HYPERLEDGER

BLOCKCHAIN TECHNOLOGIES FOR BUSINESS

Sawtooth v1.0

Zac Delventhal*
Hyperledger Sawtooth 维护者

2018年3月

*Slides by Dan Middleton

Agenda | 议程

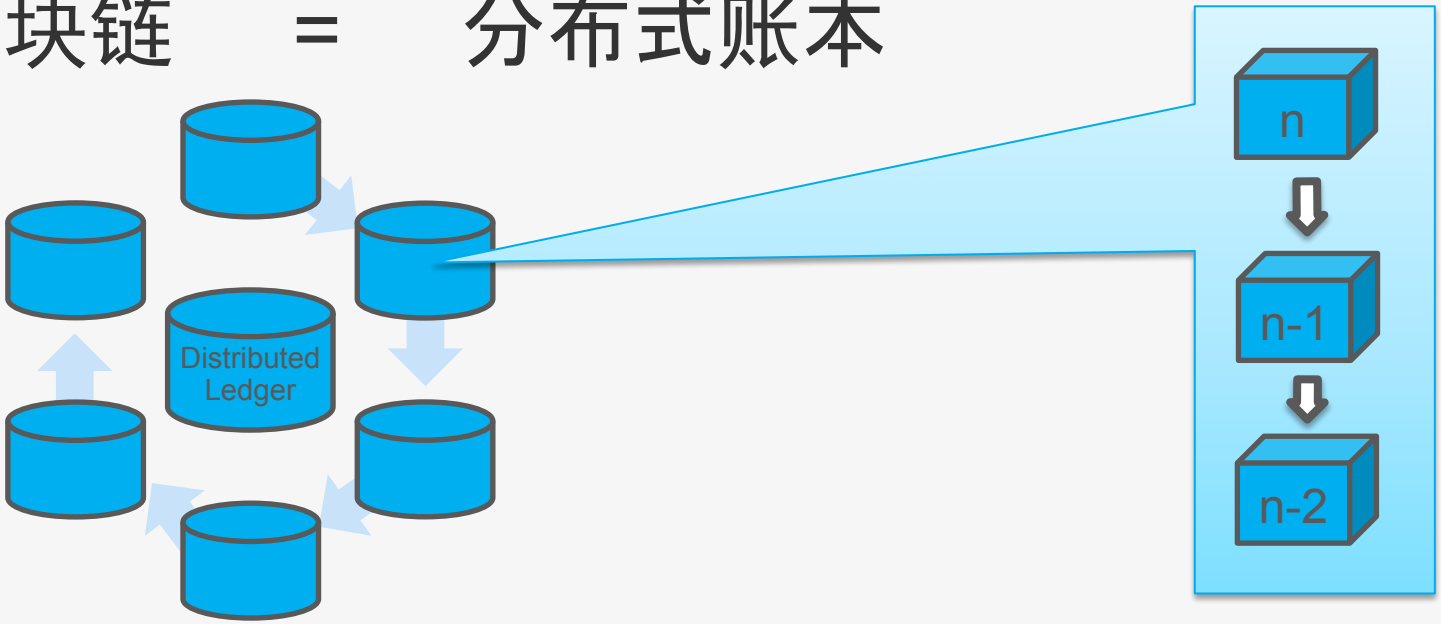
Sawtooth Design Motivations | 设计愿景

Sawtooth 1.0 Features | 特征

Distributed App Development | 分布式App开发

- 代码下载Code: https://github.com/hyperledger/sawtooth-core
- 网上演示Demos: https://sawtooth.hyperledger.org/examples/
- 文档查看Docs: https://sawtooth.hyperledger.org/docs/

Blockchain = Distributed Ledger
区块链 = 分布式账本



Each node is an instance of a database (ledger) managed by all participants.
| 每个参与者节点管理一个数据库实例（账本）

Within each database, blocks of transactions are cryptographically chained in order.
| 每个数据库中，交易区块加密并按顺序链接

Why Blockchain | 为什么用区块链

Mutually distrusting organizations that update the same database | 不互信的组织更新相同数据库。

Immutable transaction history | 不可篡改的历史数据

High availability | 高可用性

- Crash fault tolerant | 宕机容错
- Byzantine fault tolerant | 拜占庭容错
- Liveness |永远活跃

Why *Not* Blockchain | 为什么*不用* 区块链

Wrong Usage Model | 错误的使用模式：

- Internal-only Business Process | 内部业务流程

A centralized architecture destroys the main utility of a distributed ledger. | 中心化的架构摧毁了分布式账本的主要用途。

