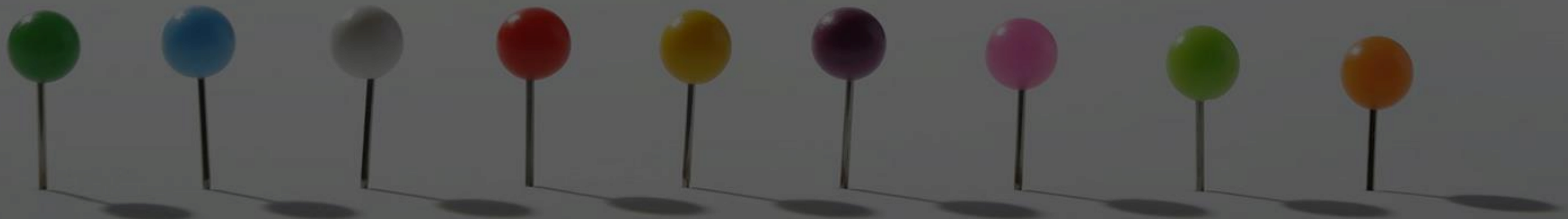
HYPERLEDGER FABRIC: A DISTRIBUTED OPERATING SYSTEM FOR PERMISSIONED BLOCKCHAINS

AUTHORS: ELLI ANDROULAKI, ARTEM BARGER, ET AL.

PRESENTED BY: IMAN CHATTERJEE

ECS 265A



WHAT IS FABRIC

- **Fabric**, one of the **Hyperledger** projects hosted by the Linux Foundation is the first blockchain system supporting **the execution of distributed applications written in standard programming languages**, in a way that allows them to be **executed consistently across many nodes, giving impression of execution on a single globally-distributed blockchain computer**. It is also the **first** distributed operating system for deploying and operating permissioned blockchains.
- First launched in 2015

ACHIEVEMENTS:

- 3500 tps throughput at a latency of a few hundred ms while scaling to more than 100 peers
- Pluggable consensus; Segregates chaincode execution and consensus(unlike EVM of Etherium-piecewise running of smart contract)
- Modularity/pluggability(compatibility with most IT infrastructures)
- No dependence on built in cryptocurrency
- Channels are used for confidentiality and privacy

DRAWBACKS OF EXISTING PERMISSIONED BLOCKCHAINS

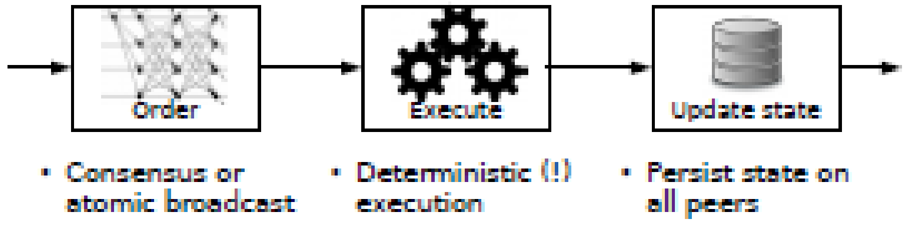
- The trust model of transaction validation is determined by the consensus protocol and is not flexible to adapt to the smart contract needs.
- Smart contracts must be written in a fixed domain-specific language.
- Sequential execution of all transactions by all peers limits performance
- Throughput is inversely proportional to the execution latency

Will this do for all of you?
No - that won't fit ME
1 SIZE FITS ALL
We are ALL unique with individual needs and requirements
© ogilviedesign.co.uk

- Lack in flexibility: Trust assumption(at protocol level) may not match the trust required for smart contract execution. Trust at the application level should not be fixed to trust at the protocol level. The typical trust assumption that upto f Byzantine faults can be tolerated for n>3f peers may not match the trust needed by smart contract.

- Hardcoded consensus: Problem-One size does not fit all. Eg: BFT protocols differ widely in their performance when deployed in different potentially adversarial environments.

