本篇论文提出了一种扩展性强且模块化的区块链系统Fabric。其能将共识协议模块化，以满足不同应用场景。并且能用通用编程语言（例如Java和Go）编写smart contracts（在Fabric中叫做chaincode）。同时不像Ethereum那样依赖加密货币。其采用的架构是execute-order-validate，与大多数区块链所采用的order-execute架构不同。

# Background

目前order-excute架构具有以下缺点：

1. Sequential execution：smart contracts在所有节点上线性执行，效率低下，且无法判断smart contracts是否是死循环。
2. Non-deterministic code：导致节点状态不一致，使区块链分叉。
3. Confidentiality of execution：因为所有transaction将会在所有节点上运行，缺乏隐私性。
4. Fixed trust model：应用层面上的信任模型和共识协议上的信任模型高度耦合。
5. Hard-coded consensus：单一的共识协议无法适用于所有的应用场景。

# Architecture

Fabric系统中存在三种角色：

1. Clients负责提交transaction给Peers（也称为endorsers，由endorsement policy决定）运行，并把收集到的endorsements打包上传给OSN进行排序
2. Peers负责运行并验证transaction，存储区块链
3. Ordering Service Nodes (OSN)负责对transaction进行排序，通过原子广播在彼此之间达到共识，并将排序好的transaction打包进区块通过gossip协议发送给所有Peers。

Fabric系统运行由三个阶段组成：

1. 执行阶段：clients向其所有endorsers发送transaction proposal，其中包括Clients的用户信息、transaction具体操作、参数、chaincode标识符、累加数和本次transaction的标识符。然后endorser在预先安装好chaincode的docker容器中模拟运行transaction。chaincode只能通过GetState、PutState和DelState三种操作访问区块链，不保存任何局部状态，且在执行过程中不会改变区块链的状态。除此之外，chaincode执行完transaction之后将保存该transaction产生的结果，称作rw-sets。之后endorsers将把rw-sets和一些元数据进行签名作为proposal response（endorsement）发送回给对应clients。clients检查收到的endorsements是否都是相同的rw-sets，最后判断是否满足endorsement policy，若满足则将整个transaction和endorsements进行打包发送给OSN进行排序。  
   该阶段能有效解决non-deterministic code所带来的危害，具体来说就是若各个endorser执行transaction所产生的rw-sets不一致的话，clients将收集不到足够多的endorsements来满足endorsement policy，进而导致无法继续后续操作。DoS攻击给endorser带来的危害也极其有限，其能根据本机的规定检测并抵御DoS攻击。
2. 排序阶段：clients将transaction和与其对应的endorsements打包发送给OSNs。OSNs在彼此之间通过原子广播对transactions顺序达成共识。之后将transactions打包进多个区块中。peer通过调用deliver函数或者gossip协议获取区块。
3. 验证阶段：peers通过validation system chaincode (VSCC)根据endorsement policy验证transaction的endorsement的合法性，若不合法则将其标记为”invalid”并注销掉它的作用。其次，peers通过验证每个transaction的read sets中所读取的区块链状态是否与当前区块链的状态相同来判断transaction的合法性。最后，peers根据合法transaction的write-set更新区块链状态，并在区块中用一比特表示该transaction的合法性。值得注意的是由于验证是在共识之后发生的，区块中会保存非法transaction，只是该transaction并没用作用于区块链。