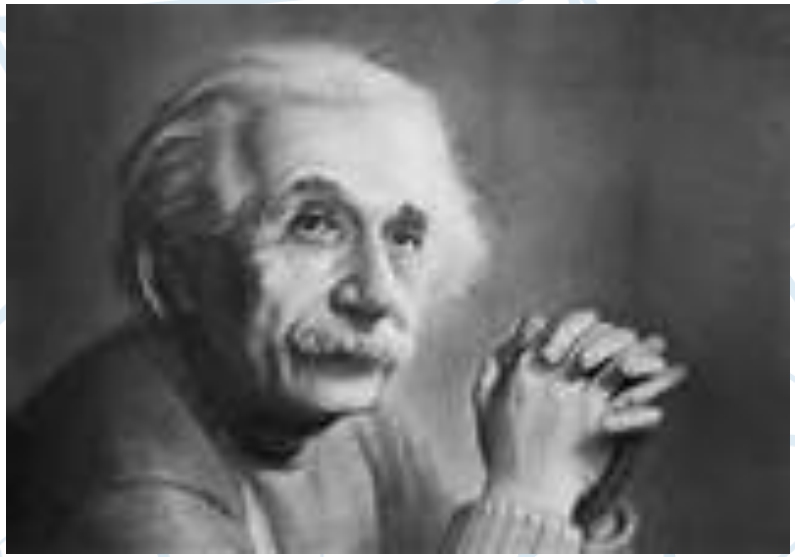
- Open-sourced at Dec, 2015
- Proposed by IBM and DAH
- Written in Golang
- 90+ contributors
- 7000+ commits
- Core code (loc): v0.6: ~49k; v1.0: ~91k

- Active now, in 1.1.0-preview



Existing Blockchain Technologies

- Limited Throughput
- Slow Transaction Confirmation
- Designed for Cryptocurrency
- Poor Governance
- No Privacy
- No Settlement Finality
- Anonymous Processors
- …

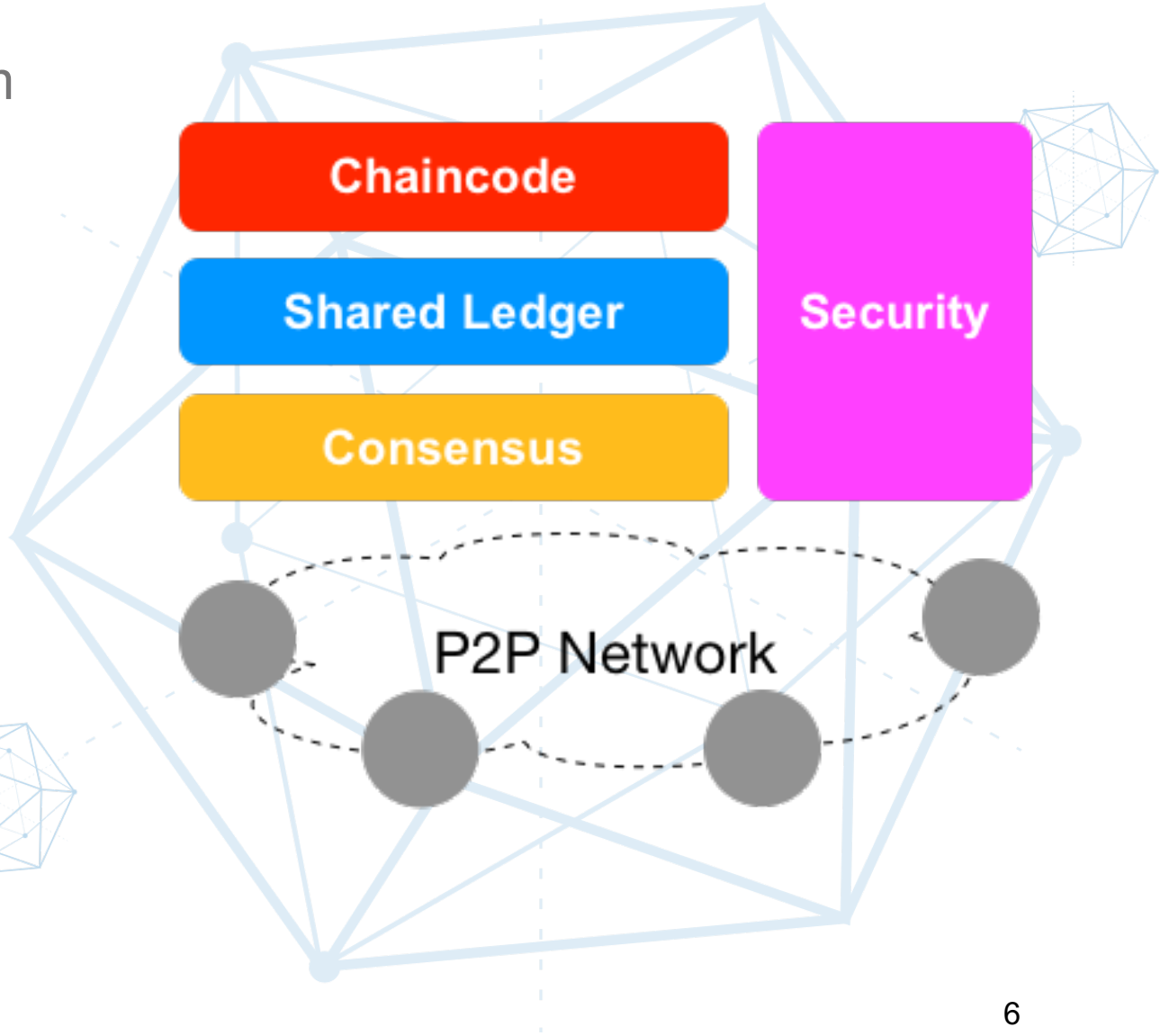


Hyperledger Fabric: Ledger for Enterprise

- Intended as a foundation for developing applications or solutions with a modular architecture, Hyperledger Fabric allows components, such as consensus and membership services, to be plug-and-play
- Privacy, Confidentiality, Auditability, Performance and Scalability
- Permissioned with better trust among members, while enable optimized consensus
- Open protocol/standard with open-source code

Fabric Main Components

- Shared Ledger
  - Append-only distributed record system
  - Blocks + States
- Smart Contract (Chaincode)
  - Business logics with transactions
  - Stateless and deterministic
- Consensus
  - Verified and ordered transactions
- Security
  - Access control
  - Privacy protection
  - Verification
  - CA