Active Research Areas | 活跃研究领域:

- Throughput | 高吞吐
- “Private” Transactions | “私人” 交易

When people talk about blockchain security, they mostly mean availability and integrity guarantees. Confidentiality is open research. | 人们谈论区块链安全时，主要指可用性和完整性保 证。保密性是开放的研究。

HYPERLEDGER
BLOCKCHAIN TECHNOLOGIES FOR BUSINESS

Hyperledger Sawtooth 1.0

Architecture & Features

架构和功能

1.0 Released January 2018

Sawtooth Design Philosophy | 设计理念

Design for *Scale*
扩展性 设计

Keep Distributed Ledgers *Distributed*
保持分布式账本*的分布式*

Ease-of-use for
Development & Deployment
易用性，*开发和部署*

Make Smart Contracts *Safe*
让智能合约 *安全*

v1.0 Highlighted New Features亮点新功能

Advanced Transaction Execution | 高级交易执行

- Parallel Execution |并行执行
- Multi-Language Support |多语言支持
  - Build apps in your language of choice |用你的语言创建APP

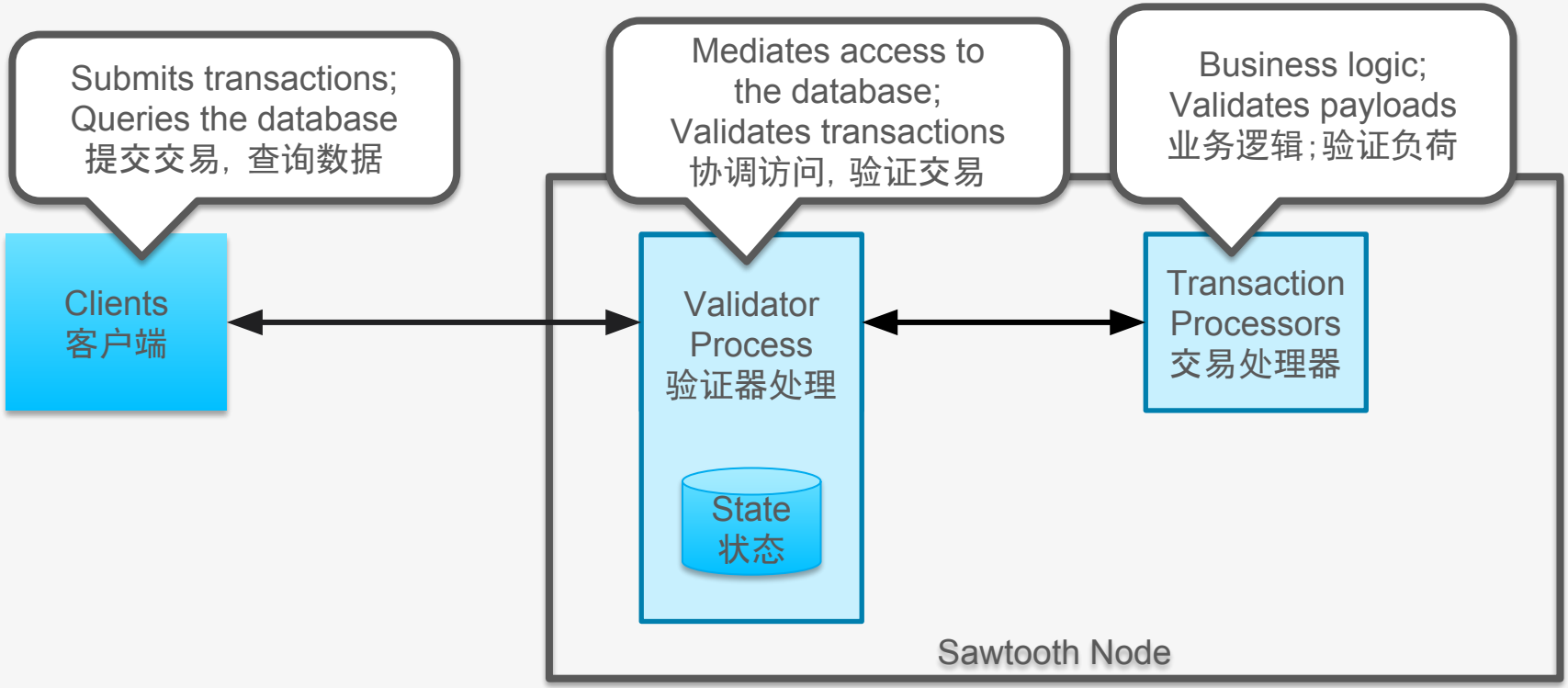
On-chain Governance | 链上治理

- Dynamic Consensus | 动态共识
  - Proof of Elapsed Time (PoET) | 经历时间证明
- New Permissioning Features | 新的许可功能