ORDER-EXECUTE ARCHITECTURE (FOLLOWED BY ALL PREVIOUS BLOCKCHAINS)


Order
Execute
Update state
Consensus or atomic broadcast
Deterministic (!) execution
Persist state on all peers

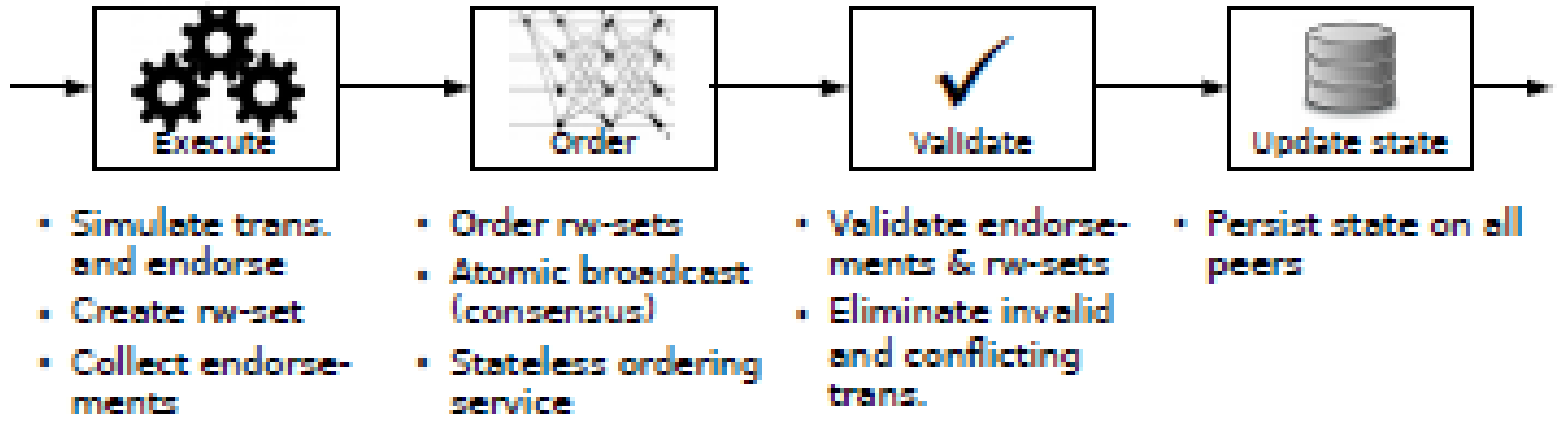
1. A protocol for consensus or atomic broadcast first orders the transactions and propagates them to all peers
2. Each peer executes the transactions sequentially.
3. All peers are required to execute every transaction and all transactions need to be deterministic(same operation must produce the same result in each peer).

LIMITATIONS OF ORDER-EXECUTE

Performance bottleneck due to sequential execution.

