Blockchain as a Business Platform

Dr. Sachiko Yoshihama,
Senior Technical Staff Member,
Senior Manager, FSS & Blockchain, IBM Research - Tokyo
Self Introduction

Sachiko Yoshihama, Ph.D.
STSM, Senior Manager, FSS & Blockchain Solutions, IBM Research - Tokyo

3,000 researchers WW
All 13 labs are involved in Blockchain
• Development of Hyperledger Fabric
• Proof-of-concept, pilot,
• Solution development
Outline

Blockchain for Businesses

Technical Challenges

Smart Contract Generation
<table>
<thead>
<tr>
<th>Outline</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blockchain for Businesses</td>
</tr>
<tr>
<td>Technical Challenges</td>
</tr>
<tr>
<td>Smart Contract Generation</td>
</tr>
</tbody>
</table>
Blockchain for Businesses

Public Network

• Anybody can join
• Virtual currencies without needing a central bank or government

Permissioned Network

• Only permissioned parties can join
• Transforming businesses across organizations
• High security and reliability requirements
• Based more on traditional technologies such as distributed DB and transaction systems
Hyperledger®, a Linux Foundation project

– Announced by The Linux Foundation on December 17, 2015 with 17 founders, now over 260 members

– Hyperledger® is an open source and openly governed collaborative effort to advance cross-industry blockchain technologies for business, hosted by The Linux Foundation.

– Hyperledger® Fabric is a blockchain framework implementation and one of the Hyperledger projects, intended as a foundation for developing applications/solutions with a modular architecture

Enable adoption of shared ledger technology at a pace and depth not achievable by any one company or industry
Hyperledger Community

Frameworks

- **Hyperledger Burrow**: Permissionable smart contract machine (EVM)
- **Hyperledger Fabric**: Permissioned with channel support
- **Hyperledger Indy**: Decentralized identity
- **Hyperledger Iroha**: Mobile application focus
- **Hyperledger Sawtooth**: Permissioned & permissionless support; EVM transaction family

Tools

- **Hyperledger Caliper**: Blockchain framework benchmark platform
- **Hyperledger Cello**: As-a-service deployment
- **Hyperledger Composer**: Model and build blockchain networks
- **Hyperledger Explorer**: View and explore data on the blockchain
- **Hyperledger Quilt**: Ledger interoperability
- **Hyperledger Ursa**: Shared Cryptographic Library

Hyperledger Activities

- 3 Years since launch
- 65K+ Commits
- 44 12! Projects
- 789 Contributors
- 11.3M Lines of code across Hyperledger projects
- 260+ Members (50+ in China)
- 9 Active Community Working & Special Interest Groups
- 150+ Meetups Worldwide
- 50K Meetup Participants
- 140K+ EdX Students

Hyperledger Fabric Basic Functionalities

Immutable record / data shared across participants in a business network.

Blocks and State DB

Authentication, authorization, access control, encryption...

PKI-based membership management

Ledger

Smart Contract

Shared business rules and processes

Chaincode

Consensus over the approval of transactions and their orders

Distributed multi-party consensus algorithm

Security & Privacy
Roadmap

- Channels
- Selective endorsement
- SOLO/Kafka orderers
- LevelDB or CouchDB

07/17

v1

- JavaScript chaincode
- Connection profile
- Encryption library
- Attribute access control
- CouchDB indexes
- Channel based events

03/18

v1.1

- ACLs
- Service discovery
- Pluggable endorsement and validation
- Private Data Collections

06/18

v1.2

- State based endorsement
- Java chaincode
- CouchDB pagination
- Identity Mixer

10/18

v1.3

- Operational metrics and logging
- SDK and shim improvements
- Long Term Service (LTS) support

1Q/19

v1.4

- Local collections
- SDK improvements
- Lifecycle changes
- Revocation for Idemixer
- Token Support
- RAFT Consensus
- EVM support