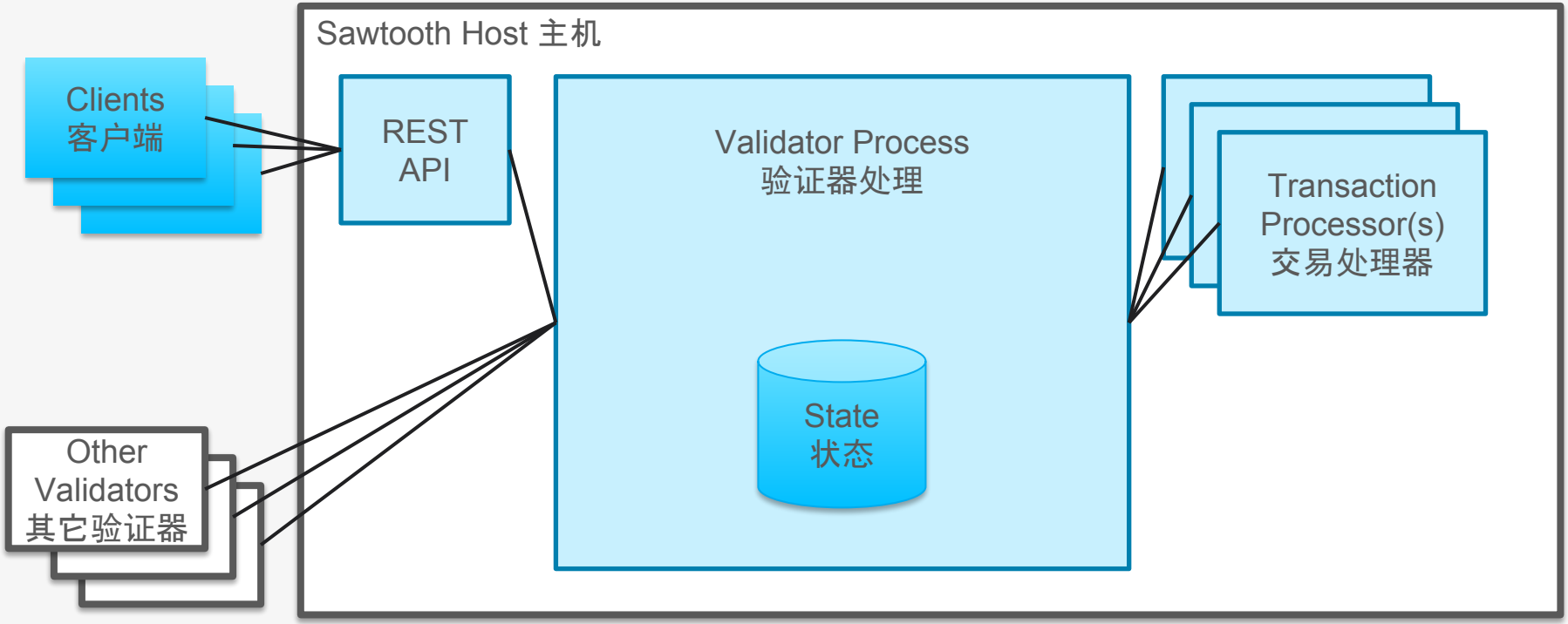
Distributed Applications | 分布式应用

- Seth
  - Sawtooth + Ethereum
  - Run solidity on Sawtooth | 运行健壮
- Supply Chain | 供应链
  - Provenance of goods | 商品出处
  - Telemetry / tracking | 计量/追踪