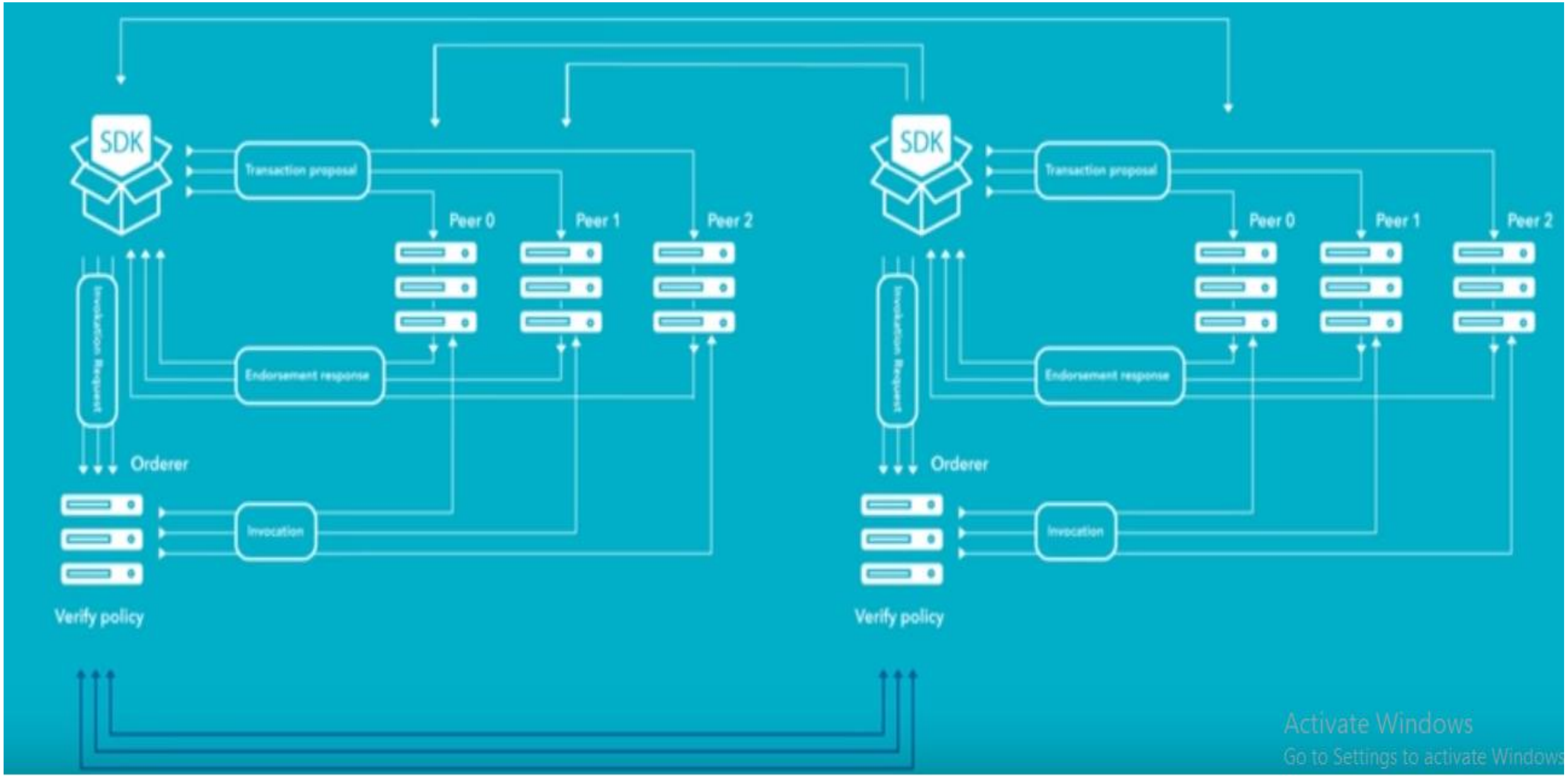
Operations executed after consensus must be deterministic, or the distributed ledger “forks” and violates the basic premise of a blockchain, that all peers hold the same state.

Maintaining confidentiality often becomes a significant challenge because data encryption techniques often contribute a significant overhead that render them unviable practically.


Execute
Order
Validate
Update state
• Simulate trans. and endorse
• Create rw-set
• Collect endorse-ments
• Order rw-sets
• Atomic broadcast (consensus)
• Stateless ordering service
• Validate endorse-ments & rw-sets
• Eliminate invalid and conflicting trans.
• Persist state on all peers

Fabric introduces the execute-order-validate blockchain architecture.

ARCHITECTURE OF FABRIC



BLOCK DIAGRAM OF THE SYSTEM

COMPONENTS OF FABRIC

- **Shared ledger**: world state and blockchain
- **Smart contract**: chaincode, contains the business logic of the system
- **Client node**: a client application
- **Peer nodes**: host ledgers and smart contract, can be endorser
- **Channel**: a logical structure formed by a collection of peers
- **Membership Services Provider (MSP)**: certificate authority that issues node credentials for authorization

EXECUTION PHASE(EXECUTE AND ENDORSE)