2Q/19

v2.0

Based on https://wiki.hyperledger.org/projects/fabric/roadmap - Dates determined by the Hyperledger community - (*) Subject to change
Blockchain Use Cases are Expanding

Financial
- Financial Markets
- Derivatives
- KYC/AML
- Contract Managements
- Reinsurance

Non-Financial
- Global Trade
- Supply Chain Management
- Food Traceability
- Provenance of diamonds

Automation of Contracts and Processes

Data Sharing

Trading coins / tokens / assets

Efficient
Eliminate redundancy
Standardized

Asset management

Cross-Border Payments

Digitization of Contracts and Processes

Trading coins / tokens / assets

Provenance of diamonds

Asset management

© IBM Corporation, 2019
Contract Execution

**Processes**
- Issues report when product is shipped
- Issues invoice when report is approved
- Makes payment when invoice is approved

**Consistency Checks**
- Is PO consistent with MSA?
- Is received product the one in PO?
- Is amount of invoice correct?

**Buyer**
- agree
- issue
- receives products
- approve
- approve
- make payment

**Blockchain**
- Master Agreement
- Purchase Order
- Delivery Report
- Invoice

**Seller**
- agree
- accept
- ships product
- issues
- issues

$1000

© IBM Corporation, 2019
Complicated netting and DVP (Delivery-versus-Payment) settlement can be automated by Turing-complete smart contract.

Today’s Financial Market Infrastructure

- Master Data Mgmt
- Settlement of Securities
- Central Securities Depository, CSD
- Interbank Payment Network
- Settlement of Money
- Exchange
- Clearing
- Central Counter Party (CCP)
- Trade
- Reconciliation
- Audit

Financial Market Infrastructure with Blockchain

- Administration
- Audit
- Admin
- Brokers
- Depository

- Bank A
- Investors
- Bank B
- Investors
- Bank C
- Investors

- Brokers
- Depository
- Investors
- Bank A
- Bank B
- Bank C

- Securities Depository as Distributed Ledger
- Reconciliation, clearing and settlement as a distributed shared processes, enabled by smart contract
- Corporate Actions (dividend, stock split) are automated
Not only proof-of-concepts

Emerging Blockchain Production Systems

- Digital Identity
- Trade Finance
- Global Trade
- Food Traceability
- Remittance

IBM Blockchain World Wire
Food Trust
Only 1 in 4 Consumers Trust the Food Ecosystem

- **Food Safety**: 1 out of 10 people get sick each year, and 420,000 die from foodborne illness.
- **Supply Chain Inefficiency**: 80% of CPGs business are partially or entirely paper-based.
- **Food Waste**: 1/3 of fresh food is thrown out because it is considered unacceptable.
- **Food Fraud**: 1 in 5 seafood samples mislabeled worldwide (43% mislabeled in NYC).

The root of these issues, and many others, are the lack of trust and transparency.
The Problem:

- **Data is siloed** within each company and accessing it requires a request and time.

- Exchange of information takes place between a pair of partners; to get information from a distant partner may require **intermediaries**, time, resources.

- Most transactions are still **paper-based**, creating inefficiencies and opportunities for fraud.

- Because everyone maintains their own record of transactions, **differences** take time and resources to reconcile.

---

IBM Blockchain
IBM Food Trust offers industry-specific functionality targeted at key pain points

**Capabilities**

**Trace**
- Trace the location and status of food products upstream and downstream across the supply chain

**Certifications**
- Enable reliability and accountability with instant access to digitized records and documents

**Fresh Insights**
- Access real-time and aggregate supply chain data to extend product freshness and shelf life

**Third-party**
- Partner to expand functionalities and deliver new value across the food system

**Blockchain Technology**

**Food Supply Ecosystem**

**Information-sharing Platform**

**IBM Blockchain Platform**

**Hyperledger Fabric**
The effectiveness of the IBM Food Trust solution was demonstrated with a Walmart mango pilot.

**Pilot Test Case**

How long does it take to trace a package of sliced mangoes back to the farm?

