Blockchain: Distributed Event-based Processing in a Data-Centric World

A Perspective from Business Process Management

Richard Hull (IBM Research)

Drawing on many discussions with: Vishal Batra, Yi-Min Chee, Yaoliang Chen, Michael Coblenz, Pralhad Deshpande, Alin Deutsch, Terry Heath, Yuliang Li, Yuichi Nakamura, Krishna Ratakonda, Yunjie Qiu, Jianwen Su, Noi Sukaviriya, Shin Saito, Takaaki Tateishi, Victor Vianu

21 June 2017 @ DEBS in Barcelona

Final version

How do organizations collaborate in today's world?

By exchanging documents, in many cases on paper:

- Trade finance: letter of credit, export documents (eg., SWIFT MT700,…)
- Logistics/Supply Chain: Purchase Order (EDI 850), Load Tender (EDI 204), Tender Response (EDI 990), …
- Mortgage & Loan processing: many scanned PDF's
- …

Are these simply messages exchanged between services?

- No, because they persist, and are referred to at later times
- In fact, the documents refer to an implicit body of shared data

Blockchain (for businesses) will dramatically streamline data/document sharing

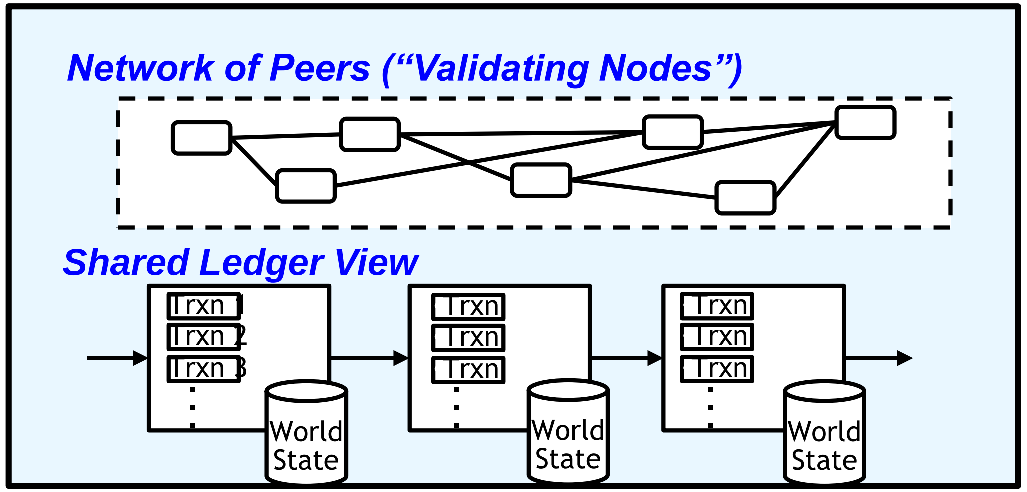
- Blockchain provides a trusted repository for holding persistent shared data
- Blockchain enables selective privacy
- Blockchain will enable deep business-level efficiencies

Why is this seismic shift in business collaboration relevant to the Services & Distributed Event-Processing communities?

Blockchain is fundamentally a Distributed Event-based Processing Framework at 2 layers

“Smart Contracts”

Logical Abstraction Separation



Programming Layer

- Specification of logical behavior, e.g., for business collaborations
- *Event-driven transition system*

Foundational Layer

- Encryption
- Consensus algorithms
- Distributed copies of data that are kept synchronized
- Nonrepudiable
- Enables selective privacy
- *Event-driven transition system*

- Reminiscent of “Physical Data Independence” in databases

One broad area for Services & DEBS Research contributions:

Business-Level "Smart Contract" Language and Framework

- Blockchain today is programmed using Turing-complete languages such as GOLANG, Java, ???
- Some domain-specific languages are emerging …

We need

- ***Principled approach*** for event-driven, modular, data-centered services
- ***Domain-specific language*** aimed at business users
- ***Workbenches for business analysts*** to understand, create, test, modify the "smart contracts" that run on Blockchain
- ***Foundational understanding*** of biz-level "smart contracts"

Agenda

- Blockchain enables a new level of trust & communication
- What is Blockchain, and why is it useful for Business Collaborations?
- Logical separation between Blockchain mechanics and Biz-level programming
- Artifact-centric paradigm as starting point for Business Collaboration Language
- Research challenge areas
  - Language design
  - Reasoning about artifacts
  - Relationship to natural language contracts
- Conclusions