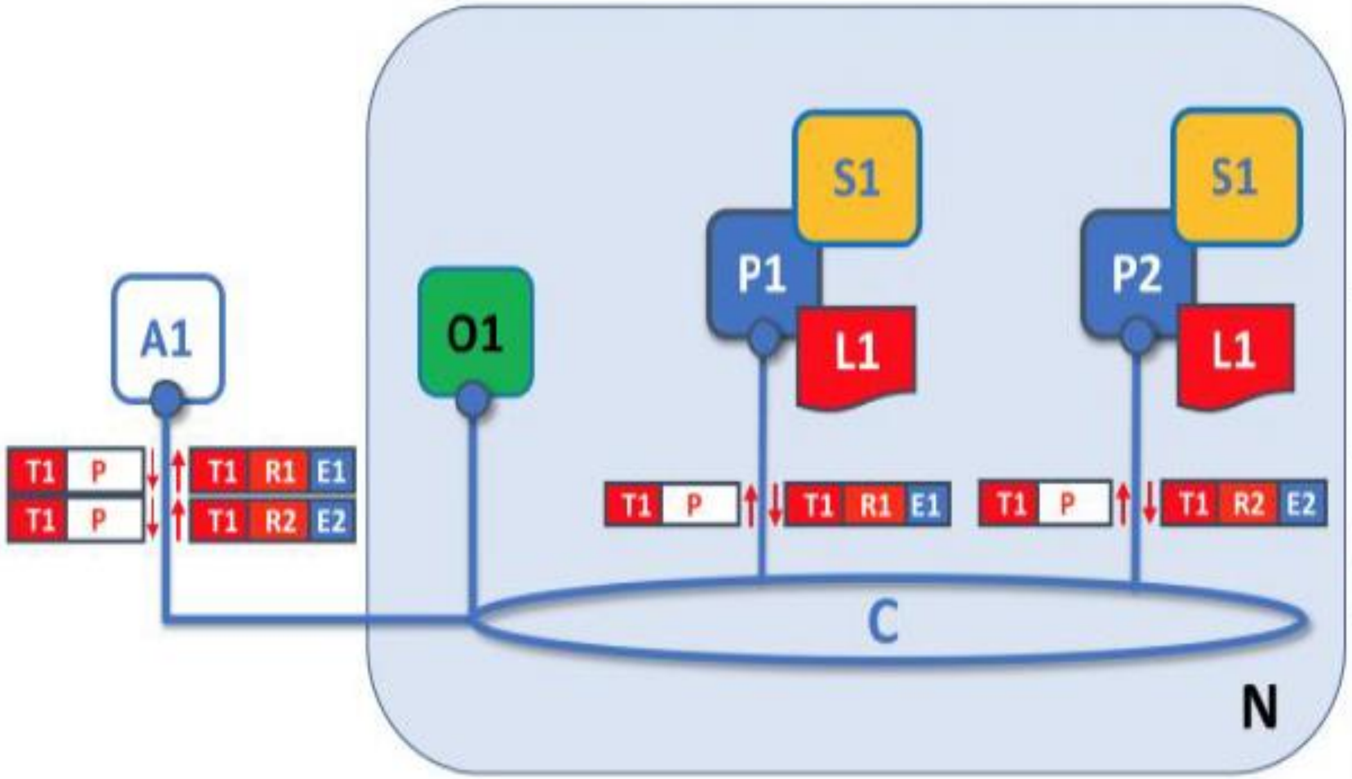
- A **smart contract,** called **chaincode** implements the application logic and runs during the **execution phase**

Endorsement policy lets the chaincode specify the endorsers for a transaction in the form of a set(through parameterization) of peers that are necessary for endorsement;

Client signs and sends transaction proposal to peers specified by the endorsement policy. Each transaction is then INDEPENDENTLY executed by specific peers(by running the chincode operations based on local blockchain state) and its output is recorded; this step is also called **endorsement**.

- Only designated administrators may have a permission to modify endorsement policies through system management functions.
- A proposal contains the identity of the submitting client (according to the MSP), the transaction payload operation, parameters, the identifier for chaincode, and a transaction identifier derived from the client identifier and the chaincode identifier.

Execution Phase



N	Blockchain Network	S	Chaincode
C	Channel	O	Orderer
P	Peer	L	Ledger
T1 P	Transaction T proposal P	T1 R2 E2	Transaction T1, response R2 endorsed with E2
T T C	Ledger transaction T1 flows on channel C	PA C	Principal PA (P1,P2) communicates via channel C.