**Supply Chain**

**Results**

Typical manual, mixed digital and paper-based method

- 6 days
- 18 hours
- 26 minutes

IBM Food Trust digital solution

2.2 seconds
Momentum is growing

Today, we have full in-production capability after over a year of testing:

- One of the **largest** non-crypto blockchain networks in the world
- **5M+** food products on retail shelves
- ~**1,200 SKUs/items** digitized, representing **4M+** transactions
- **200K** traces conducted to date
TradeLens
More than $16 trillion in goods are shipped across international borders each year.

80% of the goods consumers use daily are carried by the ocean shipping industry.

By reducing barriers within the international supply chain, global trade could increase by nearly 15% boosting economies and creating jobs.
Global Trade: Shipment case study

Analyzed the shipment of avocados from Mombasa to Rotterdam

We followed the shipment from Mombasa to Rotterdam and tracked the actors, people and information exchanges. The findings illustrated the complexity and costs inherent in today’s global supply chain.

![Graph showing number of people, actors, and information exchanges in the shipment process.]

© IBM Corporation, 2019
Inconsistent information across organizational boundaries and “blind spots” throughout the supply chain hinder the efficient flow of goods.

- Complex, cumbersome, and costly peer-to-peer messaging
- Manual, time-consuming, paper-based processes; high air courier expense and delays
- Risk assessments often lack sufficient information; clearance processes subject to fraud
- The administrative cost of handling a container shipment is comparable to the cost of the actual physical transport

Instant, secure access to end-to-end supply chain information; single source of the truth

- Assurance of the authenticity and immutability of digital documents; trusted cross-organizational workflows
- Better risk assessments and fewer unnecessary interventions
- Far lower administrative expenses and elimination of costs to move physical paper across international borders
- Estimated global savings from more efficient sharing of information: $27 billion
TRADELENS BLOCKCHAIN BUSINESS NETWORK

TRADELENS PLATFORM

Not exhaustive list of milestones managed by platform

STRUCTURED AND UNSTRUCTURED DOCUMENTS

Not exhaustive list of documents managed by platform

TRADELENS OVERVIEW

Introduction

15-May-19
# TRADELENS NETWORK OVERVIEW

## Enrichment Data
Supplements foundational data, in real-time and direct from the source
- Rapidly growing network
- Extensive port / terminal coverage

## Foundational Data
The core data needed to track and manage shipments end-to-end
- 20% of global container shipping volume
- Vast number of trade lanes covered

## Connected Logistics Members
60+

## Containers tracked
20M+

## TradeLens Platform
500M+ Events Captured

*Ocean carrier who are not network members can provide data as requested by their customers*
## CURRENT NETWORK STATUS – AS OF APRIL 2019