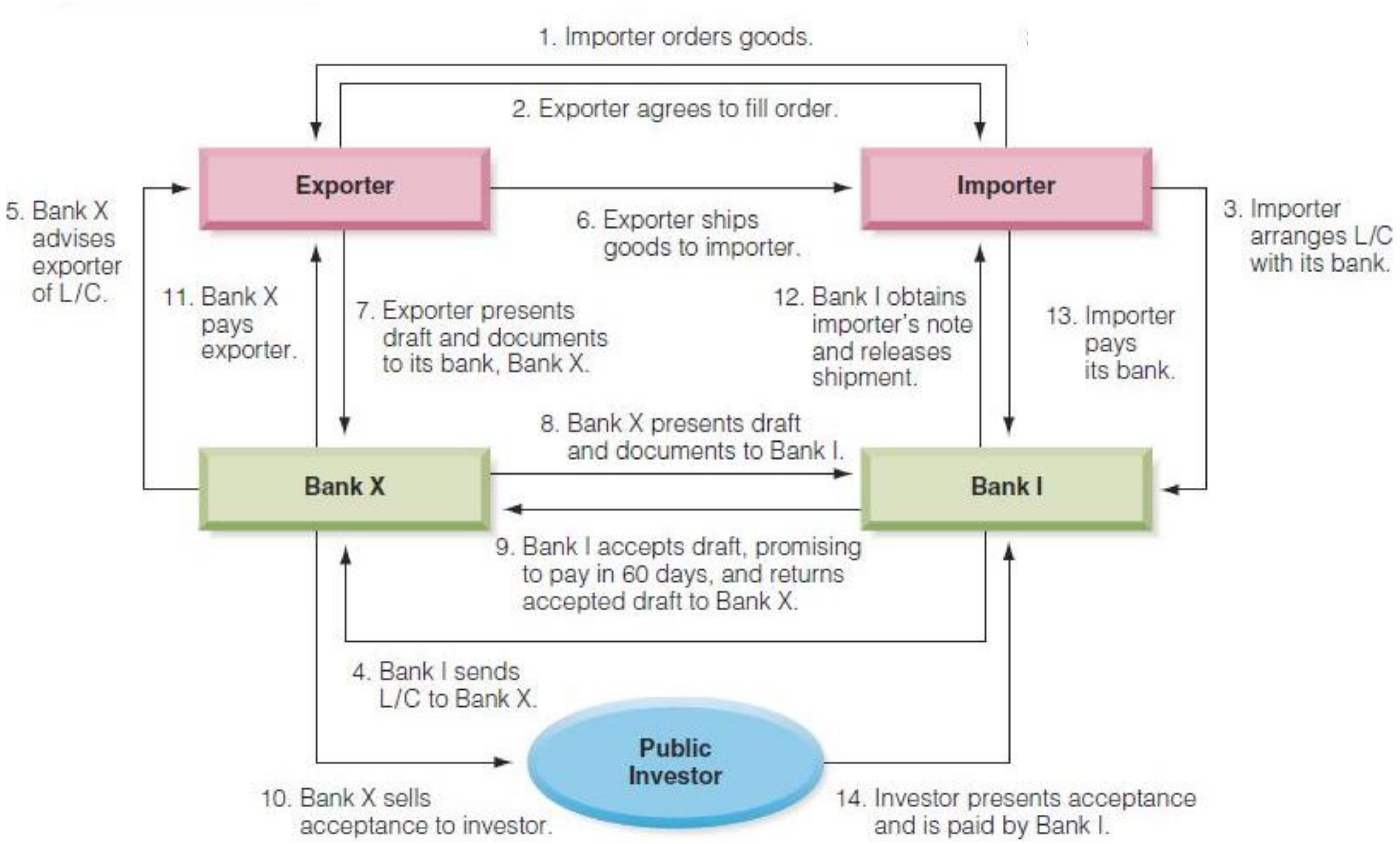
Caveat

This field is still in its infancy

This talk is mainly raising questions

Example from International Trade Finance

- Suppose that a company in Kenya is exporting pineapples to an importer in Rotterdam …

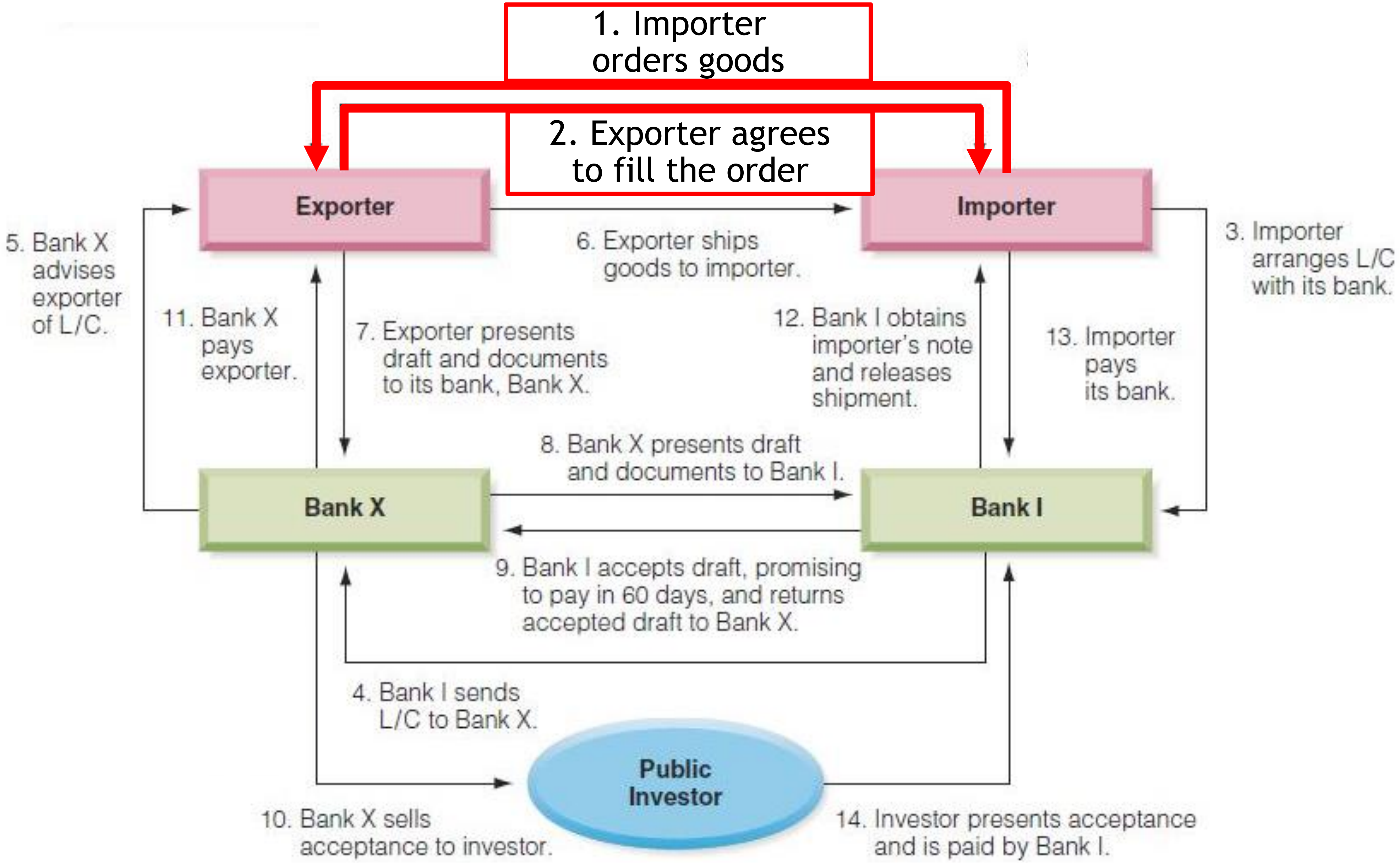

1. Importer orders goods.
2. Exporter agrees to fill order.
Exporter
Importer
3. Importer arranges L/C with its bank.
5. Bank X advises exporter of L/C.
6. Exporter ships goods to importer.
11. Bank X pays exporter.
7. Exporter presents draft and documents to its bank, Bank X.
12. Bank I obtains importer's note and releases shipment.
13. Importer pays its bank.
8. Bank X presents draft and documents to Bank I.
Bank X
Bank I
9. Bank I accepts draft, promising to pay in 60 days, and returns accepted draft to Bank X.
4. Bank I sends L/C to Bank X.
Public Investor
10. Bank X sells acceptance to investor.
14. Investor presents acceptance and is paid by Bank I.