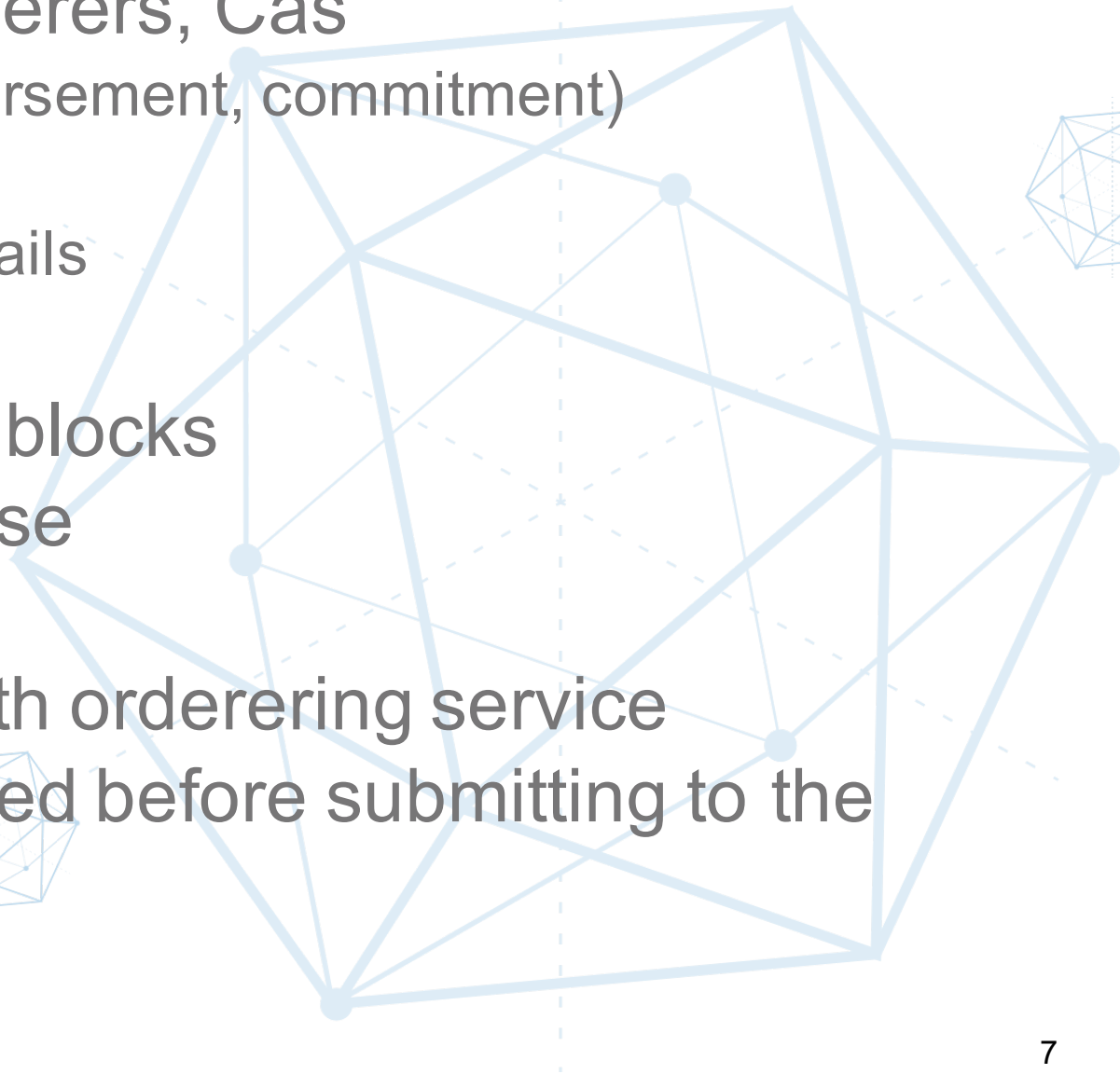


Fabric Basic Functionalities

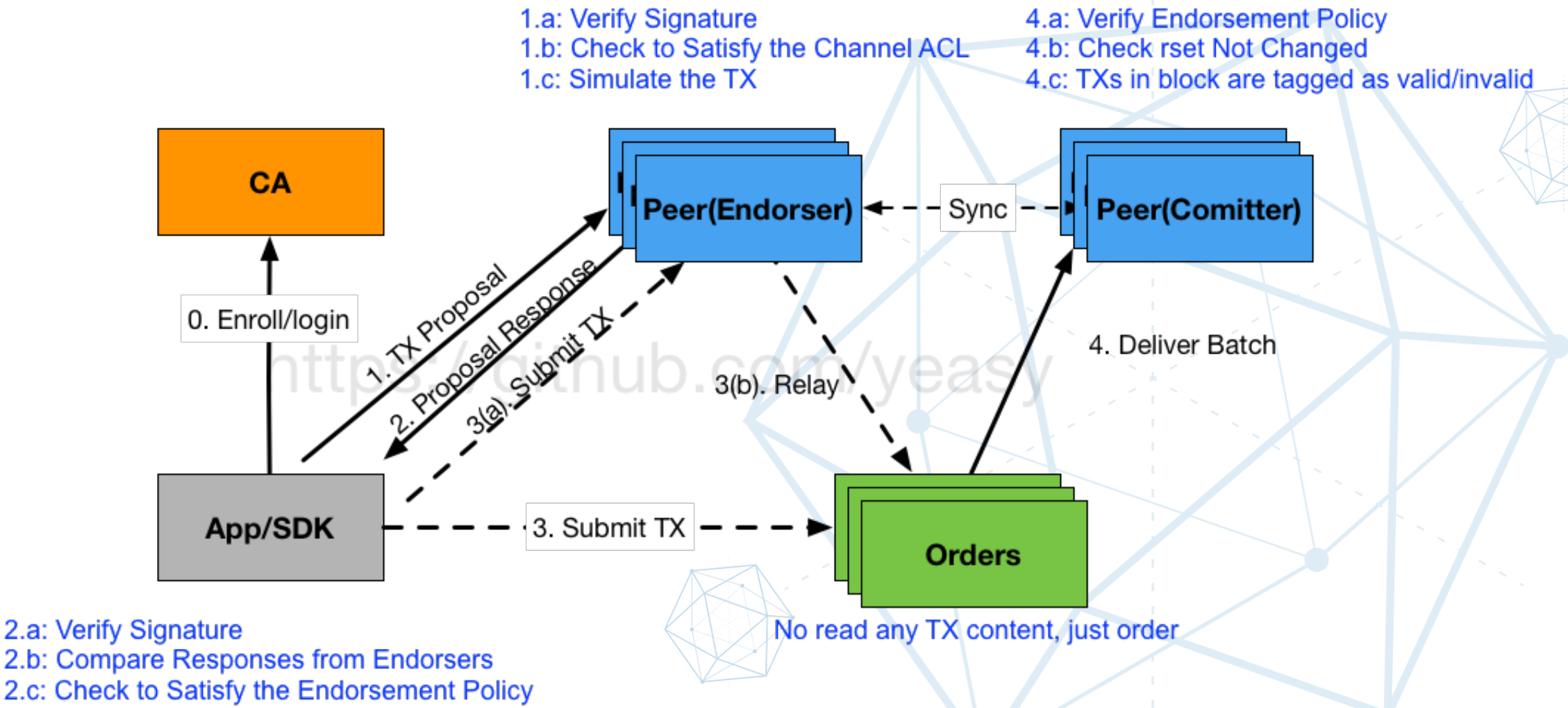
- Network consists of Peers, Orderers, Cas
  - Peer: transaction processing (endorsement, commitment)
  - Orderer: Order transaction
  - CA: Manage identities and credentails

- Transaction history is stored in blocks
- Latest state is stored in database

- Transaction is consensused with orderering service
- Transaction proposal is endorsed before submitting to the network

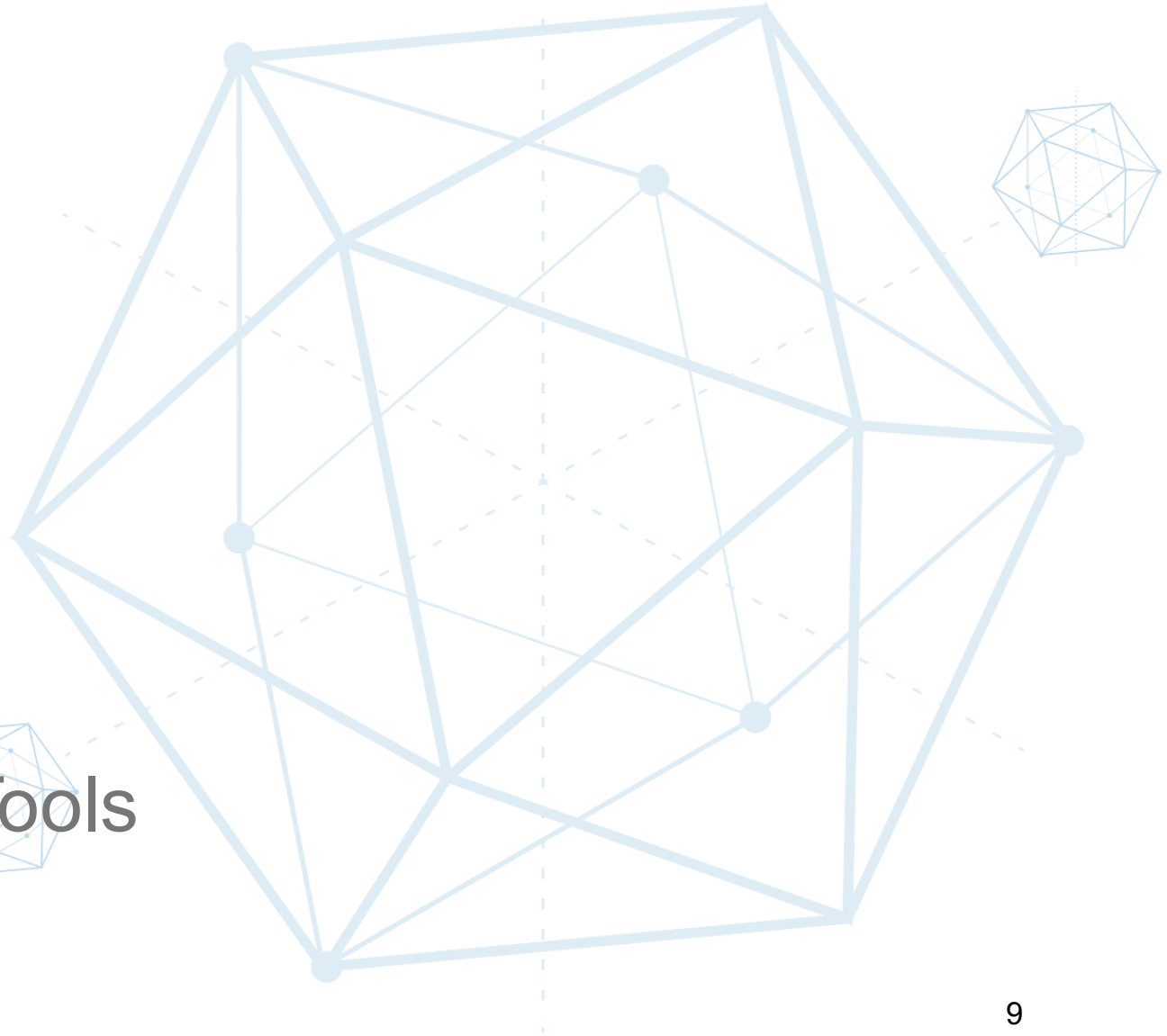


Fabric 1.x Workflow



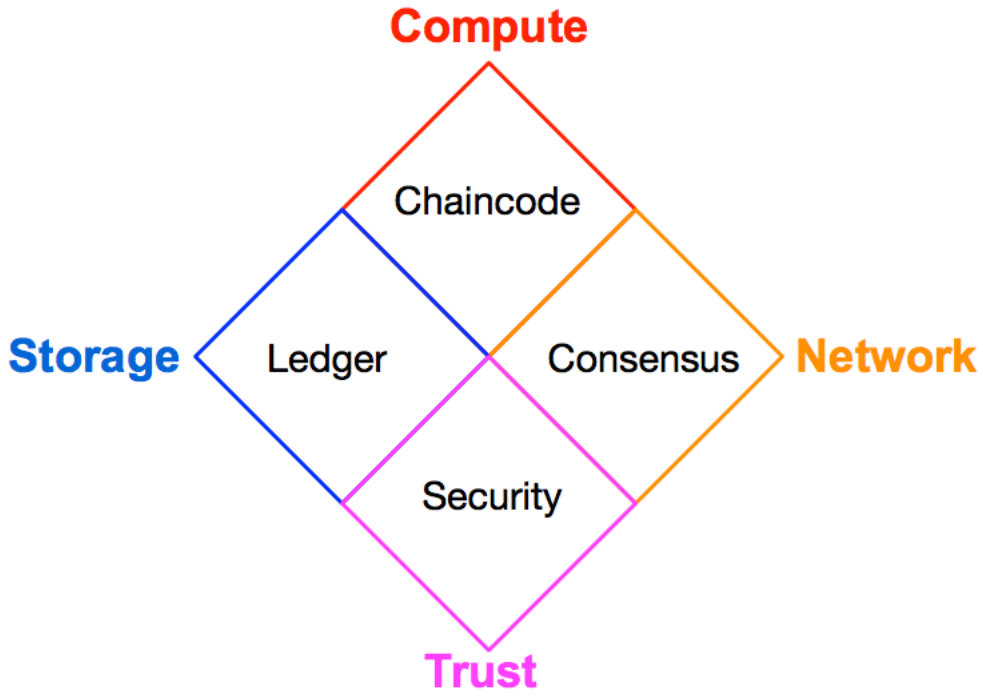
Fabric 1.x New Design

- Node Functionality Decouple
- Multi-Channel/Chain
- Private Ledger
- Consensus
- Permission and Privacy
- System Chaincode
- Pluggable Components
- Configuration Management Tools