### Shipping Events

- **+ 500M per year**
- **+ 10M per week**

### Ports and Terminals

<table>
<thead>
<tr>
<th>Terminal Location</th>
<th>Operator</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algeciras, Spain</td>
<td>Port of Algeciras</td>
<td>◎</td>
</tr>
<tr>
<td>Algeciras, Spain</td>
<td>APM Terminals</td>
<td>◎</td>
</tr>
<tr>
<td>Apapa, Nigeria</td>
<td>Port of Algeciras</td>
<td>◎</td>
</tr>
<tr>
<td>Auckland, New Zealand</td>
<td>Port Connect</td>
<td>◎</td>
</tr>
<tr>
<td>Avonmouth, UK</td>
<td>MCP</td>
<td>◎</td>
</tr>
<tr>
<td>Bahrain</td>
<td>APM Terminals</td>
<td>◎</td>
</tr>
<tr>
<td>Barcelona, Spain</td>
<td>Port of Barcelona</td>
<td>◎</td>
</tr>
<tr>
<td>Bilbao, Spain</td>
<td>Port of Bilbao</td>
<td>◎</td>
</tr>
<tr>
<td>Brisbane, Australia</td>
<td>Patrick Terminals</td>
<td>◎</td>
</tr>
<tr>
<td>Buenos Aires, Argentina</td>
<td>APM Terminals</td>
<td>◎</td>
</tr>
<tr>
<td>Busan, South Korea</td>
<td>Port of Busan</td>
<td>◎</td>
</tr>
<tr>
<td>Callao, Peru</td>
<td>APM Terminals</td>
<td>◎</td>
</tr>
<tr>
<td>Cotonou, Benin</td>
<td>APM Terminals</td>
<td>◎</td>
</tr>
<tr>
<td>Elizabeth, NJ, USA</td>
<td>APM Terminals</td>
<td>◎</td>
</tr>
<tr>
<td>Felixstowe, UK</td>
<td>MCP</td>
<td>◎</td>
</tr>
<tr>
<td>Fremantle, Australia</td>
<td>Patrick Terminals</td>
<td>◎</td>
</tr>
<tr>
<td>Gothenburg, Sweden</td>
<td>APM Terminals</td>
<td>◎</td>
</tr>
<tr>
<td>Grangemouth, UK</td>
<td>MCP</td>
<td>◎</td>
</tr>
<tr>
<td>Gunsan, South Korea</td>
<td>KL-NET</td>
<td>◎</td>
</tr>
<tr>
<td>Halifax, Canada</td>
<td>Halterm Canada</td>
<td>◎</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Terminal Location</th>
<th>Operator</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hong Kong</td>
<td>Modern Terminals</td>
<td>◎</td>
</tr>
<tr>
<td>Houston, TX USA</td>
<td>Port of Houston</td>
<td>◎</td>
</tr>
<tr>
<td>Hull, UK</td>
<td>MCP</td>
<td>◎</td>
</tr>
<tr>
<td>Immingsh, UK</td>
<td>MCP</td>
<td>◎</td>
</tr>
<tr>
<td>Itajai, Brazil</td>
<td>APM Terminals</td>
<td>◎</td>
</tr>
<tr>
<td>Incheon, South Korea</td>
<td>KL-NET</td>
<td>◎</td>
</tr>
<tr>
<td>Izmir, Turkey</td>
<td>APM Terminals</td>
<td>◎</td>
</tr>
<tr>
<td>Kwangyang, South Korea</td>
<td>KL-NET</td>
<td>◎</td>
</tr>
<tr>
<td>Lazarro, Mexico</td>
<td>APM Terminals</td>
<td>◎</td>
</tr>
<tr>
<td>Liverpool, UK</td>
<td>MCP</td>
<td>◎</td>
</tr>
<tr>
<td>Los Angeles, CA, USA</td>
<td>APM Terminals</td>
<td>◎</td>
</tr>
<tr>
<td>Maasvlakte II, Netherlands</td>
<td>APM Terminals</td>
<td>◎</td>
</tr>
<tr>
<td>Manila, Philippines</td>
<td>ICTSI</td>
<td>◎</td>
</tr>
<tr>
<td>Melbourne, Australia</td>
<td>Patrick Terminals</td>
<td>◎</td>
</tr>
<tr>
<td>Montreal, Canada</td>
<td>MGTP</td>
<td>◎</td>
</tr>
<tr>
<td>Mobile, AL USA</td>
<td>APM Terminals</td>
<td>◎</td>
</tr>
<tr>
<td>Mumbai, Brazil</td>
<td>APM Terminals</td>
<td>◎</td>
</tr>
<tr>
<td>Napier, NZ</td>
<td>Napier Port Authority</td>
<td>◎</td>
</tr>
<tr>
<td>Newcastle, UK</td>
<td>MCP</td>
<td>◎</td>
</tr>
</tbody>
</table>