- At least 4 parties, often more
  - Exporter
  - Exporter's Bank
  - Importer's Bank
  - Importer
  - There may be 10's of parties
- Kinds of documents
  - Order
  - Letter of Credit
  - Export documents
  - Draft
  - …
- Today
  - Some documents communicated electronically
  - Other documents sent by air courier

From "International Financial Management" by Jeff Madura

Example from International Trade Finance

- Suppose that a company in Kenya is exporting pineapples to an importer in Rotterdam …



1. Importer orders goods

2. Exporter agrees to fill the order

Exporter

Importer

Bank X

Bank I

Public Investor

3. Importer arranges L/C with its bank.

4. Bank I sends L/C to Bank X.

5. Bank X advises exporter of L/C.

6. Exporter ships goods to importer.

7. Exporter presents draft and documents to its bank, Bank X.

8. Bank X presents draft and documents to Bank I.

9. Bank I accepts draft, promising to pay in 60 days, and returns accepted draft to Bank X.

10. Bank X sells acceptance to investor.

11. Bank X pays exporter.

12. Bank I obtains importer's note and releases shipment.

13. Importer pays its bank.

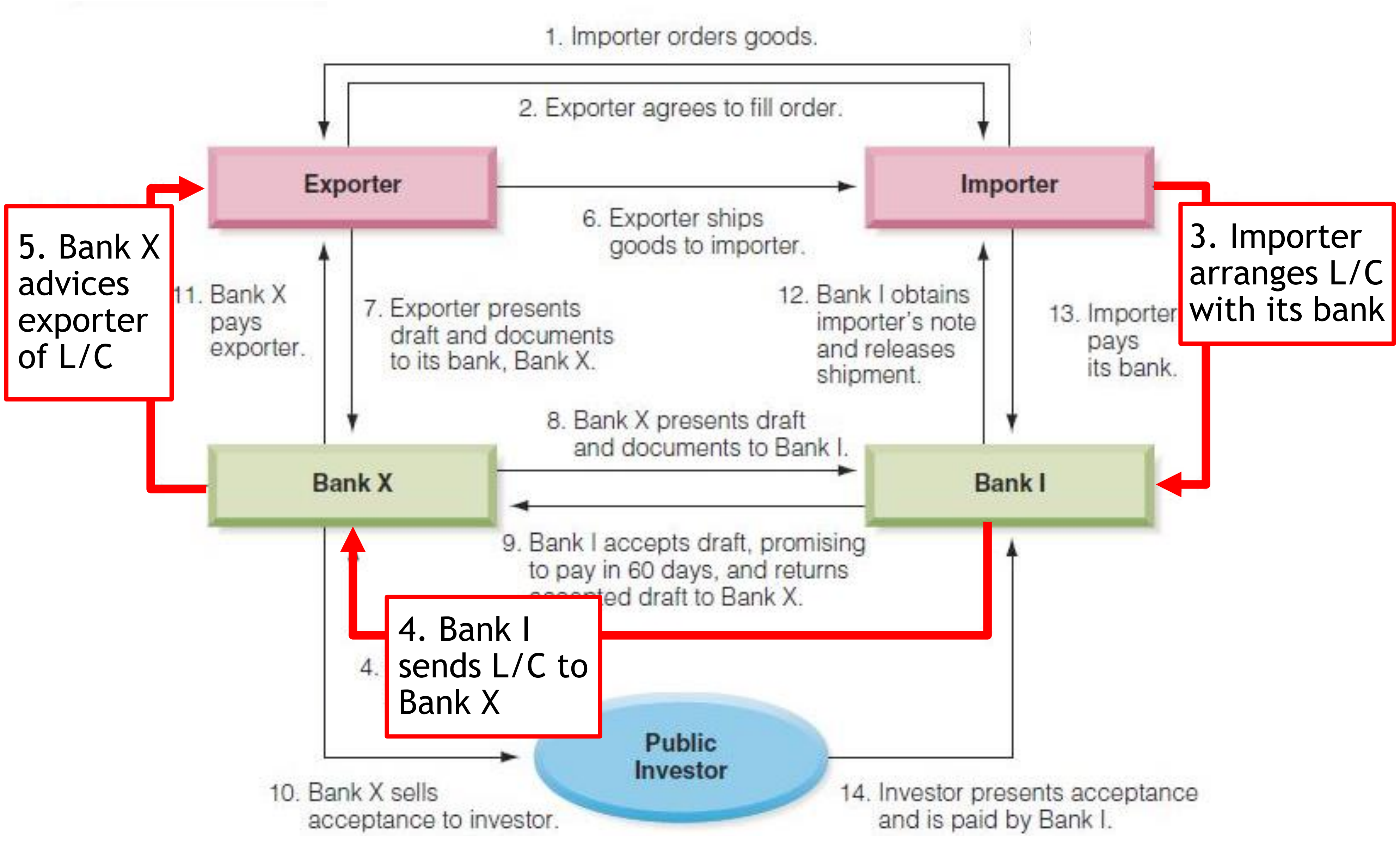
14. Investor presents acceptance and is paid by Bank I.

- At least 4 parties, often more
  - Exporter
  - Exporter's Bank
  - Importer's Bank
  - Importer
- Kinds of documents
  - Order
  - Letter of Credit
  - Export documents
  - Draft
- Today
  - Some documents communicated electronically
  - Other documents sent by air courier

From "International Financial Management" by Jeff Madura

Example from International Trade Finance

- Suppose that a company in Kenya is exporting pineapples to an importer in Rotterdam …