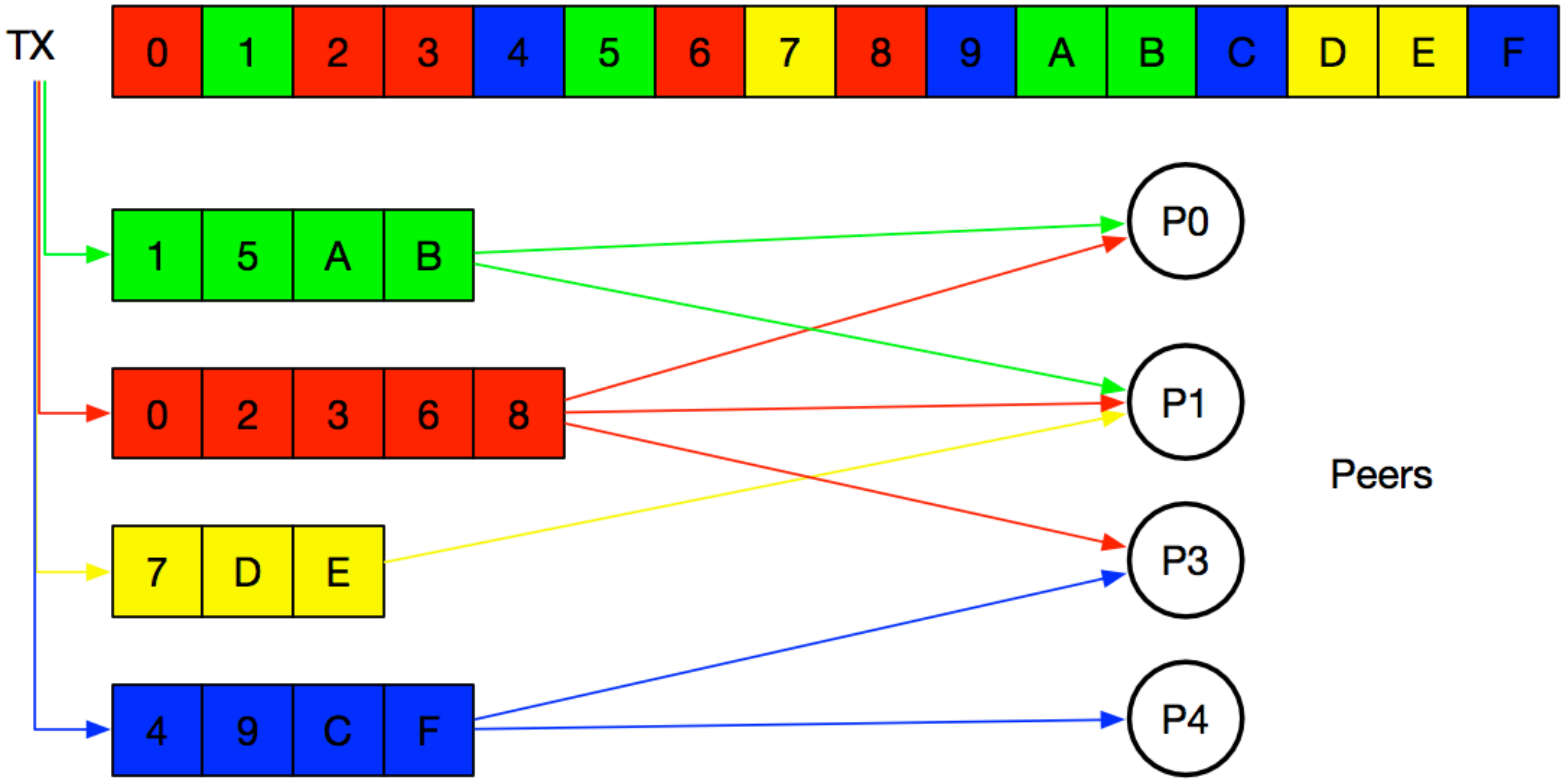
Node Functionality Decouple

- Various Intensive Requirements/Workloads
  - Chaincode: Compute intensive
  - Shared Ledger: Storage intensive
  - Consensus: Network intensive
  - Security: Trust intensive
- Decouple Full-functional Nodes
  - **Endorser**: Endorse TX proposal
  - **Committer**: Write down block
  - **Orderer**: Only order, no TX aware
  - **CA**: Certificate management



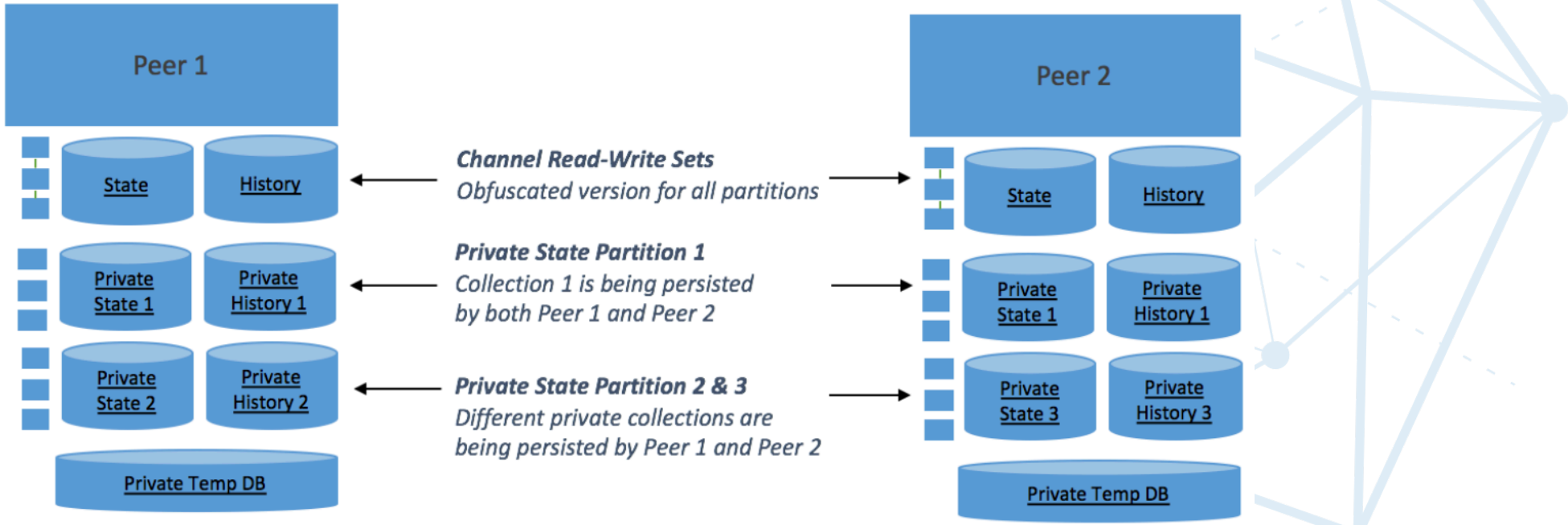
Multi-Channel/Chain

- Isolate the transactions, ledgers between organizations – Overlay Network
- Peer can join channels accordingly