### Ocean Carriers

- **Maersk Line**
- **Safmarine**
- **Sealand**
- **Hamburg-Sud**
- **Pacific International Lines**
- **KMTC**
- **Seaboard**
- **Namsung**
- **Boluda Lines**
- **Zim**

### Government Authorities

- **Australia Home Affairs**
- **Bahrain Customs**
- **Canada Customs**
- **Dutch Customs**
- **Ghana / GCNET**
- **Saudi Arabia Customs**
- **Peru Customs**
- **Singapore Customs**
- **Turkey Customs**

### Inland Transportation

- **Ancotrans**
- **CN Rail**
- **IMCC**
Outline

Blockchain for Businesses

Technical Challenges

Smart Contract Generation
Considerations for a Production-Level Blockchain System

<table>
<thead>
<tr>
<th>Category</th>
<th>Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functionality</td>
<td>• Does the system provide all the necessary functionalities?</td>
</tr>
<tr>
<td>Reliability</td>
<td>• Can the system achieve necessary availability? Is it resilience to faults? • Can the system recover from faults? • Is it resilient to wide-area disasters? Can it recover from backups?</td>
</tr>
<tr>
<td>Operability</td>
<td>• How much data get accumulated, and where? • Can data be archived, deleted, or backed-up? • Can users or peers be added or removed? • Can their permissions dynamically changed?</td>
</tr>
<tr>
<td>Performance</td>
<td>• What is the transaction throughput and latency? • Can it handle peak transactions? • Will performance degrade after a large amount of data get accumulated? • Is it scalable? (number of peers and users)</td>
</tr>
<tr>
<td>Maintenanceability</td>
<td>• Can smart contracts and applications be safely updated? • Can the blockchain platform be updated dynamically?</td>
</tr>
<tr>
<td>Security</td>
<td>• Is it resilient to tampering or impersonation? • Can data privacy be protected? • Can it enforce access control in an appropriate manner? • How to deal with compromising crypto system</td>
</tr>
<tr>
<td>Non-Technical</td>
<td>• Business Value, Business Scheme, Governance, Regulatory Compliance, etc…</td>
</tr>
</tbody>
</table>
Technical Challenges

• Finality and 51% attacks
• Privacy vs. Integrity
• Scalability / Performance
• Safety of Smart Contract
What is “Finality”?  

• Finality is a concept used in the financial industry.  
• Once a settlement or payment is completed, it is “irrevocable and unconditional”  

• **Finality** of payment refers to the instant that a payment to another party is completed, at which point the receiving institution has irrevocable access to the money. This is more commonly referred to as the moment when funds are "good" in an account. (Investopedia)
Lack of Finality in PoX

Multiple participants may add different blocks simultaneously to cause a fork.

Longer block will become valid

Transactions in invalidated block will become invalid
51% attack is increasing

- High-value altcoins with smaller hash power are targeted
- When coin price drops
  -> reward to minors reduces
  -> mining ROI reduces
  -> minors withdraw
  -> less hash power

- Bitcoin Gold
- Verge
- Zencash
- Litecoincash
- Ethereum Classic
In blockchain, security becomes more important than traditional distributed systems, because multiple entities with conflict-of-interest have transactions.

**Integrity Requirements**
- Agreement on transaction validity
- Tamper evidence
- Finality

**Privacy / Confidentiality Requirements**
- Anonymity to third parties
- Anonymity between counterparties
- Confidentiality of transaction content
- Confidentiality about existence of transaction
- Confidentiality of state

Integrity vs. Privacy: Cryptocurrencies

- Examples: Bitcoin, Altcoins
- Typically, UTXO + PoW (or variant) in a public network

- Integrity
  - All the transactions are broadcasted to all the participants
  - Everybody can participate in validation
  - Probabilistic agreement, *no finality*

- Privacy – only anonymity
  - Participants are *anonymous*
  - But transaction *content is visible* to everybody
Integrity vs. Privacy: Enterprise Use Cases