1. Importer orders goods.

2. Exporter agrees to fill order.

Exporter

Importer

3. Importer arranges L/C with its bank

4. Bank I sends L/C to Bank X

5. Bank X advices exporter of L/C

6. Exporter ships goods to importer.

7. Exporter presents draft and documents to its bank, Bank X.

8. Bank X presents draft and documents to Bank I.

Bank X

Bank I

9. Bank I accepts draft, promising to pay in 60 days, and returns accepted draft to Bank X.

10. Bank X sells acceptance to investor.

11. Bank X pays exporter.

12. Bank I obtains importer's note and releases shipment.

13. Importer pays its bank.

14. Investor presents acceptance and is paid by Bank I.

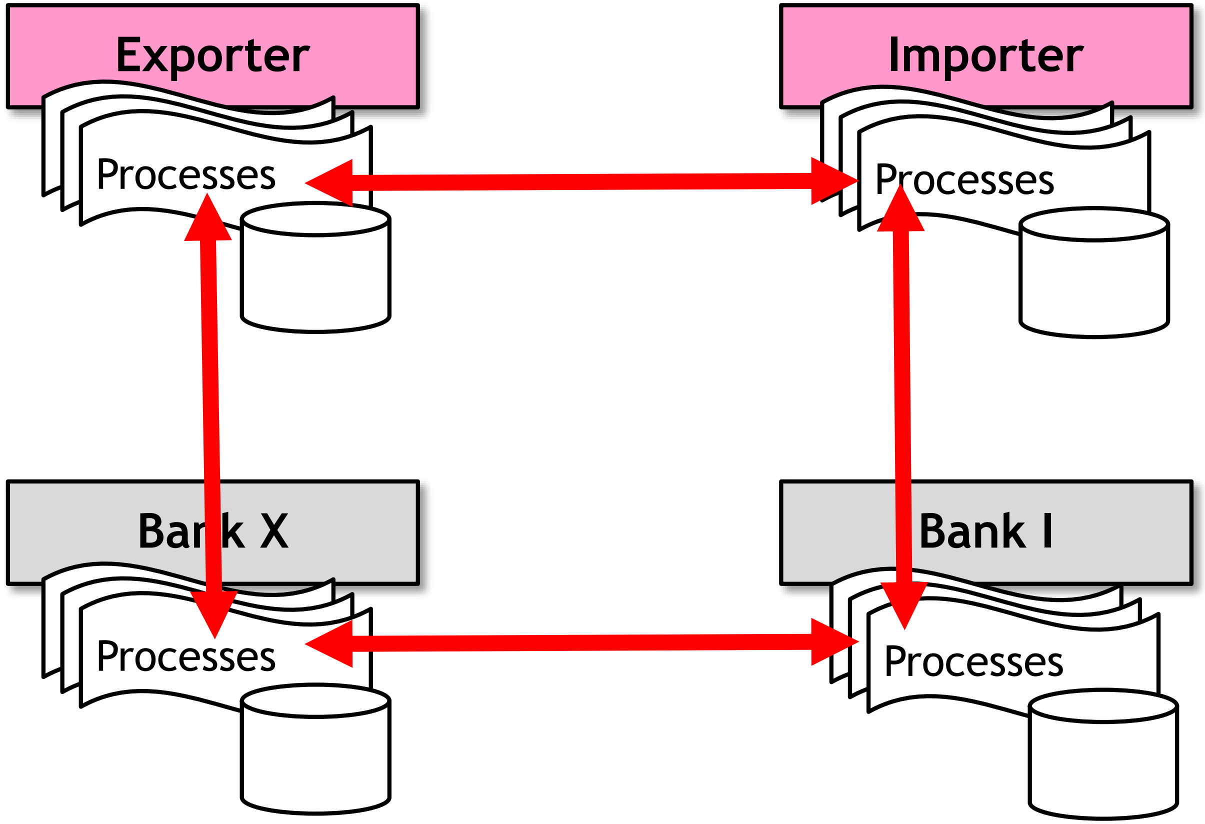
Public Investor

4.

- At least 4 parties, often more
  - Exporter
  - Exporter's Bank
  - Importer's Bank
  - Importer
- Kinds of documents
  - Order
  - Letter of Credit
  - Export documents
  - Draft
- Today
  - Some documents communicated electronically
  - Other documents sent by air courier