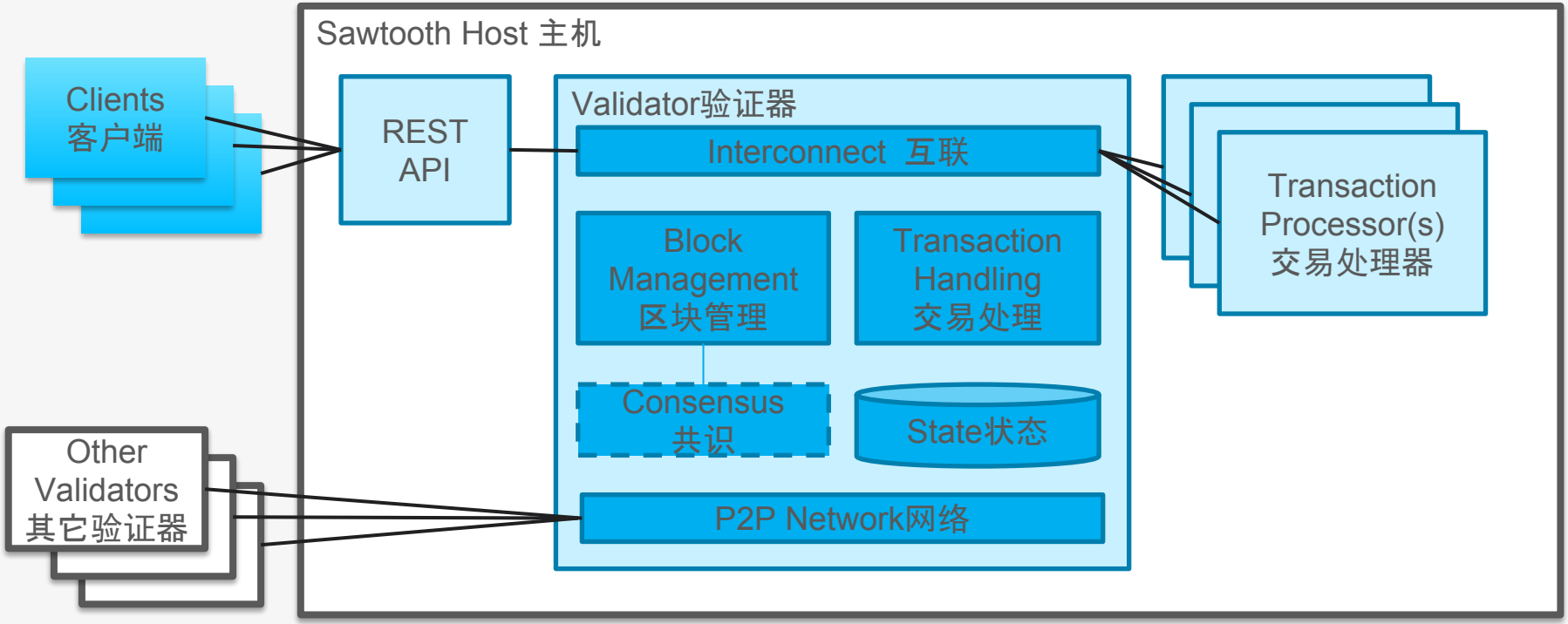
Sawtooth Architecture 架构



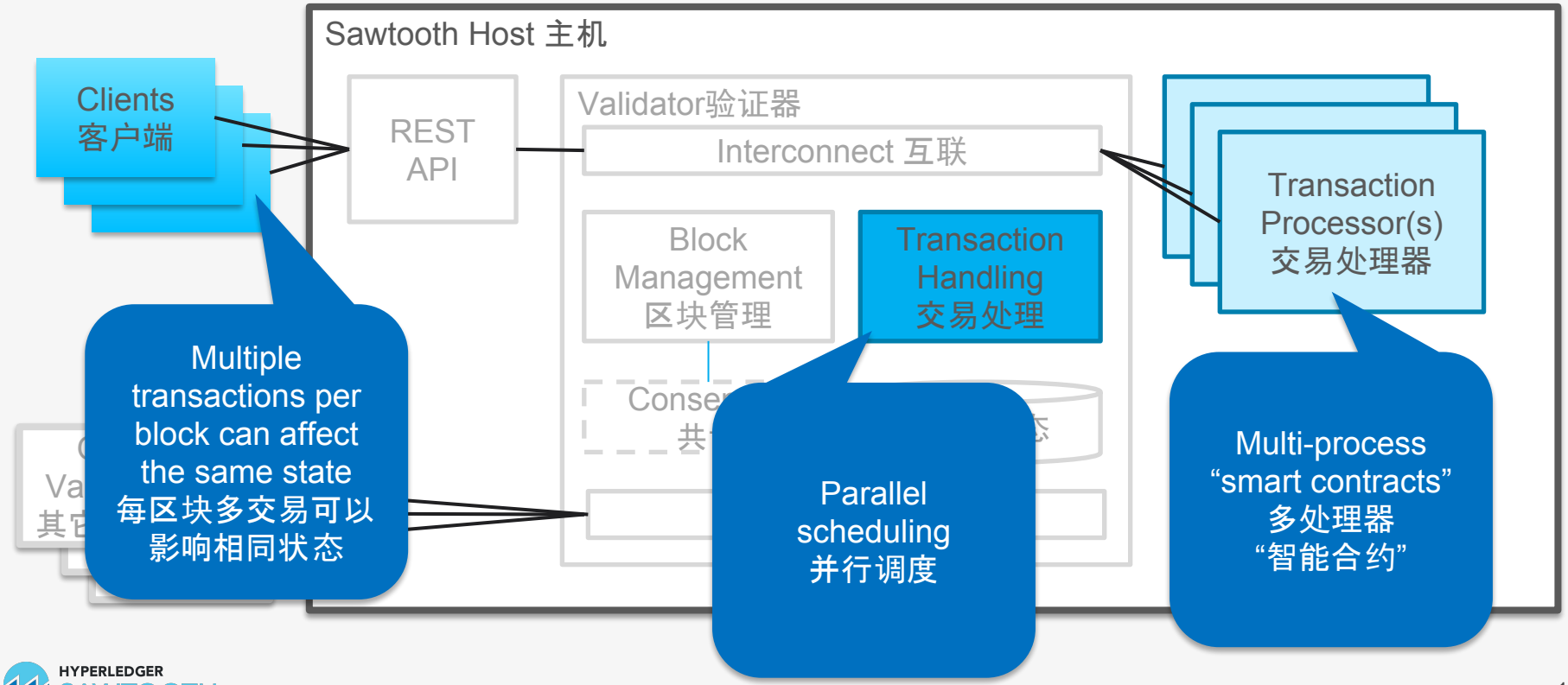
Sawtooth Architecture架构



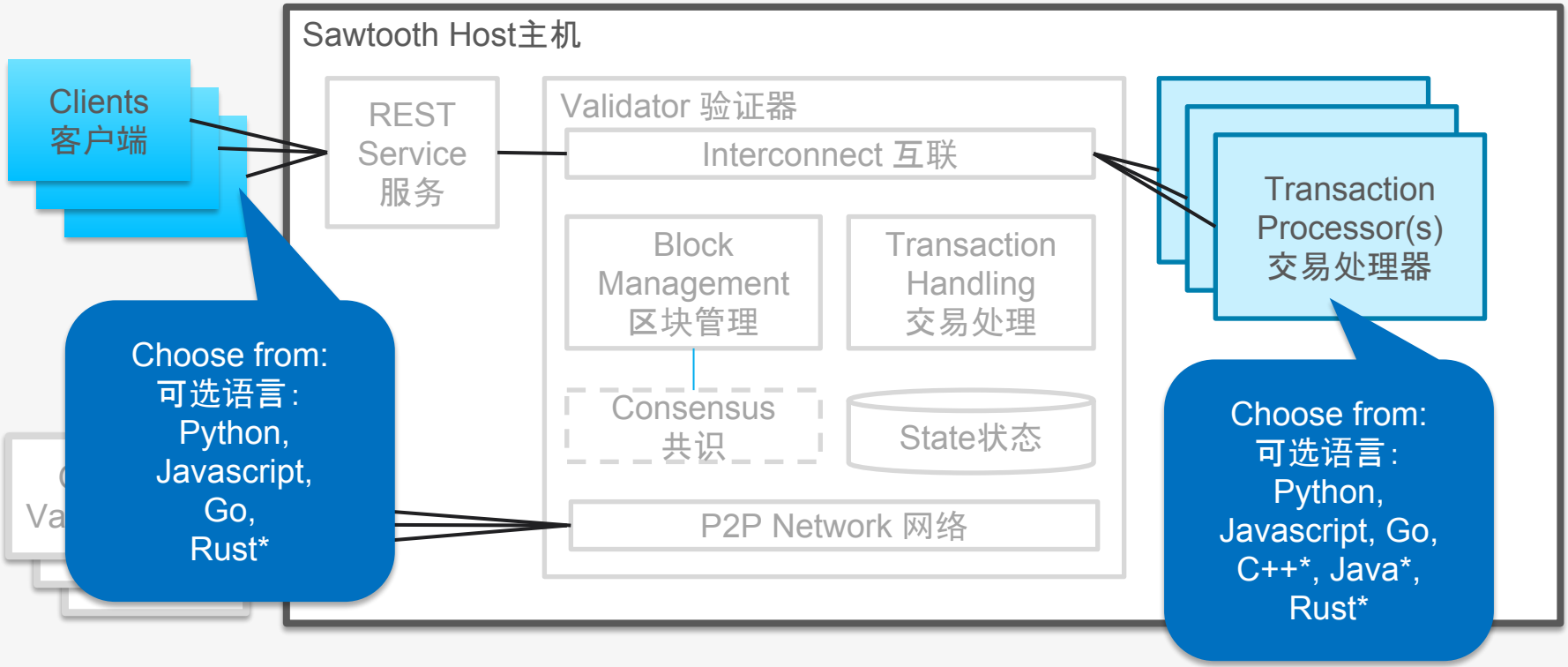
Sawtooth Architecture 架构



1.0 Features: Parallel Execution 并行执行



1.0 Features: Multi-Language Support多语言支持



*No all features are available in these languages in v1.0.1

v1.0 Features: On-chain Governance | 链上治理

Control the blockchain on the blockchain在链上控制区块链

Settings Transaction Family enables participants to agree on network policies | 设置交易家族允许参与者同意网络策略