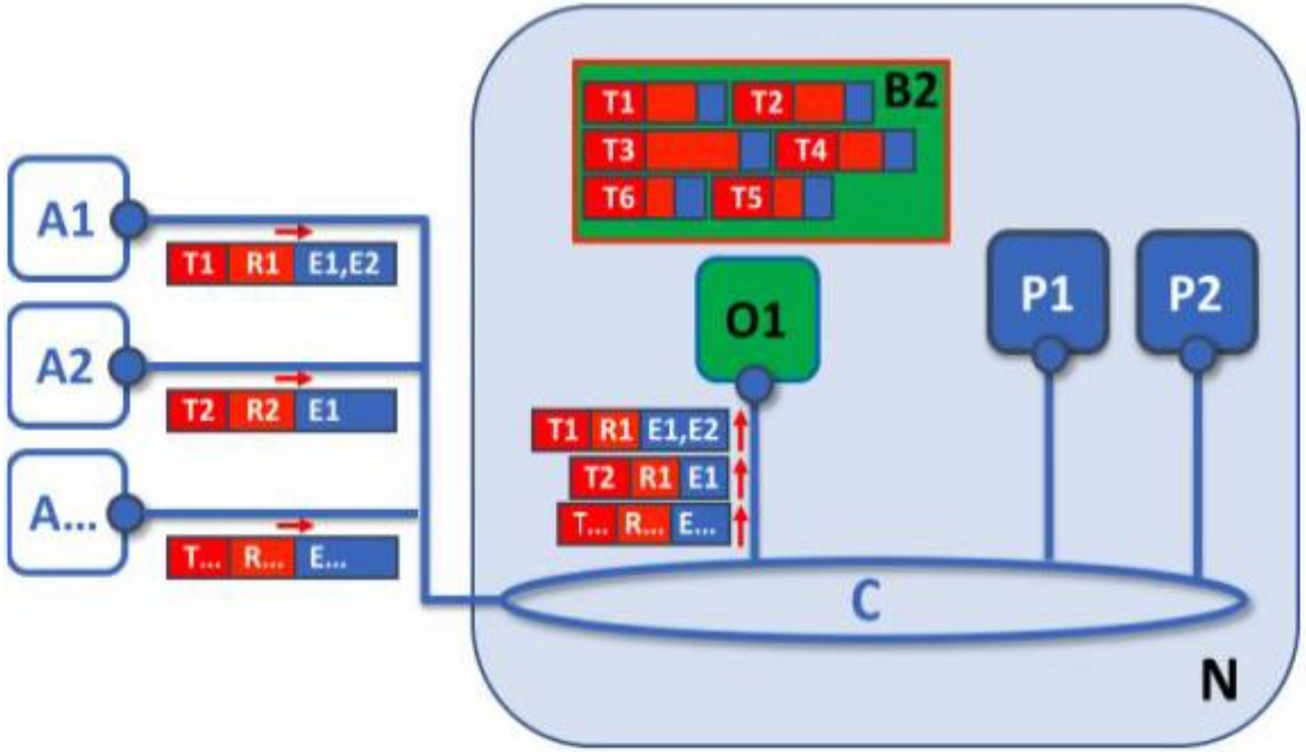
OUTPUT OF EXECUTION PHASE

- **Writeset**, consisting of the state updates produced by simulation(the modified keys along with their new values)
- **Readset**, -(all keys read during simulation +their version numbers).
- Endorser then cryptographically signs a message called endorsement, which contains readset and writeset (together with metadata such as transaction ID, endorser ID, and endorser signature) and sends it back to the client in a **proposal response.**
- If endorsement policy of the chaincode is satisfied(if all endorsers produce the same result(readset and writeset)), endorsement policy is invoked and transaction is created and passed to the **Ordering service**.

ORDERING PHASE

- During this phase, a pluggable consensus protocol produces a totally ordered sequence of the endorsed transactions grouped in blocks. These are broadcast to all peers, with the help of gossip.
- Fabric orders transaction outputs combined with state dependencies, as computed during the execution phase.
- Transaction contains the transaction payload (chaincode operation + parameters), transaction metadata, and a set of endorsements.
- Multiple transactions are batched into blocks and hash chain sequence of block containing transactions is the output.
- The Ordering service atomically broadcasts endorsements and thus achieves consensus on transactions
- Reconfiguration of channel in case of configuration updates also occurs here

Ordering Phase



N	Blockchain Network	P	Peer
B1	Block B1	O	Orderer
T1 R2a E2	Transaction T1, response R2a endorsed with E2	C	Channel
T1 T2 T3 B1	Block B1 contains transactions T1, T2, T3...		
T1 T1 C	Ledger transaction T1 flows on channel C	PA C	Principal PA (P1,P2) communicates via channel C.

- Ordering Service Nodes (OSN) (or, simply, orderers) establishes the total order of all transactions, where each transaction contains state updates and dependencies computed during the execution phase, along with cryptographic signatures of the endorsing peers.
- Orderers are entirely unaware of the application state, and do not participate in the execution nor in the validation of transactions.
- This helps make consensus in Fabric modular and allows easy replacement of consensus protocols in Fabric.