From "International Financial Management" by Jeff Madura

Before Blockchain

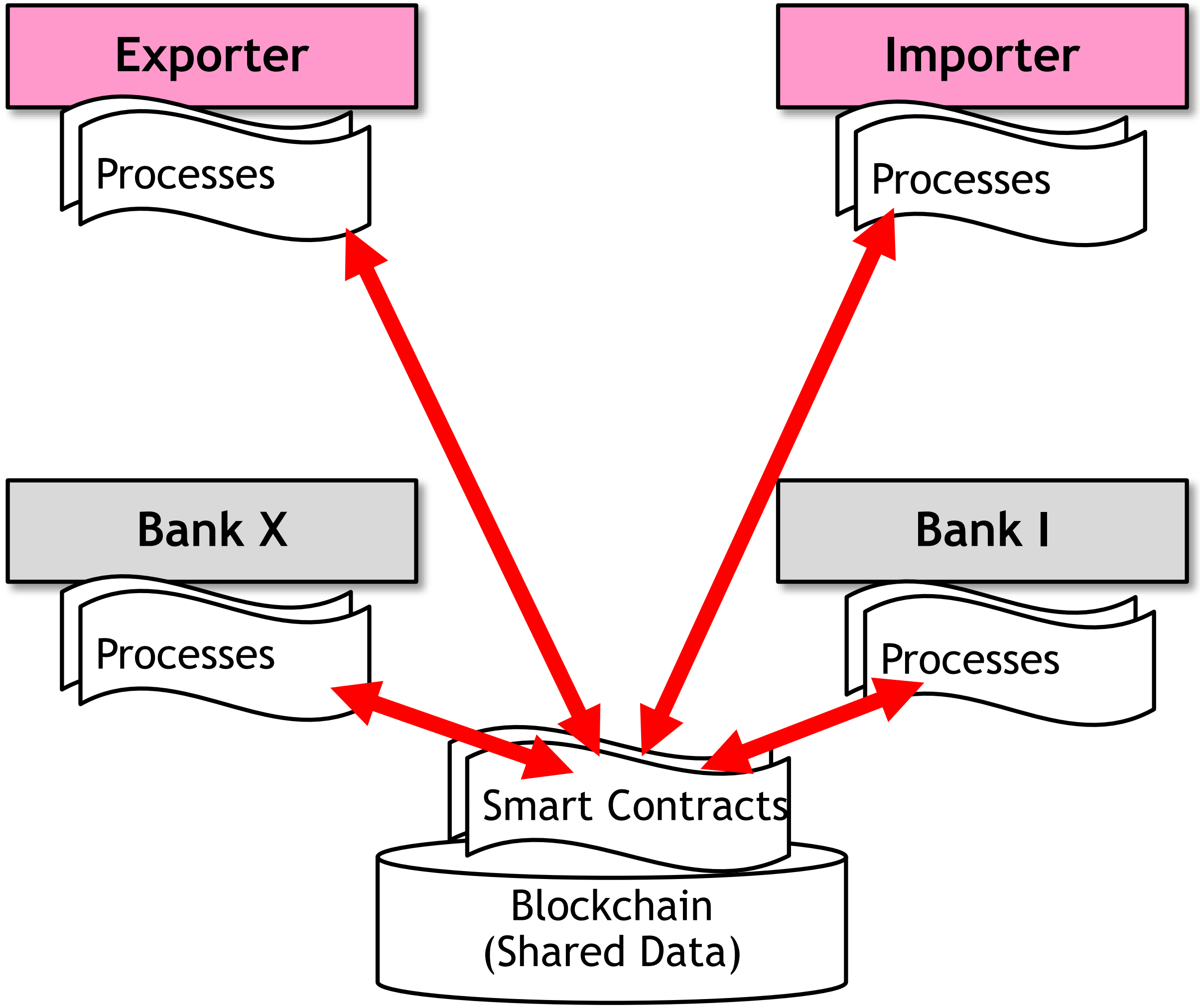


- Private copies of collaboration data

  → Disputes can take month+ to resolve
- Private copies of collaboration processing logic

  → Trust is based on binary relationships

With Blockchain



- Single shared copy of collaboration data

  → Disputes can be resolved in a day
- Single shared copy of collaboration processing logic

  → Trust becomes based on broadly visible shared data

Many application areas

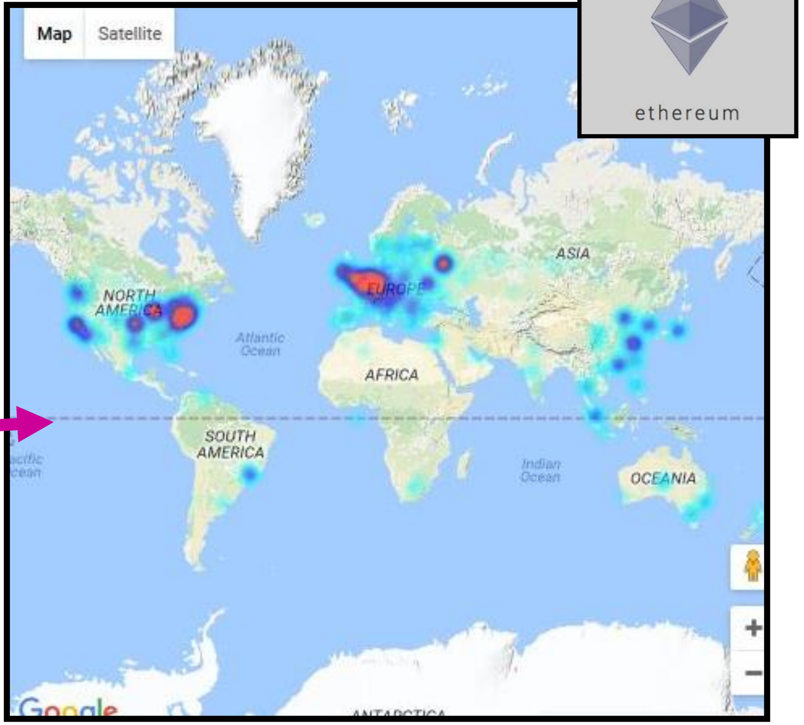
- Trade Finance
  - Trust between numerous parties, dispute resolution
- Supply chain/logistics
  - Non-disputable order tracking, dispute resolution
  - Important to both advanced and developing countries
- Mortgage processing
  - Capture machine readable data once; From redundant paper copies to single source of truth
- Certified Emissions Reduction (CER)
  - Enabling manufacturers to certify that they are producing product with low carbon footprint
- Food supply
  - Provenance from farm to fork
- Healthcare
  - More solid, robust basis for electronic health records
- Education (especially in developing countries)
  - Accurate, non-disputable student & teacher records
- …

Agenda

- Blockchain enables a new level of trust & communication
- What is Blockchain, and why is it useful for Business Collaborations?
- Logical separation between Blockchain mechanics and Biz-level programming
- Artifact-centric paradigm as starting point for Business Collaboration Language
- Selected research challenge areas
- Conclusions

A highly selective & brief history of Blockchain

- Bitcoin
  - Introduces Blockchain paradigm as basis for a crypto currency
  - Sole focus is on possession/transfer of Bitcoins
  - Privacy guaranteed for currency holders
  - Exchanges to trade Bitcoins for state-provided currencies ($, €, ¥, …)
- Etherium – a Swiss nonprofit, launched in 2014
  - General purpose, custom built Blockchain: ~7000 nodes
  - Crypto currency is called "Ether"
  - Framework includes notion of "fuel" or "gas money" – pay for transactions along the way
  - Broad usage, including by consortium including Microsoft for B2B collab.
- "The DAO" hack
  - A Distributed Autonomous Organization (DAO) can be set up on Etherium
    - Participants can contribute funding, and collectively vote on investments
  - "The DAO" launched on April 30, 2016, by German company Slock.it
    - By May 27 the DAO at raised $150M
  - An attacker drained 3.6M ether, worth about $70M, by June 18
  - Value of ether dropped from $20 to $13
- HyperLedger
  - Launched by the Linux Foundation – Dec 2015
  - 30 founding members, including: Accenture, Cisco, Digital Asset Holdings, Fujitsu, IBM, Intel, J.P. Morgan, R3, SWIFT, Wells Fargo, …



ethereum

http://ethernodes.org/network/1

- Etherium Blockchain itself did not show vulnerability nor hacking
- The smart Contract of "The DAO" was hacked

HYPERLEDGER

Blockchain 101 (with bias towards Hyperledger) (1 of 3)