For example, vote on changing consensus parameters using registered public keys of consortia members. | 例如，联盟成员通过使用登记的公钥投票选举修改共识参数

Settings are extensible – they can be added after genesis. | 设置可扩展 – 在创世以后可以增加设置

Setting (Examples) 设置（示例）	Value 值
sawtooth.poet.target_wait_time	5
sawtooth.validator.max_transactions_per_block	100000

sawtooth.validator.transaction_families	[{
	"family": "intkey",
	"version": "1.0"
	},
	{
	"family": "xo",
	"version": "1.0"
	}]

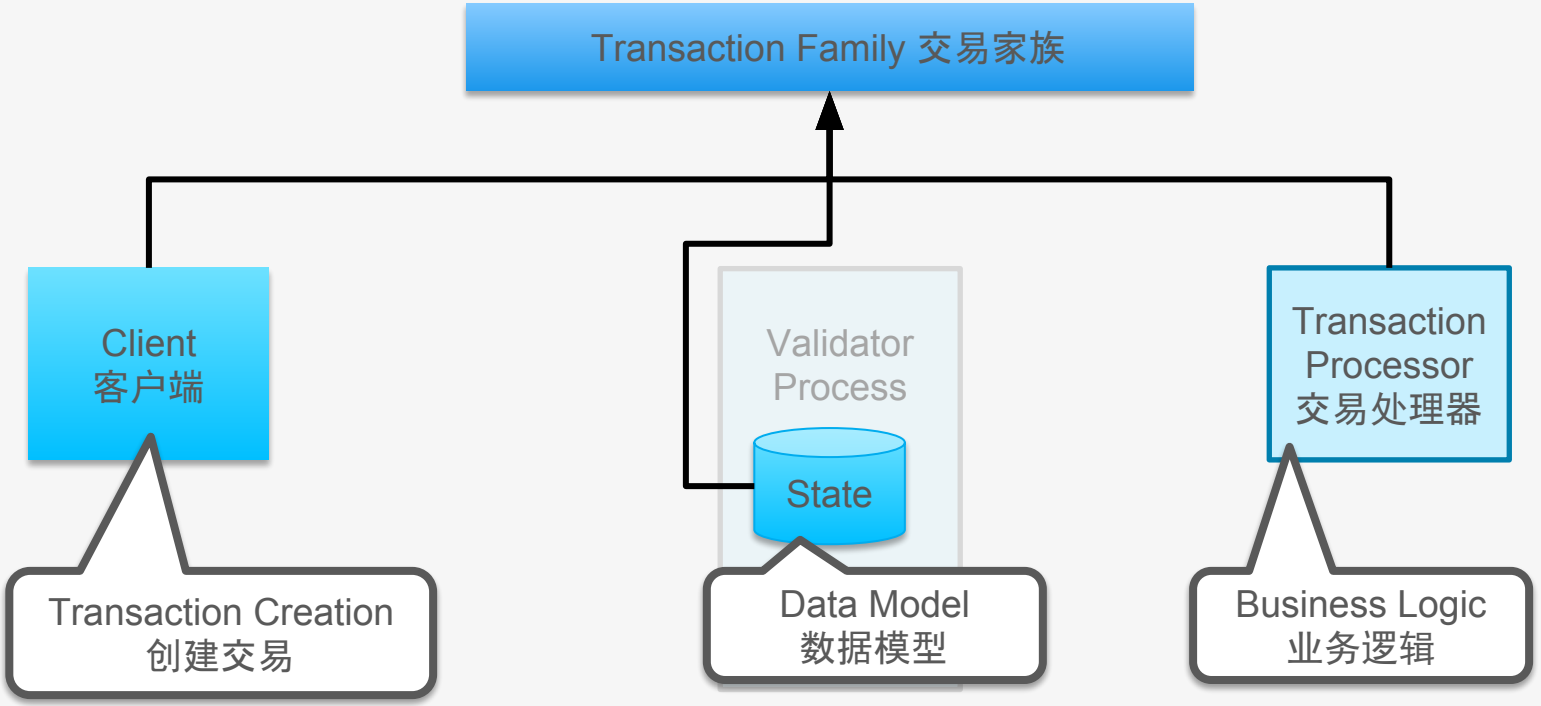
HYPERLEDGER
BLOCKCHAIN TECHNOLOGIES FOR BUSINESS

Hyperledger Sawtooth 1.0

App Development App开发