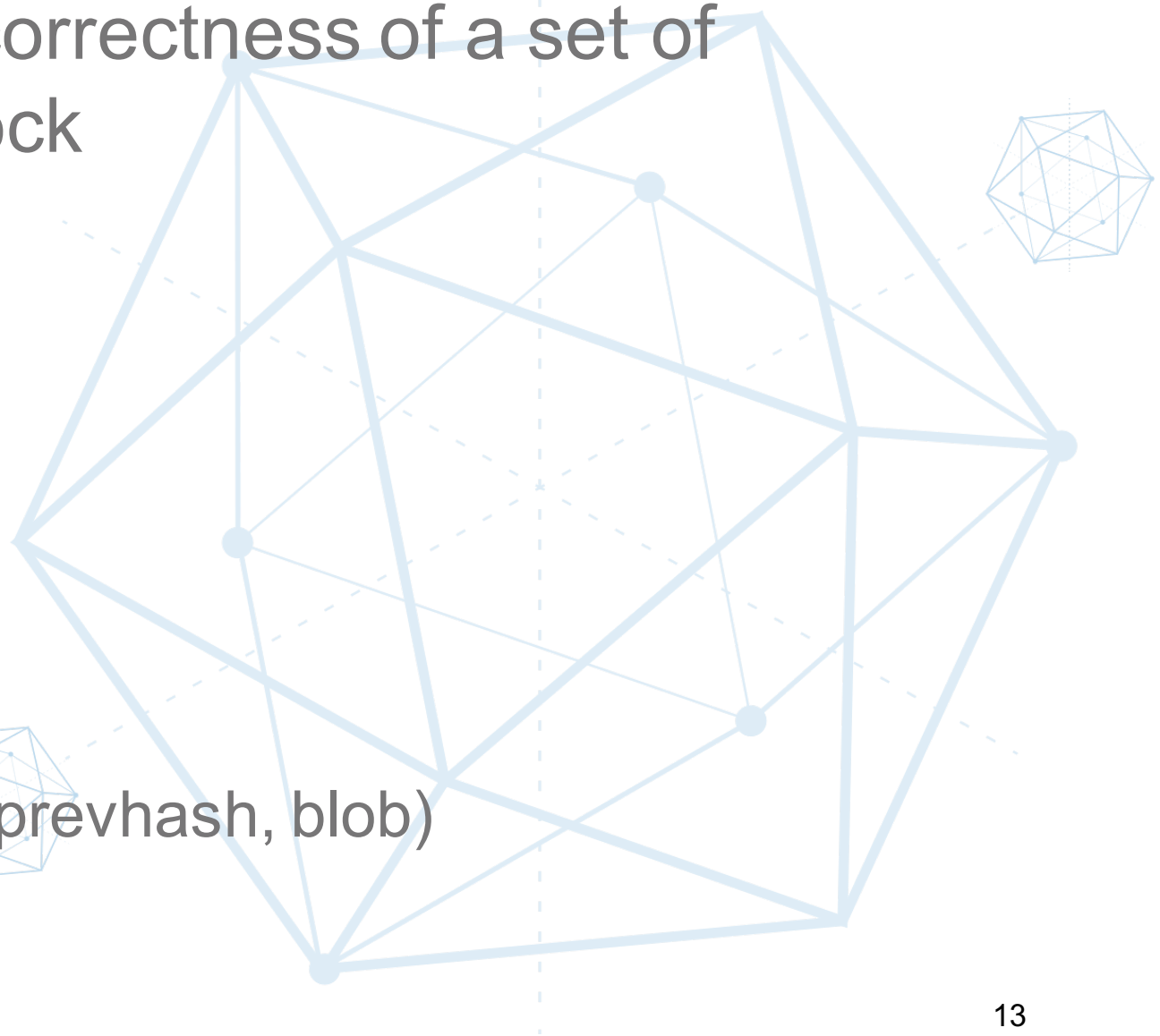
Permission and Privacy

- Private Ledger
  - Full data needn't send to orderer
  - Peers can have private transaction even in the same channel



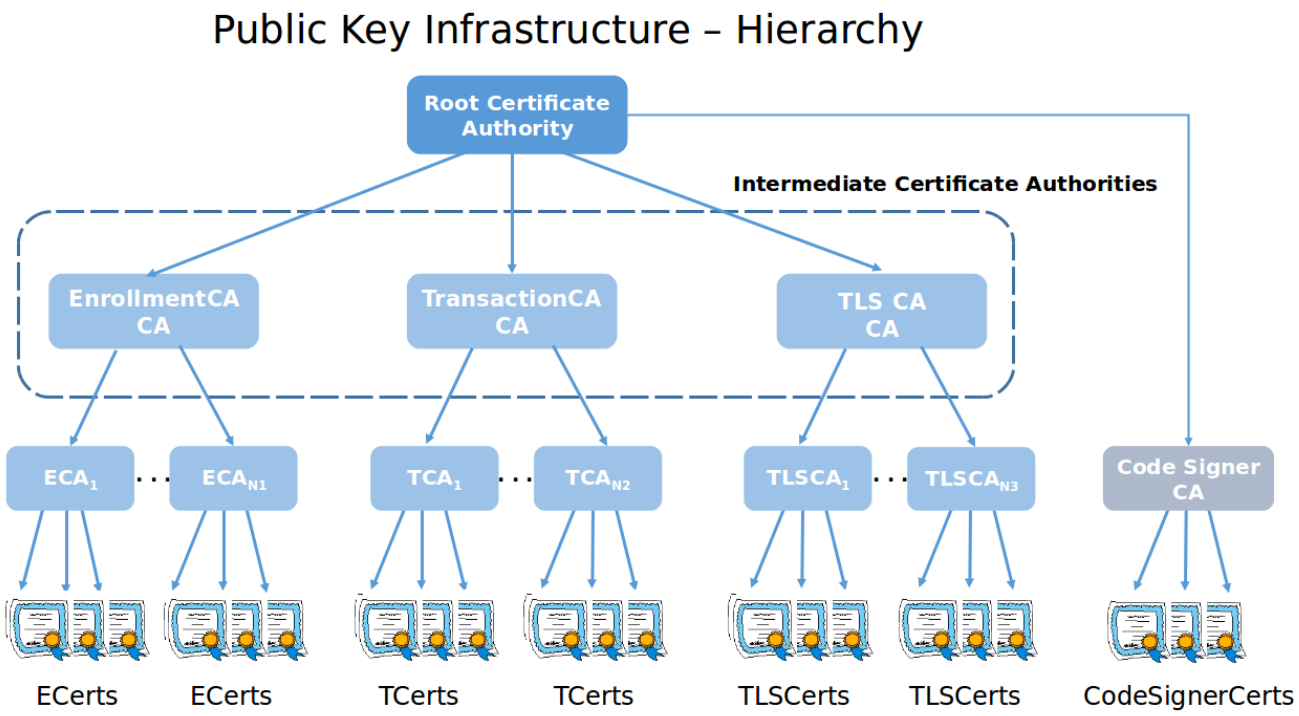
Consensus

- Full-circle verification of the correctness of a set of transactions comprising a block
  - Endorsement policy
  - MVCC validation on RW sets
  - Ordering
  - ACL
- Orderer
  - Solo, Kafka, BFT, and more…
  - Broadcast(blob), Deliver(seqno, prevhash, blob)



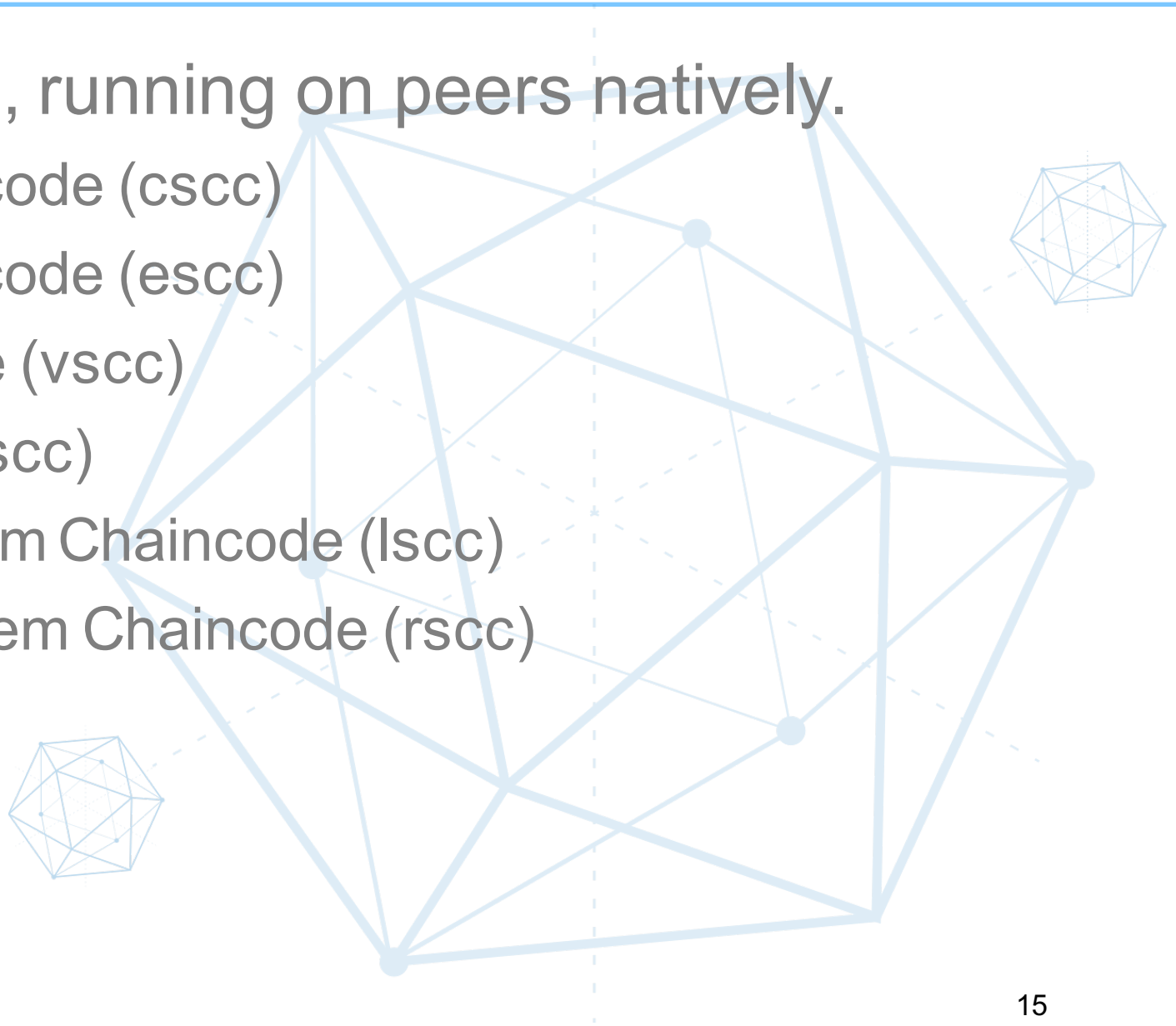
Permission and Privacy

- Permission at Various Levels
  - Network, channel, transaction
- Privacy for Business
  - Anonymity
  - Un-linkability
  - Auditability and Accountability
- Fabric CA (PKI)
  - Identity Registration Management
  - Enrollment Cert (Ecert) and Transaction Cert (Tcert)