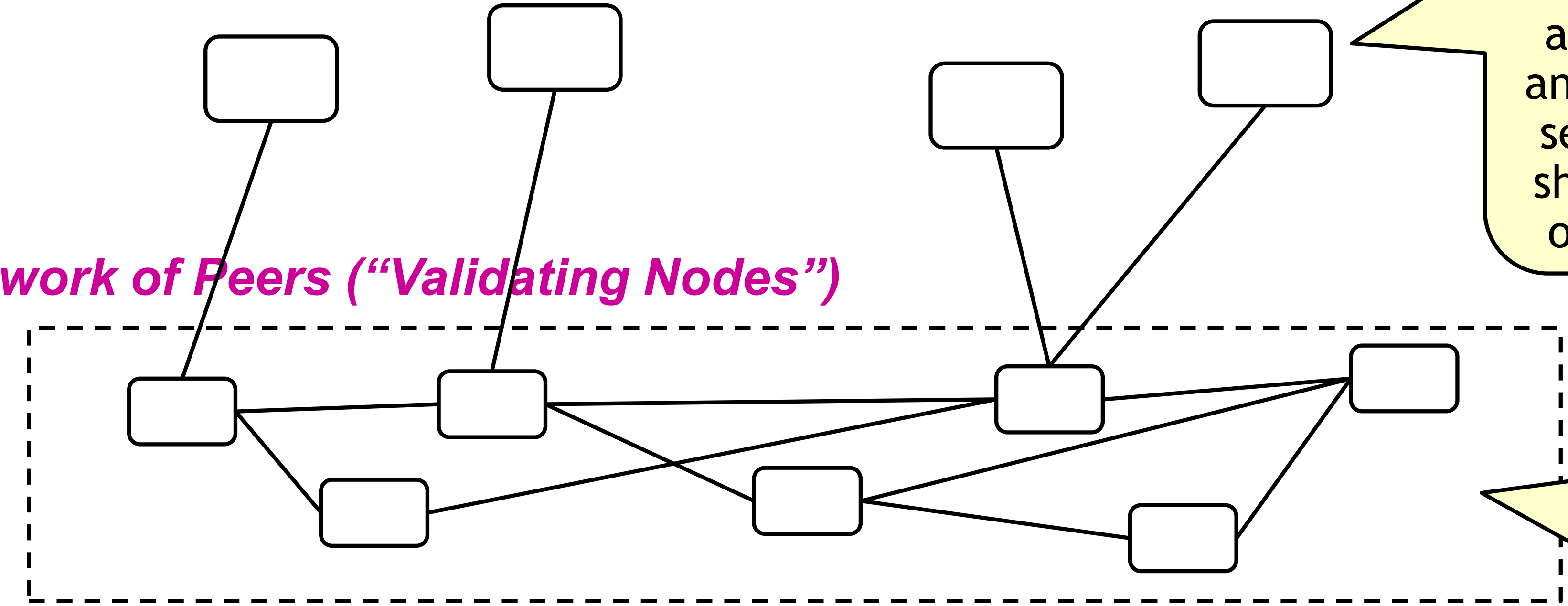
- A blockchain provides
  1. High reliability
  2. Shared single source of truth
  3. Trusted
  4. Selective privacy
  5. Non-repudiable data updates

- A blockchain consists in a network of servers
  - They may not trust each other at level of individuals
- Blockchain network supports ACID transactions
  - Consensus algorithm, such as Practical Byzantine Fault Tolerance (PBFT)
- Blockchain network supports selective privacy
  - Deep usage of encryption technologies
  - Selective access to data and service calls
  - (Often, the “smart contracts” are broadly visible)

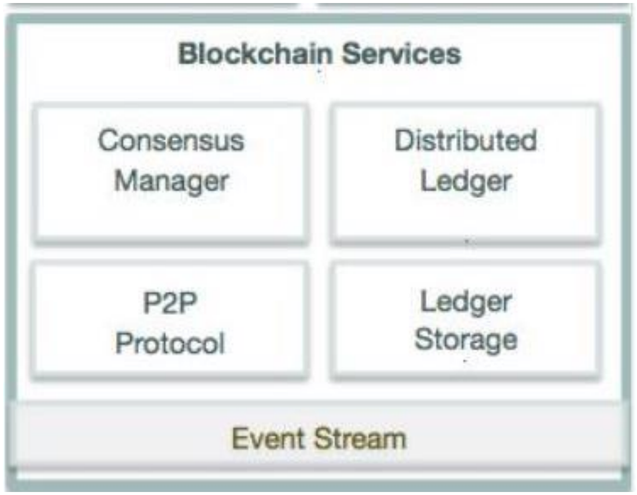
Blockchain 101 (with bias towards Hyperledger) (2 of 2)

Participants (executing on behalf of businesses)

Network of Peers ("Validating Nodes")



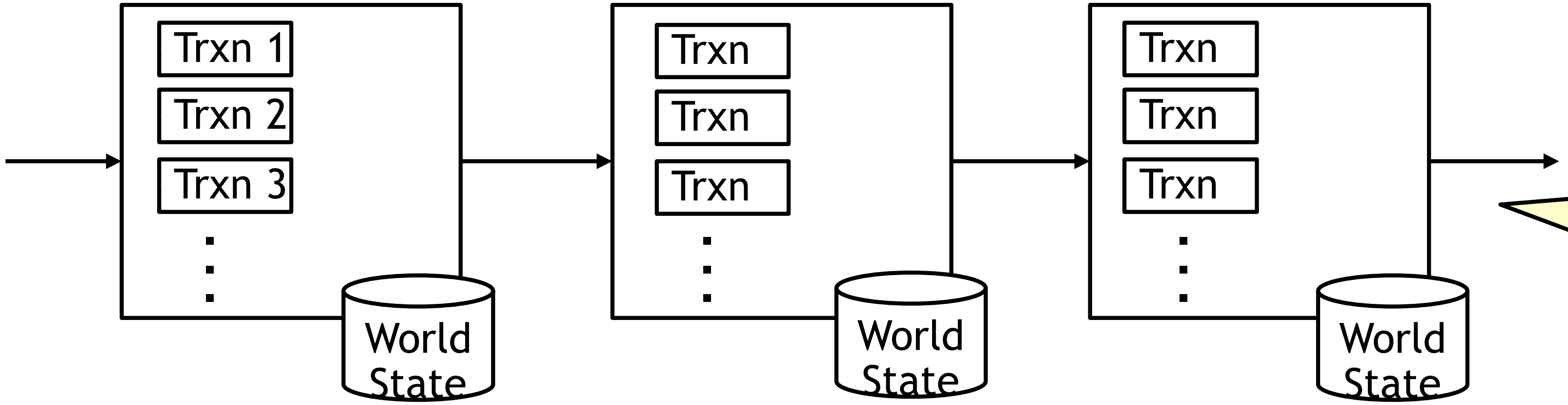
A participant can connect to a single peer, and will always see the single shared version of the ledger



http://www.the-blockchain.com/docs/Hyperledger%20Whitepaper.pdf

After each round of consensus, each peer holds a replica of the ledger

Shared Ledger View



- A "chain" of "blocks"
- The sequence of blocks is the shared "ledger"