1.0 Released January 2018

Application Development应用开发

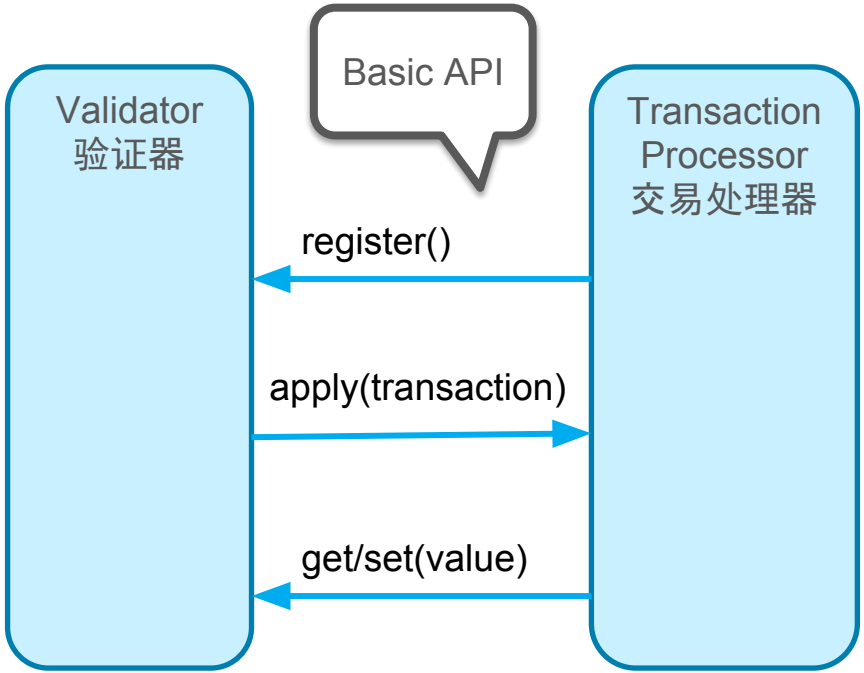


Transaction Families: The Transaction Processor
交易家族:交易处理器

All validators in the network run every authorized transaction processor. | 网络上所有验证器运行每一个授权的交易处理器

On receipt of a transaction the validator will call the TP's *apply()* method. | 一旦接收到交易, 验证器将调用*apply()* 方法

Business logic simply goes in *apply()* and gets and sets state as needed. | 业务逻辑简单地参与*apply()* 并根据需要获得或设置状态。



Transaction Families: The Client | 交易家族:客户端

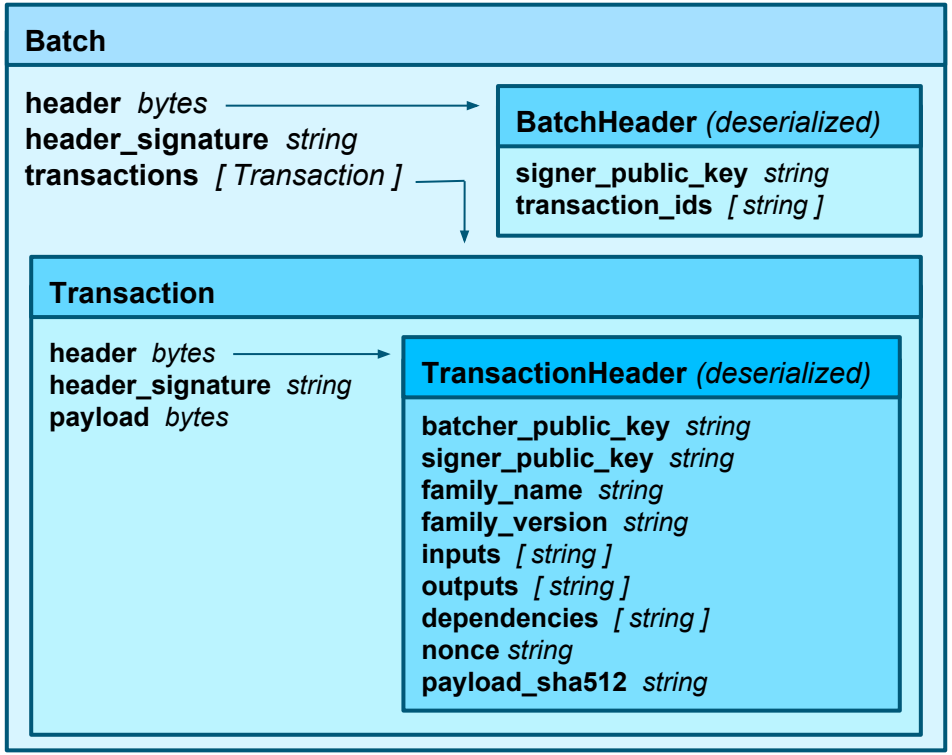
Clients can be browser apps, CLIs, etc. | 客户端可以浏览app，CLI等