SAFETY PROPERTIES

- Blocks delivered on a particular channel must be totally ordered.
- Agreement: For any two blocks B delivered with sequence number s and B′ delivered with s ′ at correct peers such that s = s ′, it holds that B = B′.
- Hash chain integrity: If some correct peer delivers a block B, with number s and another correct peer delivers block B′ = ([tx1, . . . , txk],h′) with number s+1, then it holds h′ = H(B) where H(·) denotes the cryptographic hash function.
- No skipping: If a correct peer delivers a block with number s > 0 then for each i = 0, . . . , s − 1, peer p has already delivered a block with number i.
- No creation: When a correct peer delivers block B with number s, then for every tx ∈ B some client has already broadcast tx.

CHANNELS

- Fabric allows multiple blockchains to connect to the same ordering service.
- Each such blockchain is called a channel and may have different peers(subsets) as its members.
- Maintaining consensus across channels is difficult
- The total order of transactions in each channel is different from the others.
- Certain deployments with trusted orderers implement by-channel access control for peers.

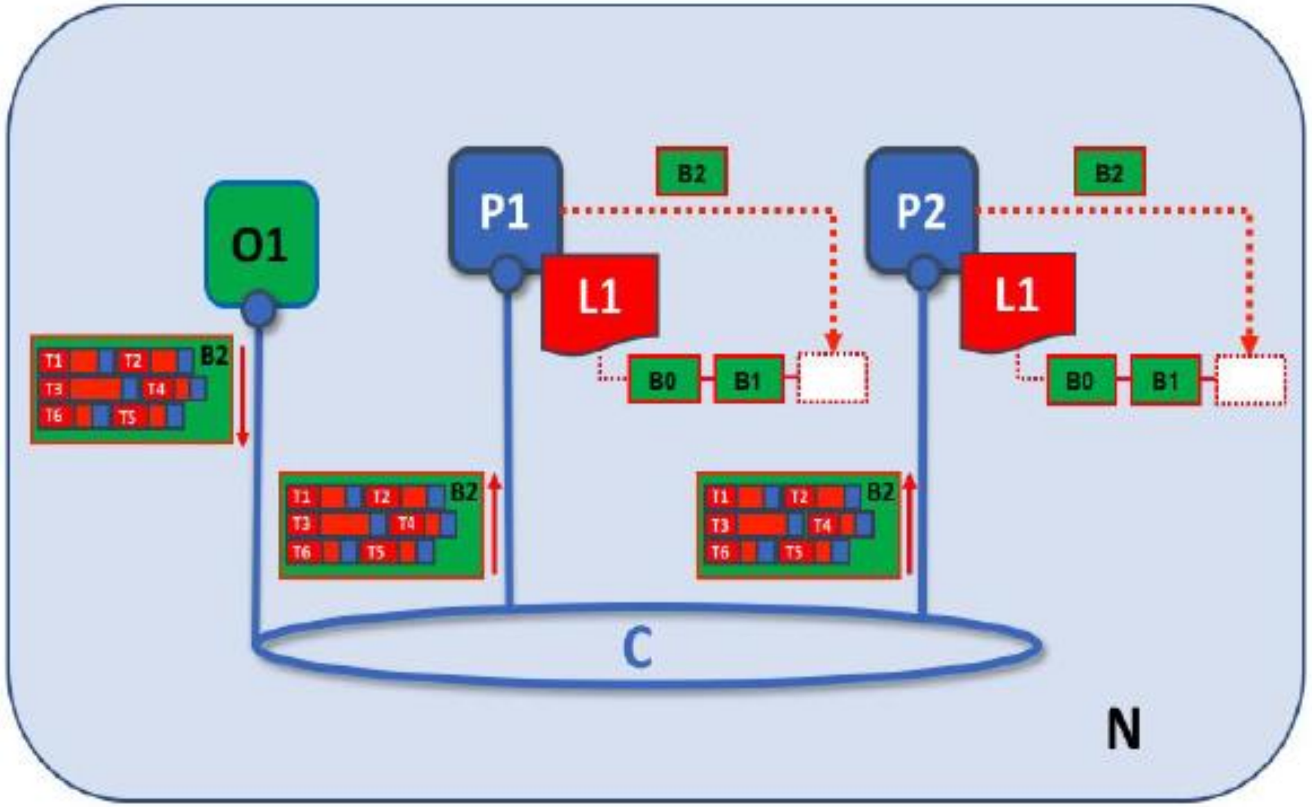
VALIDATION PHASE

- **The role of the validation phase** is to evaluate the endorsement policy(using validation system chaincode). If the endorsement is not satisfied, the transaction is marked invalid and its effects are disregarded.
- A read/write conflict check is performed to compare the key versions in the readset with the current version in the peer(disregard for invalid)
- If first 2 steps pass , a bit mask is added to indicate validity of transactions in the block.