• Examples: Commercial Agreement or Financial Contract executed as Smart Contracts
• Typically implemented on permissioned blockchain with smart contracts

• Integrity
  – Participants are identified and authenticated
  – Transactions should be validated by at least by all the stakeholders

• Stronger Privacy is Required
  – Transacting parties should be anonymous to third parties
  – Content of contract and transactions should not be visible to third parties
Privacy Protection on Permissioned Blockchain

Private Data Collection (Hyperledger Fabric)

- Only public transactions and hash value of private transactions are broadcasted to all the participants
- Private transactions are send to the limited participants
Scalability / Performance

• Public Blockchain
  – Longer latency in general
  – Easier to scale with a large number of nodes
    • Bitcoin: approx. 10,000 nodes

• Permissioned Blockchain
  – Higher throughput and shorter latency in general
  – Less scalable on the number of nodes (e.g., 10s ~ 100s)

• Real-World Requirement Examples
  – VISA: 1700 TPS/average, 24000/max *1
  – Financial market post-trade: 5,000+TPS /peak

*1: Some data points are approximations and may vary.
Distributed Consensus Protocols

- **State machine replication**
  - Make agreement among nodes without needing to trust any single node
  - Total order, validity, and integrity

- **Tolerant to failure**
  - Communication failures
  - Crash failure
  - Byzantine Failure

**Known Protocols**

- **PBFT (Practical Byzantine Fault Tolerance)**
  - Tolerant to up to 1/3 of faulty nodes
- **Raft, PAXOS**
  - Crash tolerance only

**Example: PBFT (Practical Byzantine Fault Tolerance)**

- One node becomes a leader
- Leader initiate consensus making
- Each node exchange messages each other, and reaches the consensus after 3 phases
- When leader fails, a new leader is elected
Typical **Order-Execute** architecture

- Sequential execution is required
- No consensus over the execution result
- Smart contract has to be deterministic

**Novel Execute-Order-Validate architecture of Hyperledger Fabric**

- Parallel Execution
- Makes consensus over the execution result

---

**Consensus architecture of Hyplerledger Fabric V.1+**


© 2019 IBM Corporation
never been a bottleneck in our experiments. The validation phase, as ordering with Kafka ordering service has identify bottlenecks. Our experiment focused on the validation coarse-grained latency staging of block validation to better peers run on 4, 8, 16 and 32 vCPU VMs, while also doing throughput, we performed a set of experiments in which 4 cryptographic operations. To estimate the impact of CPU on actions because they carry CB certificates. This is an avenue large because they carry certificate information. Besides, and 4.33kB for transactions, i.e., the average transaction size is 3.06kB for the 2MB-blocks contained 473 transactions. In every experiment, client threads submit at least 500k transactions. In particular, transactions of Fabcoin are larger than transactions. In general, transactions in Fabric are

Figure 6.
Impact of block size on throughput and latency.
We can observe that throughput does not significantly
improve beyond a block size of 2MB, but latency gets worse (e2e) latency. Therefore, we adopt 2MB as the block size to evaluate the impact of block size on performance, we ran experiments varying block size from 0.5MB to 4MB. Results to fix the block size for subsequent experiments, and a parameter that impacts both throughput and latency is block size. Choosing the block size.

Figure 7. Scalability / Performance

Performance Evaluation of Hyperledger Fabric V1.1

Achieved 3,560TPS
Latency <100ms

Use case: Fabcoin / Fabtoken
UTXO implementation on Hyperledger Fabric
– Mint and Spend transactions are recorded on Key-Value Store
– Written in Golang


*1: https://usa.visa.com/run-your-business/small-business-tools/retail.html

© 2019 IBM Corporation
Safety of Smart Contracts

• Smart Contracts are just “Computer Programs” and takes cost and time to develop
• It is difficult to verify that the executable smart contract correctly represents the business-level agreements
• Risks of bugs and vulnerabilities
  – e.g., the DAO incident
**Objective:** Streamlining smart contract life cycle from contract documents to execution and monitoring