Two types of txns

- Code Deploying
- Code Invoking

Blockchain 101 (with bias towards Hyperledger) (3 of 3)

- What makes Hyperledger different?
  - No built-in crypto currency
  - Cost of processing & data storage is not of major concern
    - Smaller number of peers
  - Anticipation of many Blockchain networks – spectrum including
    - Some more public
    - Some more private
  - All of the nodes are white-listed within a Blockchain network
    - Transactors are granted an identity by an issuing authority
  - Modular consensus
    - Consensus algorithms are pluggable

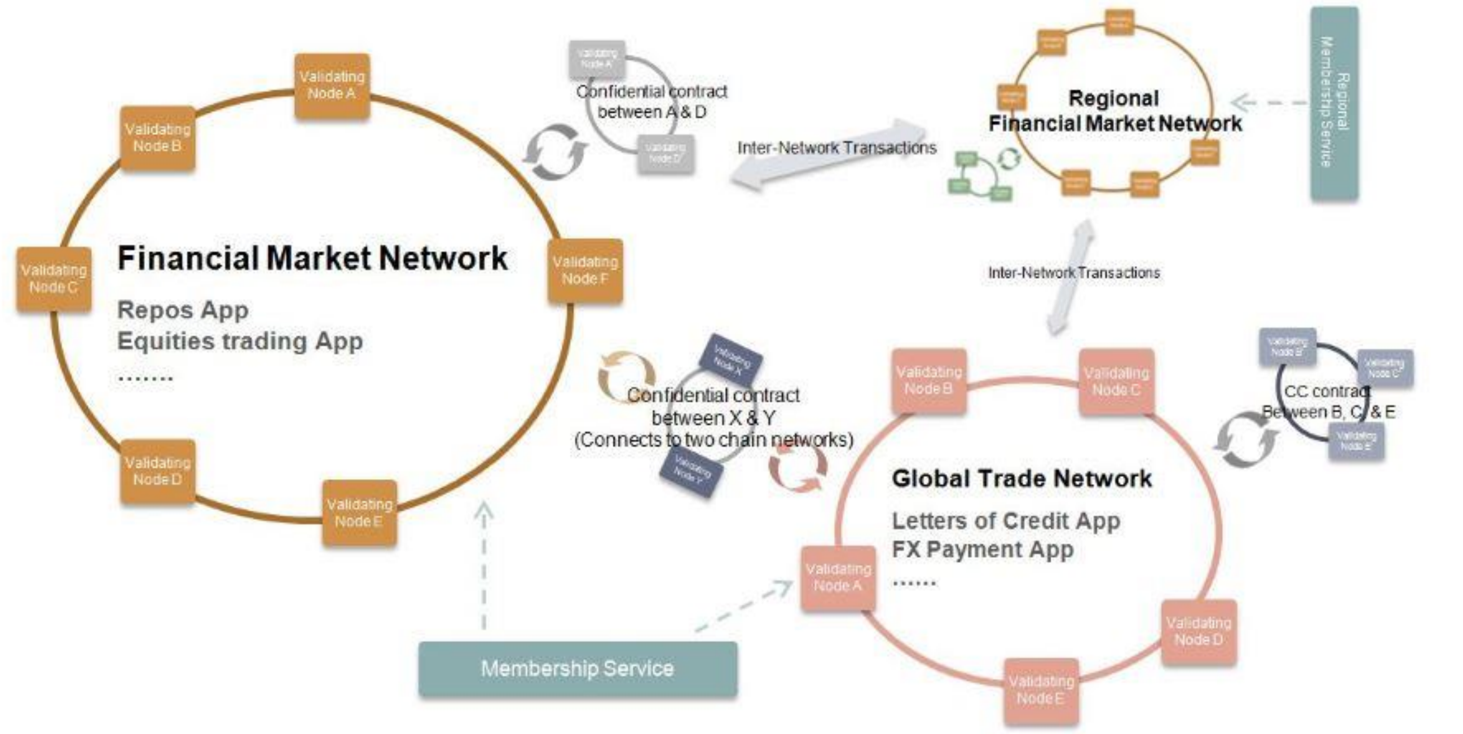
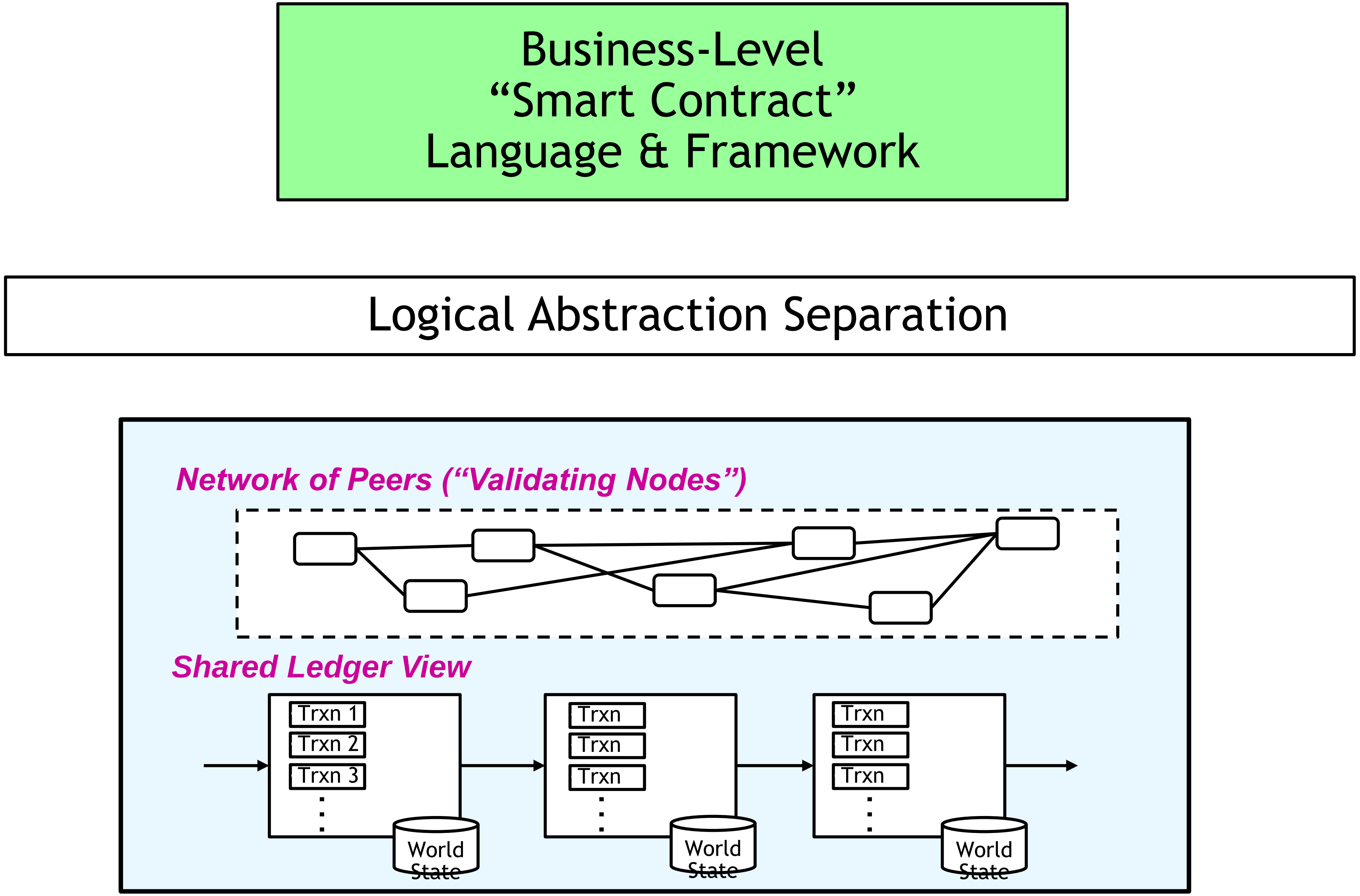