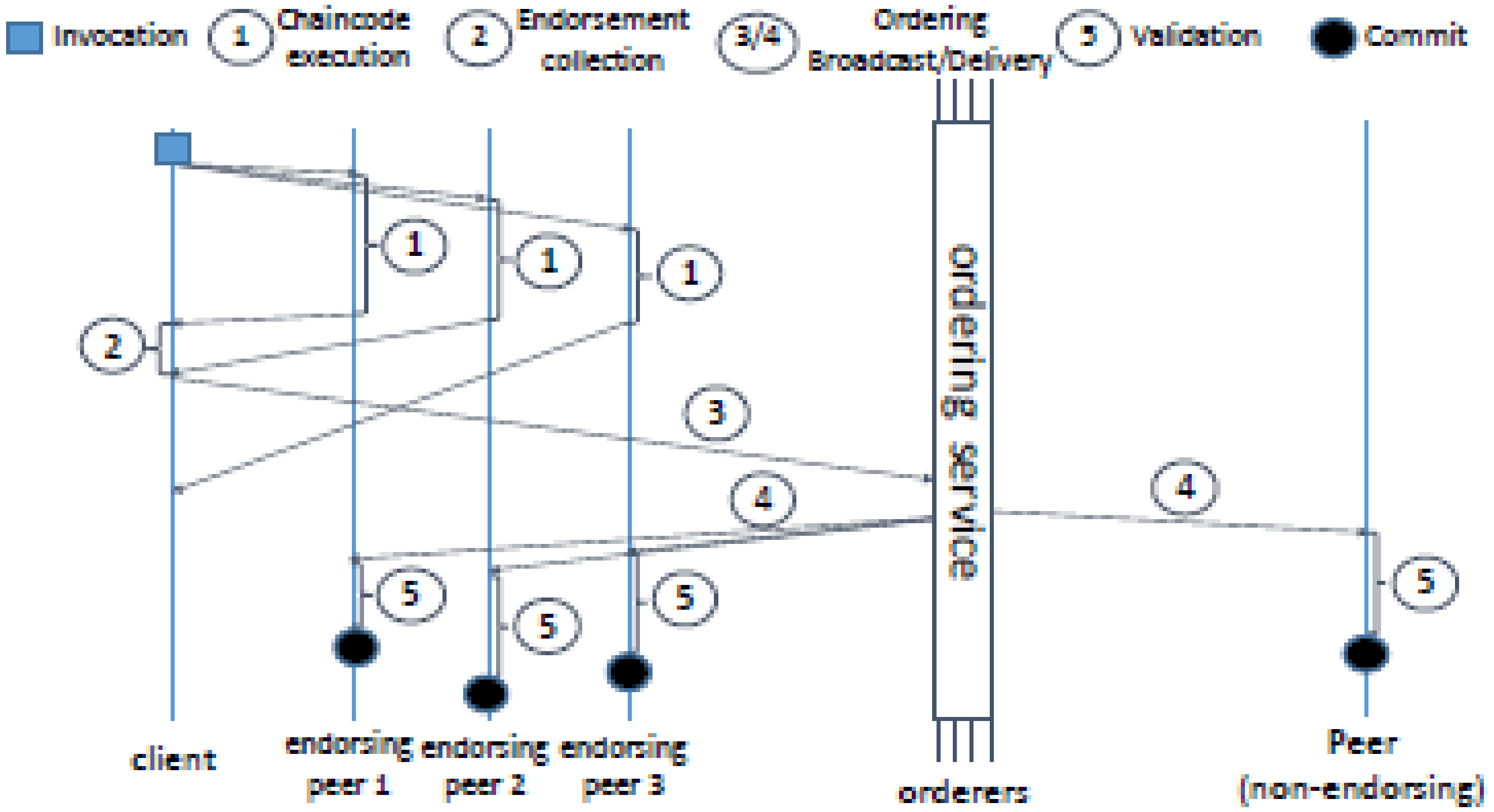
During this phase, each peer validates the state changes from the endorsed transaction with respect to endorsement policy, appends the block to the locally stored ledger and updates the state.

- Validation is deterministic since all the peers validate transactions in the same order.