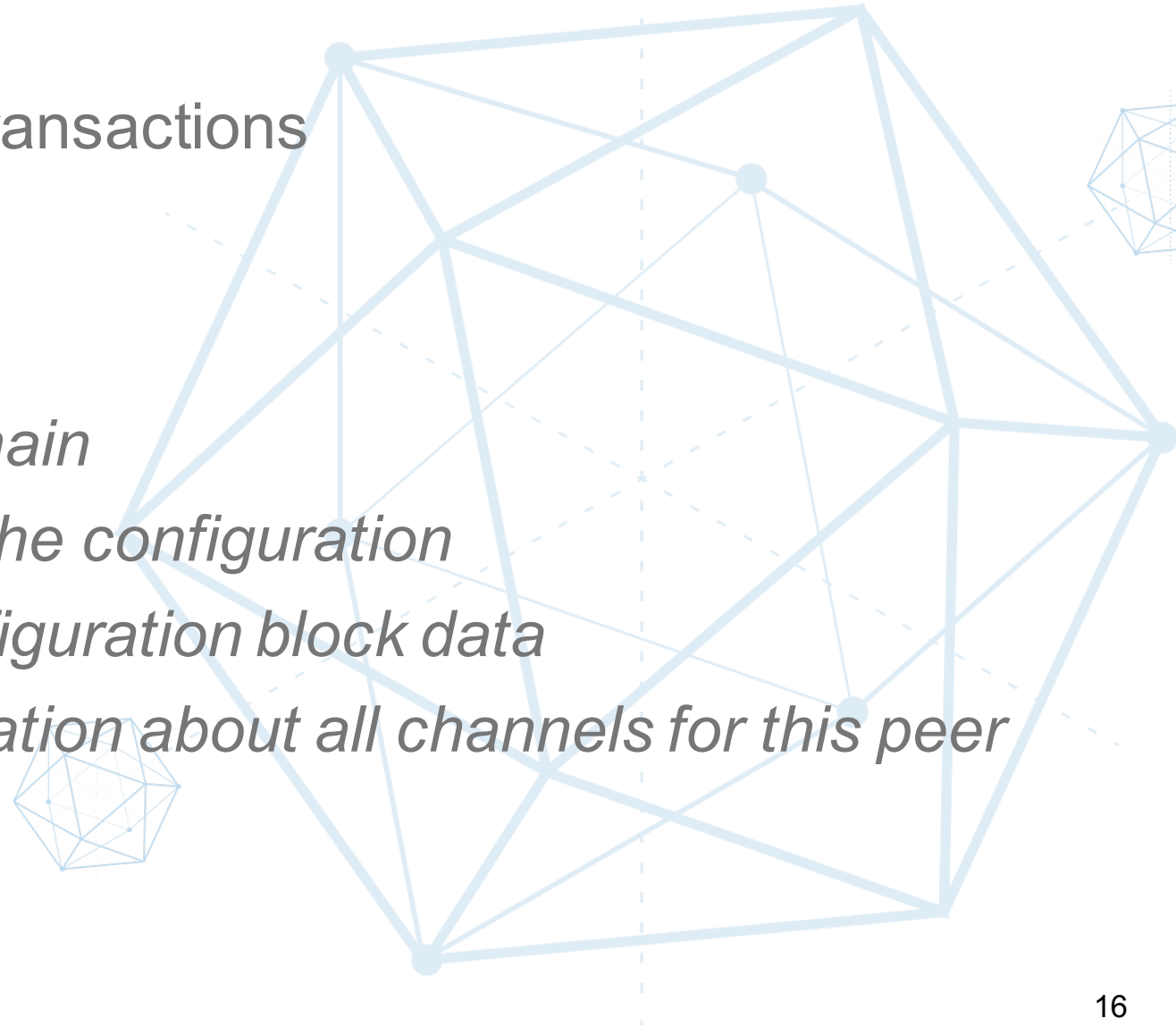
System Chaincode

- Handle system operations, running on peers natively.
  - Configuration System Chaincode (cscc)
  - Endorsement System Chaincode (escc)
  - Validation System Chaincode (vscc)
  - Query System Chaincode (qscc)
  - Lifecycle management System Chaincode (lscc)
  - Resource management System Chaincode (rscc)



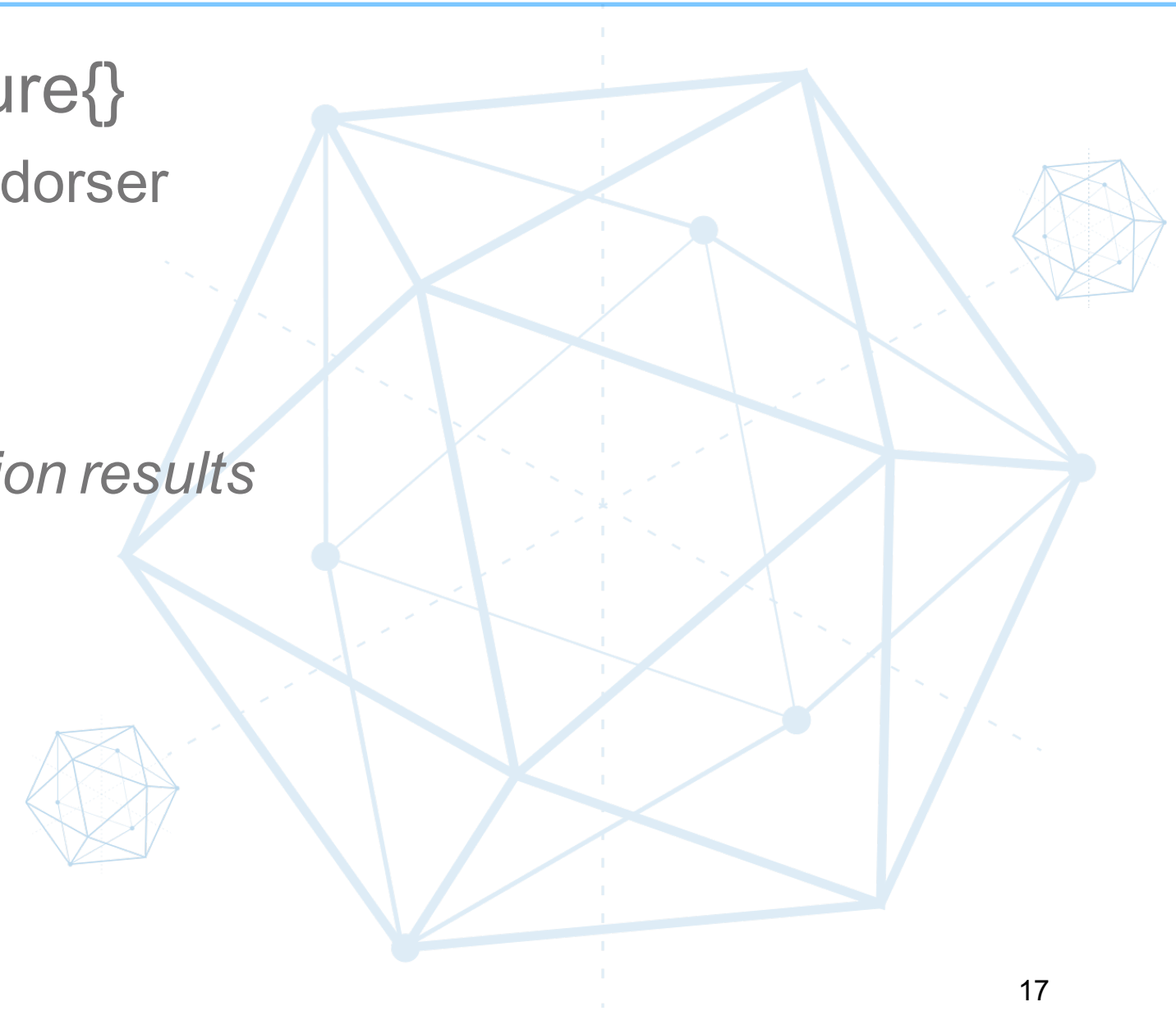
Configuration System ChainCode

- PeerConfiger{}
  - Handle those configuration transactions
- Init()
- Invoke()
  - JoinChain: *peer join into a chain*
  - UpdateConfigBlock: *update the configuration*
  - GetConfigBlock: *get the configuration block data*
  - GetChannels: *returns information about all channels for this peer*