Main job is to package and sign transactions & batches | 主要作业

Needs to know how to *encode* transaction payloads and *decode* data from state | 需要知道如何编码交易数据和从state中解码数据

Clients can post batches through the REST API or connect to the validator directly | 客户端可以通过REST API或者直接连接到验证器发布批作业

Transactions, Batches, and Blocks 交易、批处理和区块



Transactions are wrapped in batches which provide an atomic unit of commit for multiple transactions (which can span transaction families). | 交易被包裹到批作业，它为多个交易（可以跨交易家族）提供了一个原子性单位。

Transactions declare input and output addresses (including wildcards) to allow for state access isolation calculations (topological sort on DAG) in the scheduler. | 交易声明输入和输出地址（包含通配符），允许在调度器中进行状态访问隔离计算（按DAG拓扑排序）。

These inputs and outputs are enforced by the Context Manager on the context established for the transaction. | 这些输入和输出由上下文管理器强制，在交易建立的上下文中执行。

This allows parallel validation and state delta aggregation across a potentially large number of transactions (and across blocks). | 这允许对可能的大量交易（并且跨区块）进行并行验证和状态差异聚合。

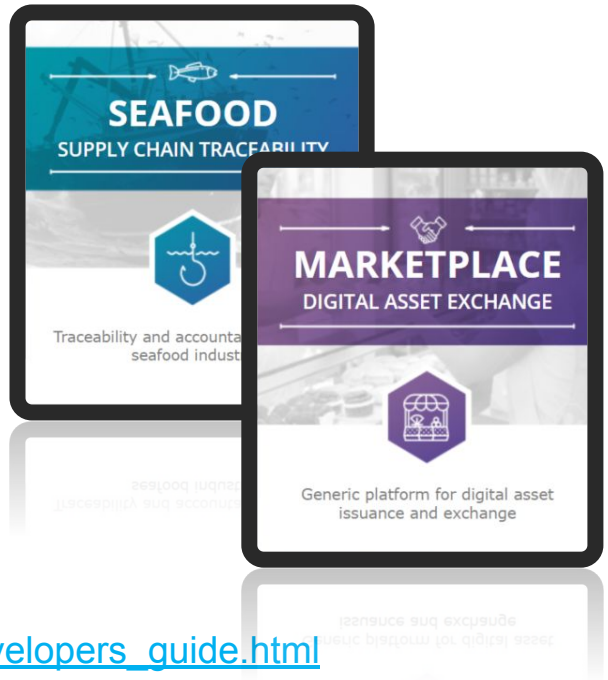
Example Applications & Code 应用和代码示例

Supply Chain:
https://github.com/hyperledger/sawtooth-supply-chain

Marketplace:
https://github.com/hyperledger/sawtooth-marketplace

Hyperdirectory:
https://github.com/hyperledger/sawtooth-hyper-directory

Dev Guide:
https://sawtooth.hyperledger.org/docs/core/releases/1.0.1/app_developers_guide.html



Check It Out | 看一看

Give Sawtooth a try | 试试Sawtooth

- Work through the tutorials | 跑一遍教程
- Run some example transaction families | 运行一些交易家族的例程
- Build your own transaction family to explore use cases | 创见自己的交易家族

Become a contributor | 成为贡献者

- Join the community | 加入社区
- Help with docs, code, examples | 帮助书写文档、代码、例程
- Become an expert and help others on chat | 成为专家，帮助他人

Links | 链接

- Code代码: https://github.com/hyperledger/sawtooth-core
- Docs文档: https://sawtooth.hyperledger.org/docs/
- Chat对话: https://chat.hyperledger.org/channel/sawtooth


HYPERLEDGER
BLOCKCHAIN TECHNOLOGIES FOR BUSINESS
Thanks!