- Formal temporal logic based approach to define the *correct* behavior of contracts as a specification
  - Automatically generated smart contracts always comply with the specification
  - Monitor detects any deviation from the specification, e.g., anomalous behavior or incidents
Example: Accumulator (derivative product)

- **Accumulators** are financial derivative products sold by an issuer (seller) to investors (the buyer) that require the buyers to buy shares of some underlying security at a predetermined **strike price**
- If **closing price < strike price**, then number of the shares to be bought is multiplied by leverage
- If **closing price > knock-out price**, then contract terminate early, before the maturity date.

![Graph showing closing price of the reference share and number of shares the buyer needs to buy](image-url)

- **Leverage**: depends on closing price of the day
- **Early termination**
- **Knock-out price**
- **Strike price**
Example: Accumulator (derivative product)

The investor have to buy a specific stock (or security) at a predefined price range (between *strike price* and *knock-out price*) within a contract period.

```plaintext
//cur_date = get_date()
bought_shares = 0
_init(EXPIRY_DATE)
_init(KO_PRICE)
_init(STRIKE_PRICE)
_init(NO_OF_SHARES)
_init(GUARANTEED_SHARES)
_init(LEVERAGE)

DayOpened
  day_close(price)

DayClosed
  [price > KO_PRICE]
  entry/
  bought_shares = max(bought_shares, GUARANTEED_SHARES)
  send(bought_shares, STRIKE_PRICE)

TerminatedByKnockOut
  [price >= STRIKE_PRICE]
  bought_shares += NO_OF_SHARES

TerminatedNormally
  [cur_date == EXPIRY_DATE]
  entry/
  send(bought_shares, STRIKE_PRICE)

s3
  [price <= KO_PRICE]
  day_open/
  cur_date += 1

s2
  [price < STRIKE_PRICE]
  bought_shares += LEVERAGE * NO_OF_SHARES

```

suppose that this workflow is instantiated in daytime
Contract Document Template

**Contract = Master agreement + Term sheet**

Term sheet includes variable parameters that are agreed in each signing.
Contract Document Template

**Contract = Master agreement + Term sheet**

Term sheet includes variable parameters that are agreed in each signing

Contract Logic Template

Pre-defined for each document template

Protocol

```plaintext
// knock-out case
init; <$CLOSING_EVENT>;
(_not_knockout; _not_terminate;
 day_open; <$CLOSING_EVENT> )*;
_knockout
...
on _knockout when {
  <$KNOCKOUT_EVENT> }
};
...
```

Template of the Contract Specification is encoded in DSL4SC, with variable part as placeholders
Example: Contract Template

Term sheet is a document template that consists of variable fields where each field data is defined in controlled natural language (CNL).

Example: defines events and its occurrence conditions

```
[ ] Event: [ ]

Knock-out Date: The first Valuation Date from and including the Trade Date to and including

"Knock-out" "The closing price is greater than or equal to the Knock-out Price."

Object := "The Closing Price" | "the Knock-out Price"
```

Example Grammar for CNL

```
KNOCKOUT_PRICE := Number
Number := [0-9]*

KNOCKOUT_EVENT := Stmt;
Stmt := Object Predicate Object
Predicate := "is greater than or equal to"
| "is greater than" | "equal to" | "is less than"
Object := "The Closing Price" | "the Knock-out Price"
```
Smart Contract Generation

Field Data

Contract Logic Template

protocol // knock-out case
    init; day_close;
    (_not_knockout;
     _not_terminate;
     day_open; day_close; result);
    _knockout
    ...
    on _knockout when {
        <$KNOCKOUT_EVENT>
    };

extract

transform

Contract Document

Formal contract spec in Temporal Logic-based language (DSL4SC)

Statechart (SCXML)