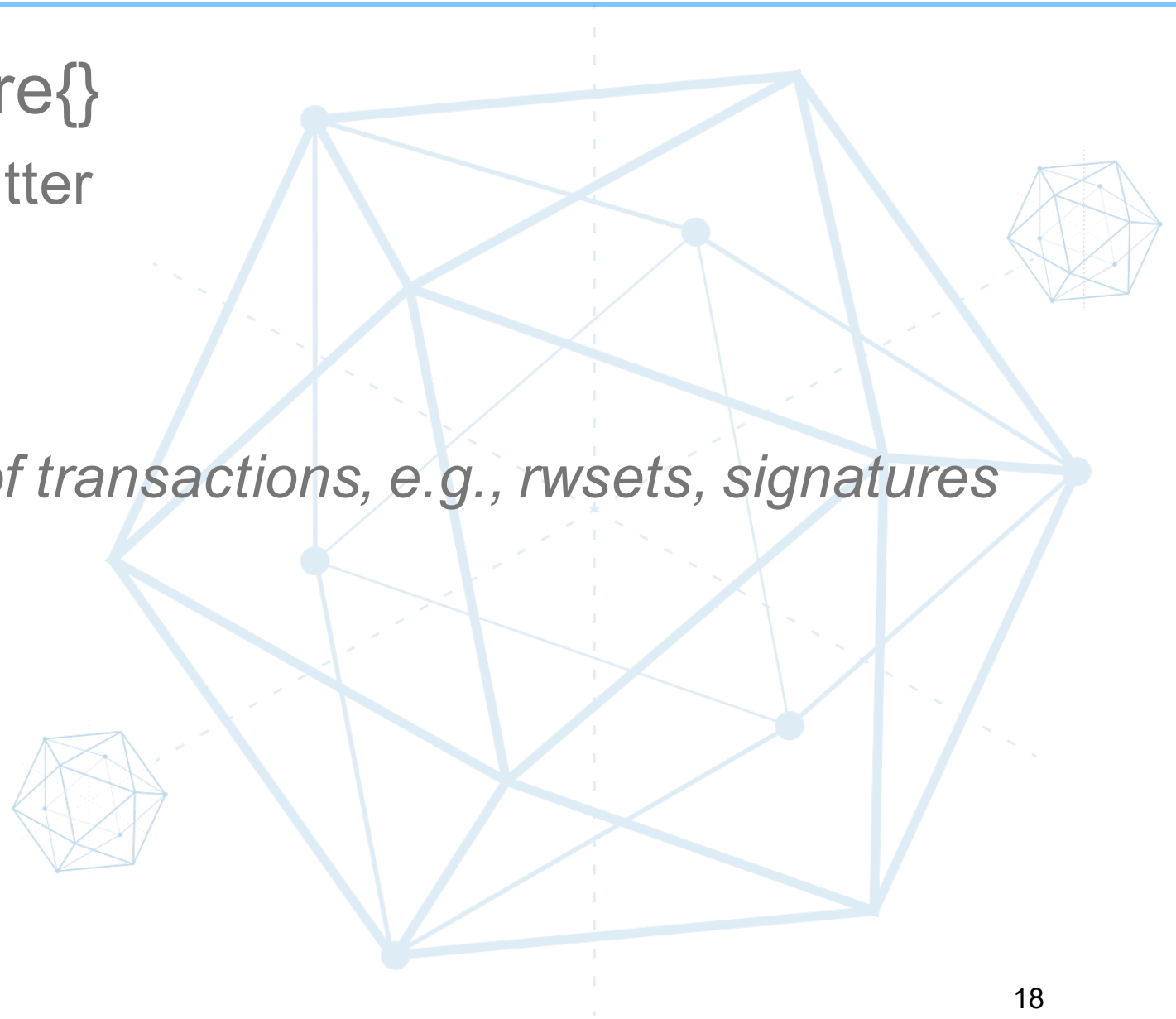
Endorsement System ChainCode

- EndorserOneValidSignature{}
  - Endorsement process on Endorser
- Init()
- Invoke()
  - *Sign on chaincode's simulation results*
  - *More explicit rules (TBD)*



Validation System ChainCode

- ValidatorOneValidSignature{}
  - Validation process on Committer
- Init()
- Invoke()
  - *Validate the specified block of transactions, e.g., rwsets, signatures*

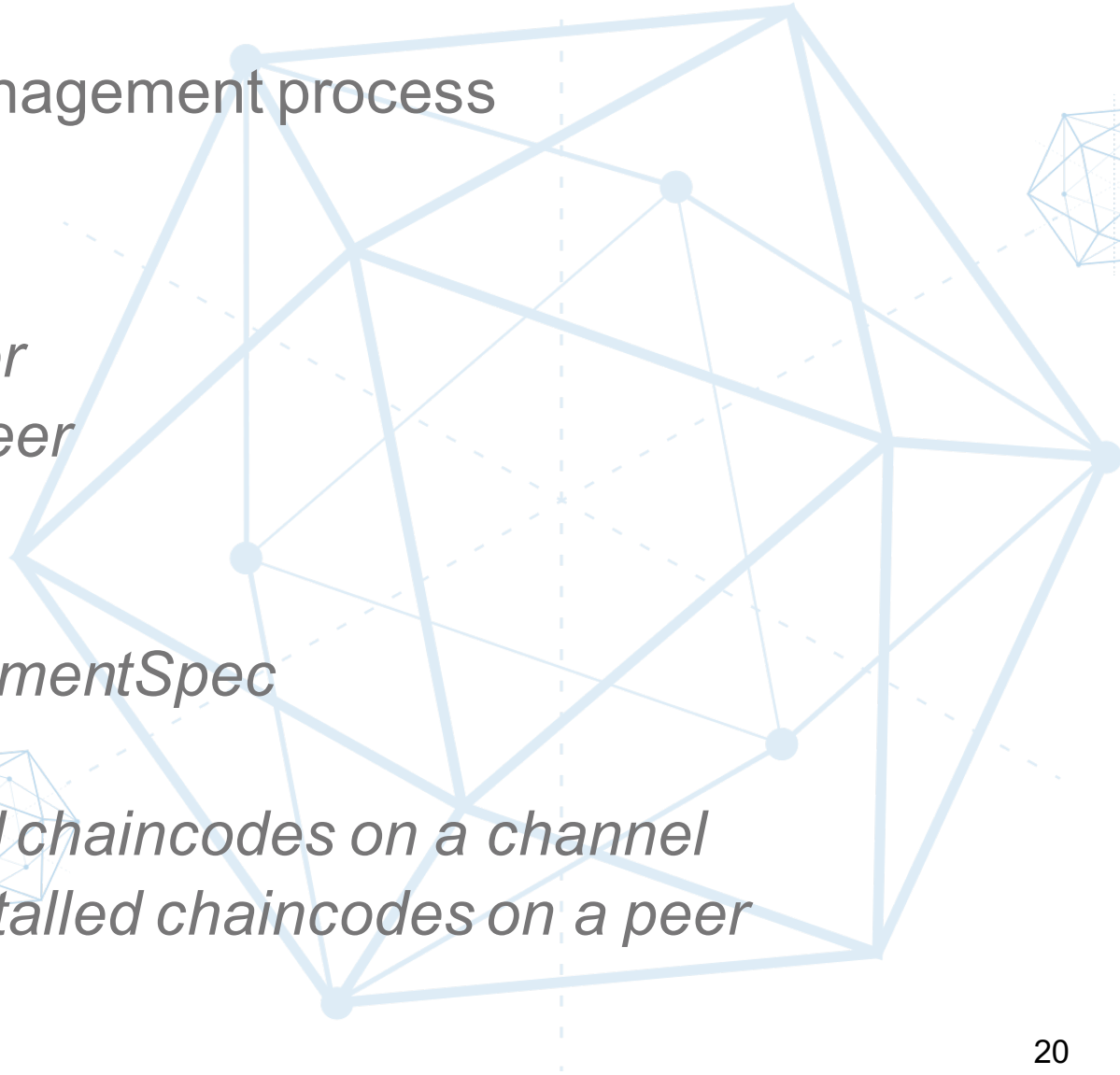


Query System ChainCode

- LedgerQuerier{}
  - Ledger query functions
- Init()
- Invoke()
  - *GetChainInfo: Get information of a chain*
  - GetBlockByNumber: Get the block data by its number
  - GetBlockByHash: Get the block data by its hash value
  - GetTransactionByID: Get the transaction data by its id
  - GetBlockByTxID: Get the block data by contained transaction id