Validation Phase



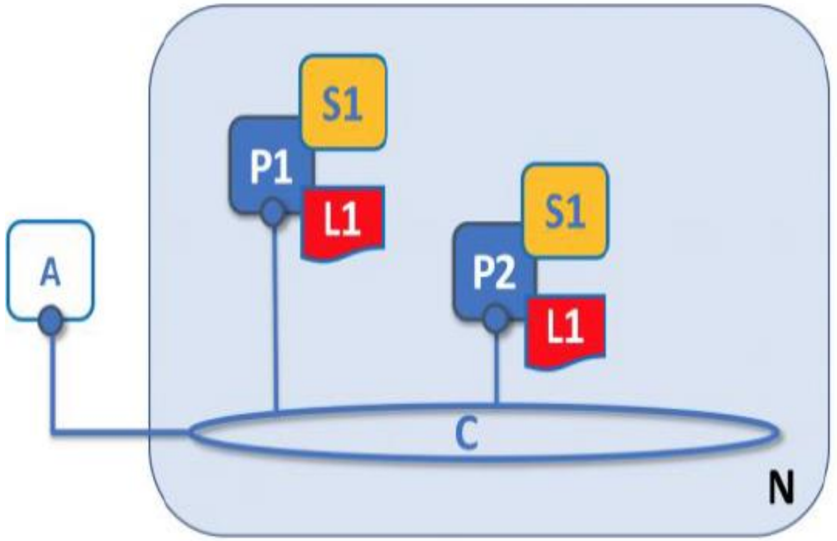
N	Blockchain Network	P	Peer
C	Channel	O	Orderer
L	Ledger	B	Block B
L1 B0 B1	Ledger L1 has blockchain with blocks B0, B1	T1 T2 T3 B1	Block B1 contains transactions T1, T2, T3...
B1 C	Block B1 flows on channel C	PA C	Principal PA (P1, P2) communicates via channel C.

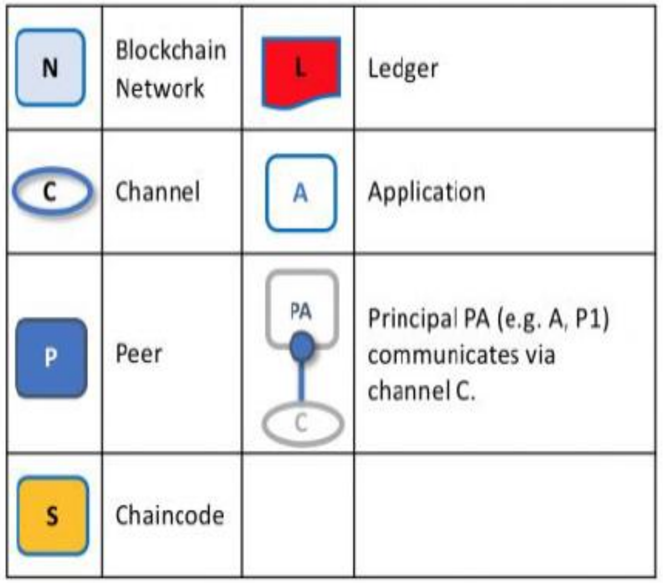

Invocation
1 Chaincode execution
2 Endorsement collection
3/4 Ordering Broadcast/Delivery
5 Validation
Commit
ordering service
1
1
1
2
3
4
4
5
5
5
5
client
endorsing peer 1
endorsing peer 2
endorsing peer 3
orderers
Peer (non-endorsing)

IDENTITY OF NODES

As Fabric is permissioned, all nodes in the network have an identity, as provided by a modular **membership service provider**

- **Clients** submit transaction proposals for execution, help orchestrate the execution phase, and, finally, broadcast transactions
- **Peers** execute transaction proposals and validate transactions. Peers maintain a copy of the ledger recording transactions as a hash chain and the current ledger state. ONLY a subset of peers execute transaction proposals.
- OSN





N	Blockchain Network	L	Ledger
C	Channel	A	Application
P	Peer	PA C	Principal PA (e.g. A, P1) communicates via channel C.
S	Chaincode		

CHAINCODE EXECUTION

- Chaincode is executed within a secure Docker container environment loosely coupled to the rest of the peer, which supports plugins for adding new chaincode programming languages. Eg: Go, Java, and Node

CONCLUSION

- Fabric introduces a novel architecture that separates transaction execution from consensus and enables policy-based endorsement
- Future versions of Fabric could benefit from pipelining the sequential validation stages.

THANK YOU