Chaincode (Golang)

protocol // knock init; <$CLOSING_EVENT> ;
    (_not_knockout;
     _not_terminate;
     day_open; <$CLOSING_EVENT> ; result);
    _knockout
    ...
    on _knockout when {
        <$KNOCKOUT_EVENT>
    };

// mycc.go
    func Invoke(...) {
        ...
    }
    func main(...) {
        ...
    }

events

monitor

HYPERLEDGER

Fabric
**DSL4SC: LDL\(_f\) -based DSL**

DLS4SC consists of **specification** and **implementation**

<table>
<thead>
<tr>
<th>Speculative Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>contract</strong>: protocolDecl propertyDecl ruleDecl</td>
</tr>
<tr>
<td><strong>protocolDecl</strong>: protocol protocol ;</td>
</tr>
<tr>
<td><strong>protocol</strong>: event_name</td>
</tr>
<tr>
<td><strong>propertyDecl</strong>: property (ldl_formula ; )†</td>
</tr>
<tr>
<td><strong>ruleDecl</strong>: rule (rule ; )†</td>
</tr>
<tr>
<td><strong>rule</strong>: except? on event_name (, event_name)*</td>
</tr>
<tr>
<td><strong>condition</strong>: ldl_proposition</td>
</tr>
<tr>
<td><strong>action</strong>: ensure ldl_proposition</td>
</tr>
<tr>
<td><strong>ensure</strong>: preserve (var_name (, var_name)*)</td>
</tr>
<tr>
<td><strong>preserve</strong>: action (, action)†</td>
</tr>
</tbody>
</table>

### Protocol (specification)
- Regular-patterns of **events** to be processed by a contract

### Property (specification)
- Temporal constraints on the **states** that the contract need to satisfy
  - e.g., some properties *always / never* hold, or *eventually* hold

### Rule (Specification + implementation)
- **ECA** (Event-Condition-Action) rules that define how to react to events
- Detailed implementation can be defined in a conventional language such as JavaScript or Golang
Example: Web Browser Interface to Control the Contract

The table view to list and control assets (parts, docs, contracts, stocks, …) managed by blockchain.

The State Chart view to visualize the contract logic and the current data in the State DB.
## SCDT: Value Propositions

### Today’s pain points

| Smart contracts need to be developed by human engineers, and requires time and efforts. |
| Smart Contracts are error prone, and there are risks of financial losses due to vulnerabilities (e.g., DAO incident) |
| Business/Legal people wants to make agreement on a natural language, but it is difficult for them to verify consistency with executable smart contracts |

### SCDT’s Value

| Smart contracts are automatically generated, reducing the time and efforts for development |
| Generated smart contracts always satisfy the safety and liveness properties defined in DSL4SC. |
| A user can define Ts&Cs in controlled natural language which can be translated into smart contract using a template-based approach |
Conclusions

• Serious production systems are emerging and growing
• Still many challenges to be addressed
  – Technical and non-technical challenges

• DSL4SC is published as open source
  – https://ldltools.github.io
References

• Papers
    • http://cse.stfx.ca/~cybermatics/2018/Proceedings/pdfs/iThings!GreenCom!CPSC!SmartData!Blockchain!Cybermatics2018-1Q1rrimpxFNYcx89cHRQN/3Jk4hkhjsgsMXXCGQkelaj/6qha5FUSMJfyaP0Ps8tXaQ.pdf
    • http://cse.stfx.ca/~cybermatics/2018/Proceedings/pdfs/iThings!GreenCom!CPSC!SmartData!Blockchain!Cybermatics2018-1Q1rrimpxFNYcx89cHRQN/3kYgGavps1k80vVipnumxy5/OFyc6SSHw9AjQtik2k4vt.pdf
    • https://ieeexplore.ieee.org/document/8645646
• DSL4SC Open Source Tool Documentation and GitHub repository
  • https://ldlttools.github.io
  • https://github.com/ldlttools/
Thank You
Merci
Gracias
Danke
धन्यवाद
謝謝
ありがとうございます