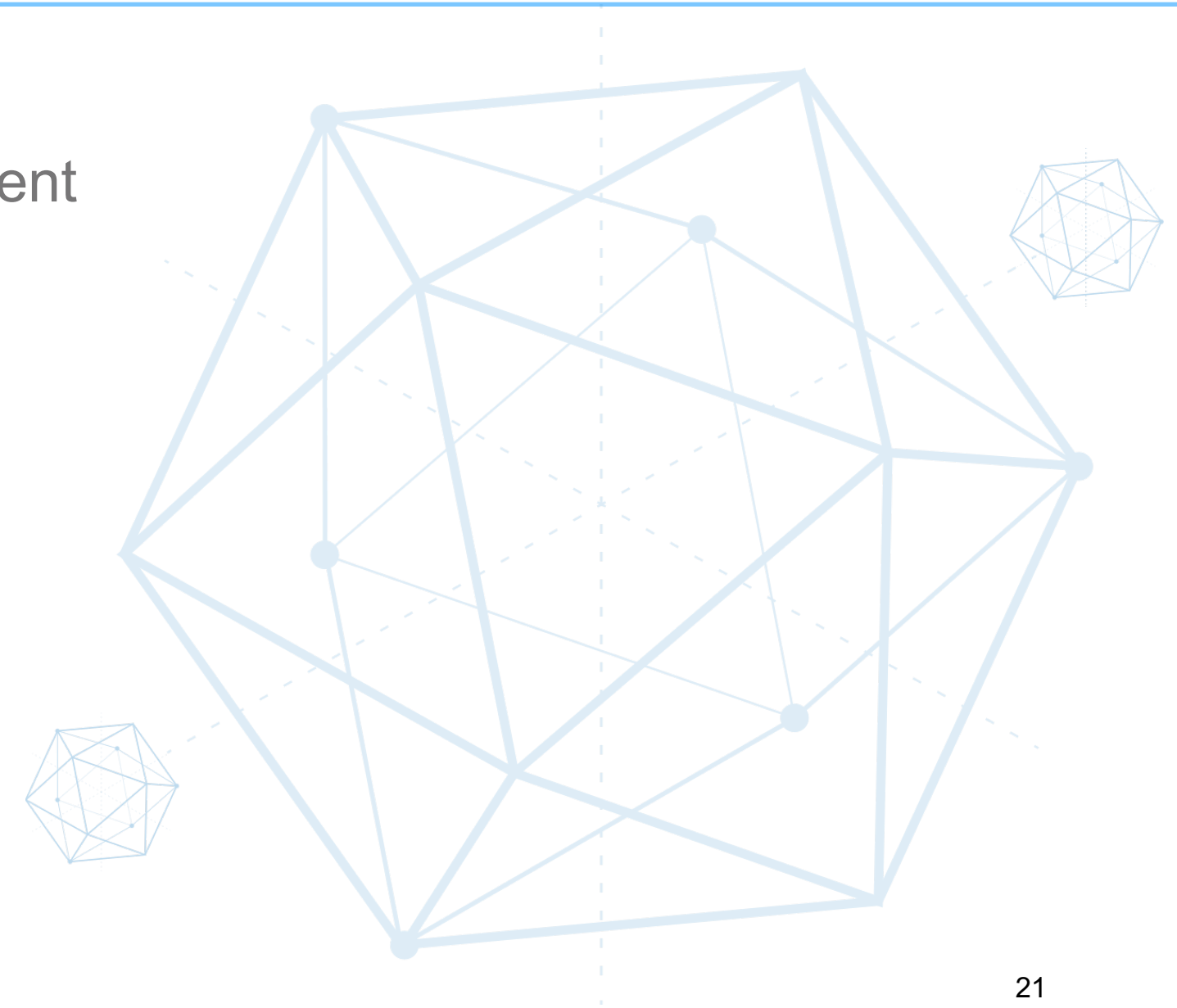
Lifecycle management System ChainCode

- LifeCycleSysCC{}
  - Application chaincode lifecycle management process
- Init()
- Invoke()
  - install: *install a chaincode on a peer*
  - deploy: *deploy a chaincode on a peer*
  - upgrade: *upgrade a chaincode*
  - getid: *get chaincode info*
  - getdepspec: *get ChaincodeDeploymentSpec*
  - getccdata: *get ChaincodeData*
  - getchaincodes: *get the instantiated chaincodes on a channel*
  - getinstalledchaincodes: *get the installed chaincodes on a peer*



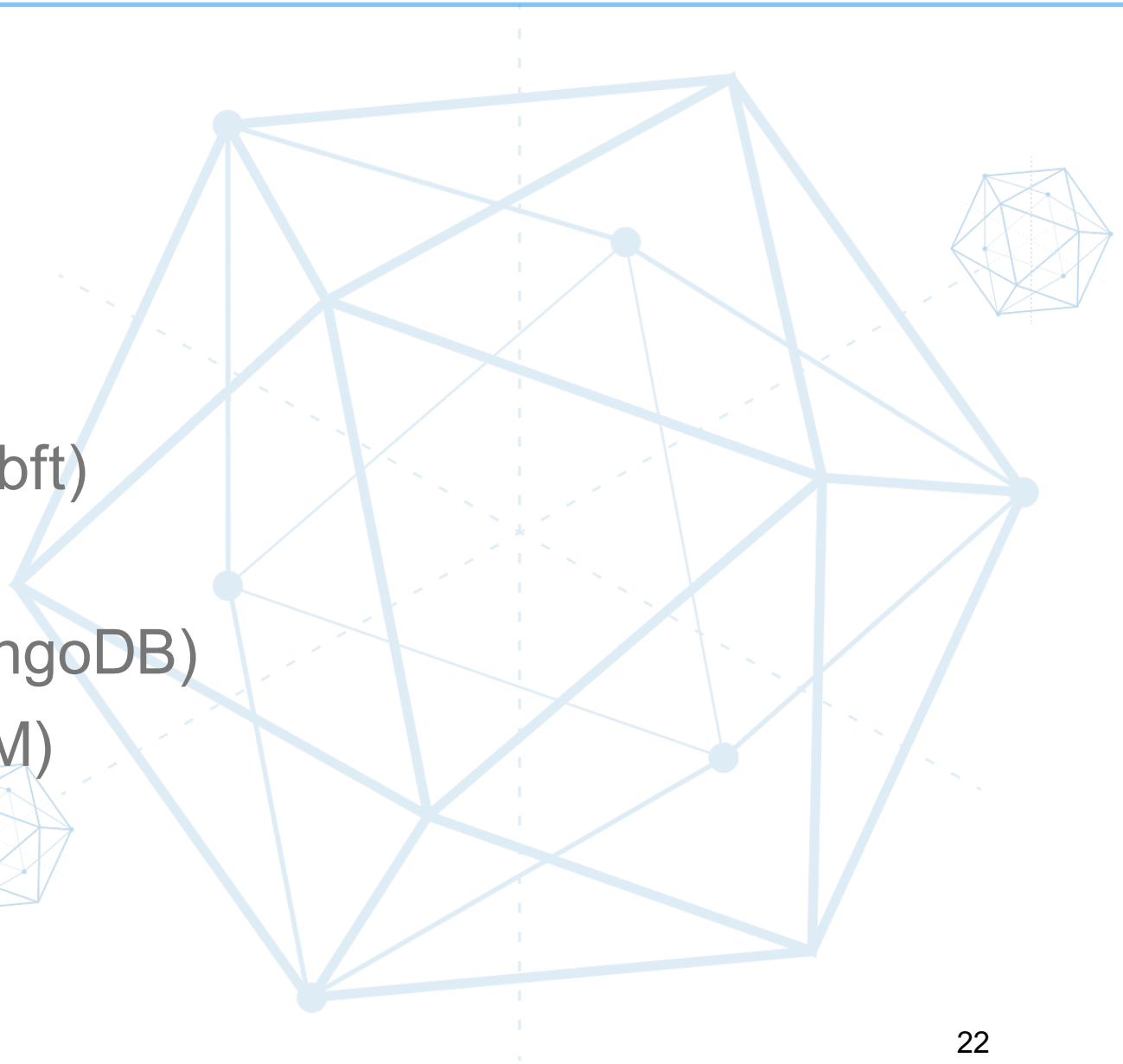
Resource management System ChainCode

- Rscc{}
  - Resource’s policy management
- Init()
- Invoke()
  - TBD



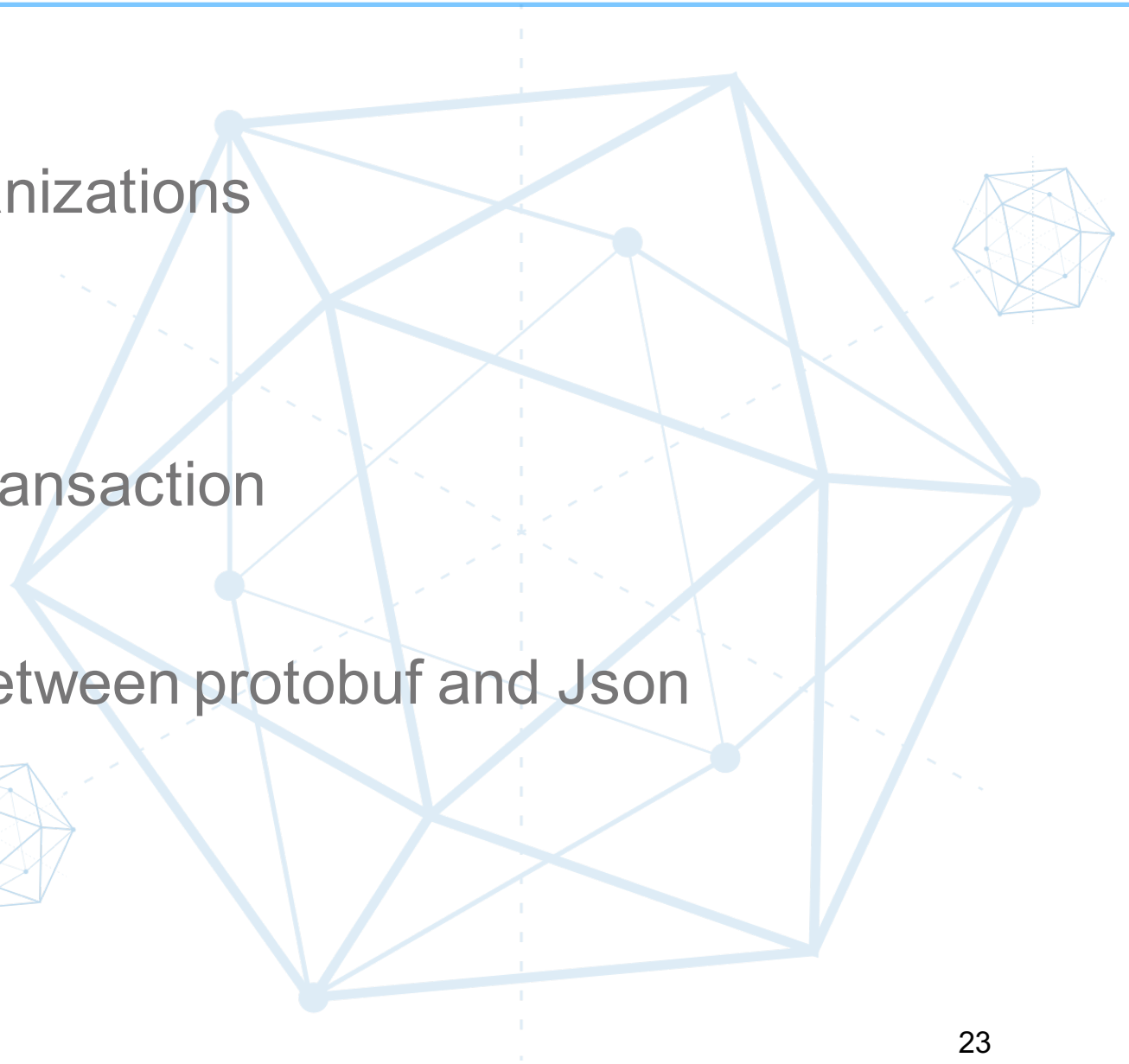
Pluggable Components

- Modular and Pluggable
  - Membership Services (CA)
  - SDKs (node, python, java, go)
  - Endorsement/Verification
  - Consensus service (solo, kafka, bft)
  - Ledger
  - StateDB (levelDB, couchDB, MongoDB)
  - Crypto algorithms (software, HSM)

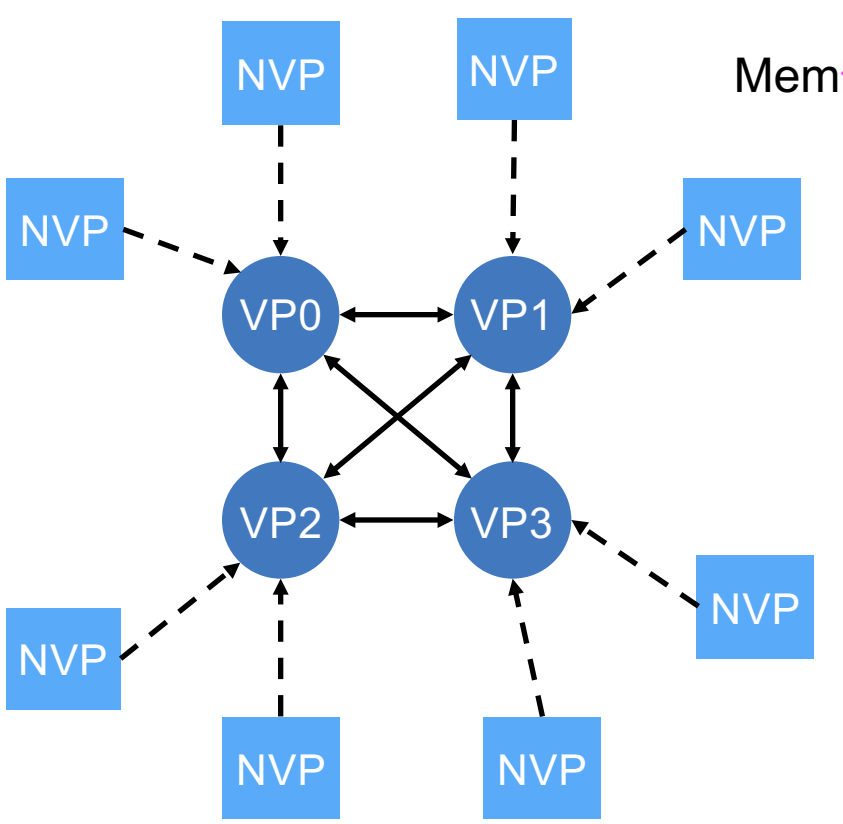


Configuration Management Tools

- Cryptogen
  - Generate certificate files for organizations
- Configtxgen
  - Generate genesis block
  - Generate configuration update transaction
- Configtxlator
  - Convert configuration artifacts between protobuf and Json

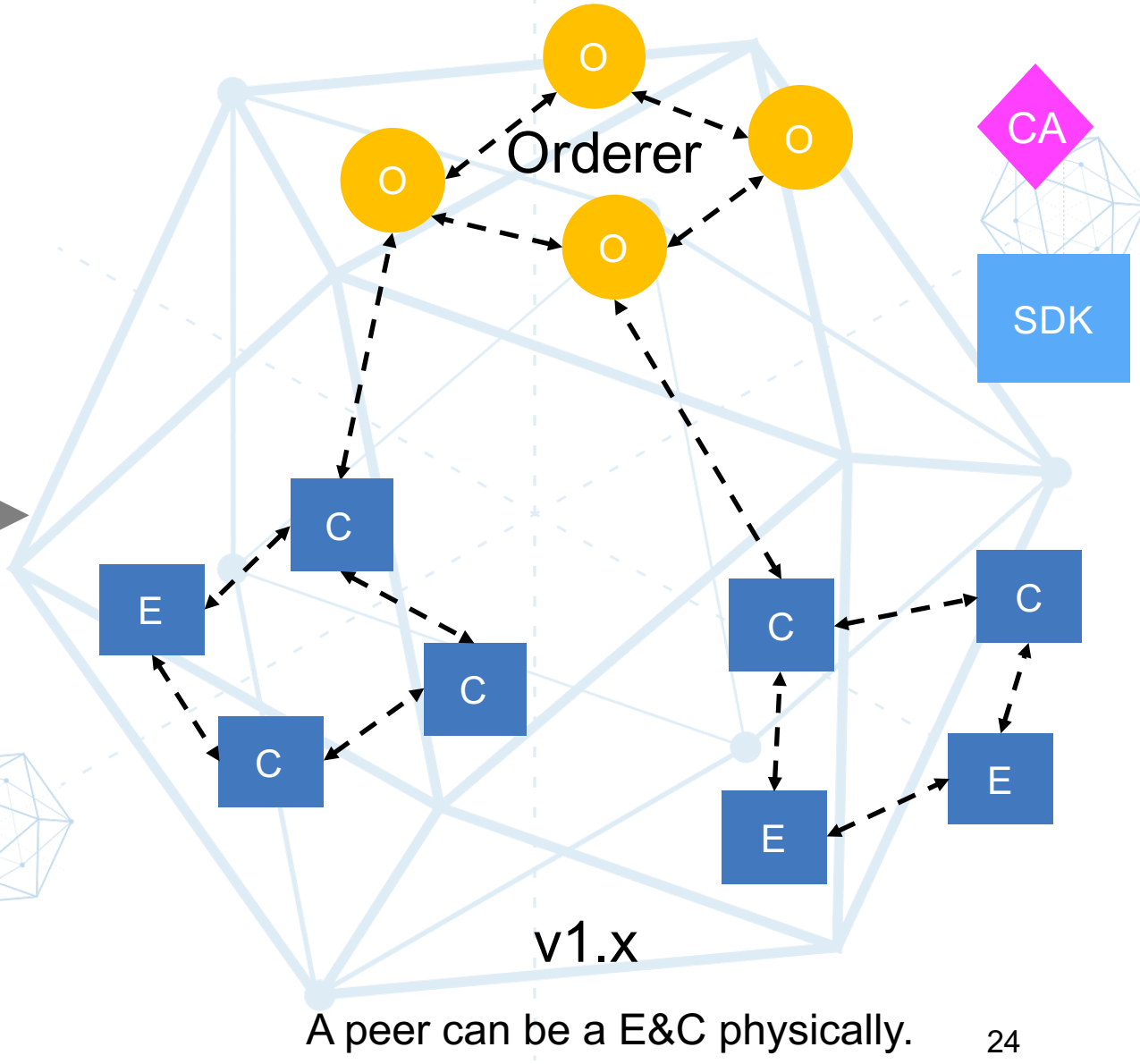


Fabric 1.x Deployment Scenarios



Member Service

v0.6



v1.x

A peer can be a E&C physically.

Hyperledger Fabric Roadmap

Hack Fest docker images

- **60 participates tested**
- Basic v1 architecture in place
- Add / Remove Peers
- Channels
- Node SDK
- Go Chaincode
- Ordering Solo
- Fabric CA

V1 Alpha *

- Docker images
- Tooling to bootstrap network
- Fabric CA or bring your own
- Java and Node SDKs
- Ordering Services - Solo and Kafka
- Endorsement policy
- Level DB and Couch DB
- Block dissemination across peers via Gossip

V1 GA *

- Hardening, usability, serviceability, load, operability and stress test
- Java Chaincode
- Chaincode ACL
- Chaincode packaging & LCI
- Pluggable crypto
- HSM support
- Consumability of configuration
- Next gen bootstrap tool (config update)
- Config transaction lifecycle
- Eventing security
- Cross Channel Query
- Peer management APIs
- Documentation

V Next *

- SBFT
- Archive and pruning
- System Chaincode extensions
- Side DB for private data
- Application crypto library
- Dynamic service discovery
- REST wrapper
- Python SDK
- Identity Mixer (Stretch)
- Tcerts

2016/17 December | March | June | Future

Connect-a-thon

- 11 companies in Australia, Hungary, UK, US East Coast, US West Coast, Canada dynamically added peers and traded assets

Connect-a-cloud

- Dynamically connecting OEM hosted cloud environments to trade assets

HYPERLEDGER
BLOCKCHAIN TECHNOLOGIES FOR BUSINESS

*** Dates for Alpha, Beta, and GA are determined by Hyperledger community and are currently proposals.**

Proposed Alpha detailed content:
https://wiki.hyperledger.org/projects/proposedv1alphacontent

Reference

- Hyperledger Project
- 《区块链原理设计与应用》
- 《Docker 技术入门与实战》
- 超级账本 Fabric 源码剖析
- github.com/yeasy/blockchain_guide

HYPERLEDGER

BLOCKCHAIN TECHNOLOGIES FOR BUSINESS

Questions?

Thank You!
@baohua

Slides available at github.com/yeasy/seminar-talk#hyperledger