Figure 1: A world of many blockchain networks

http://www.the-blockchain.com/docs/Hyperledger%20Whitepaper.pdf

Business Collaboration Language logically above Shared Ledger



Business-Level
“Smart Contract”
Language & Framework

Logical Abstraction Separation

Network of Peers (“Validating Nodes”)

Shared Ledger View

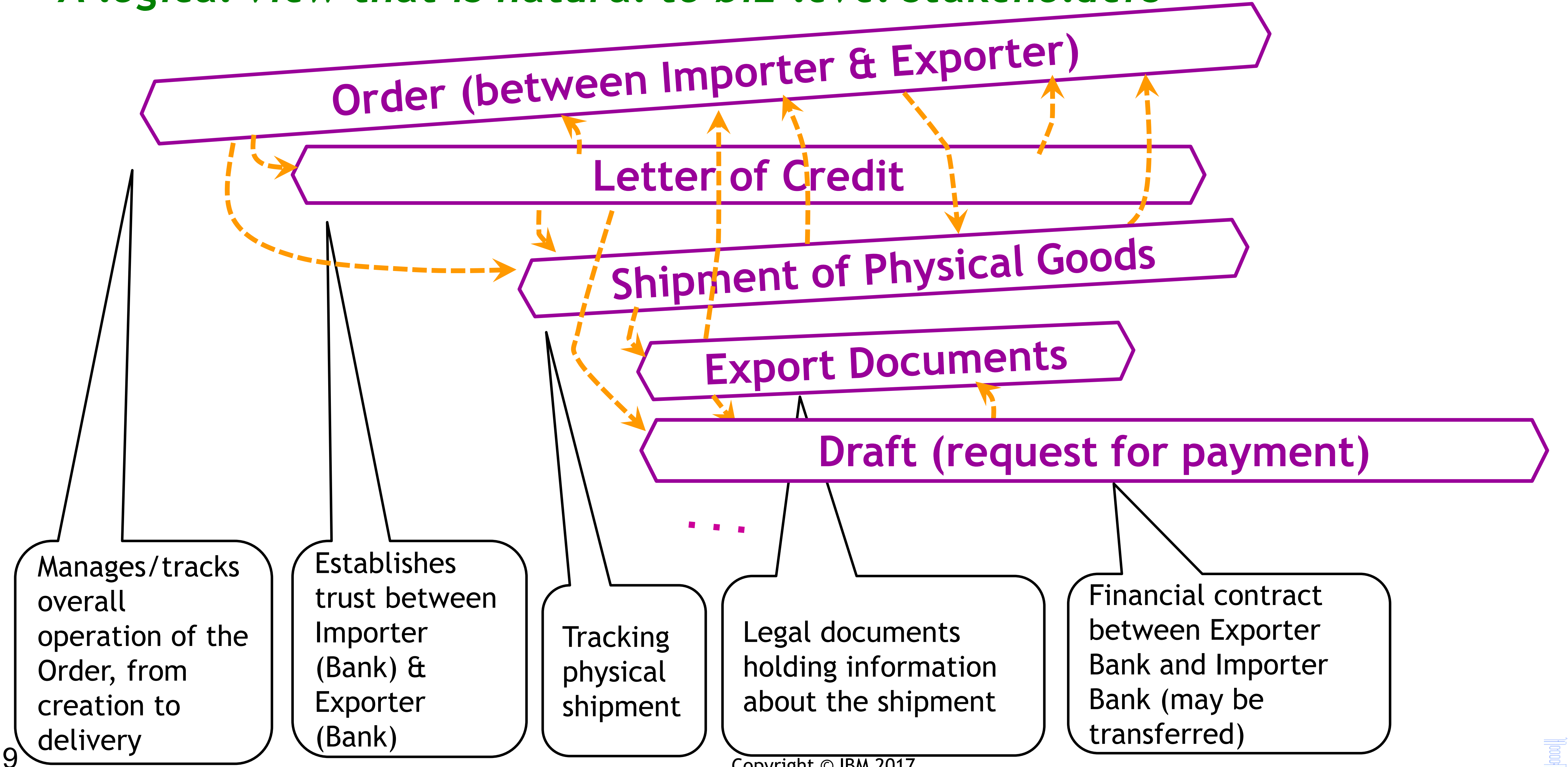
- Reminiscent of “Physical Data Independence” in databases
- Proof point:
  [Weber et. al., BPM 2016, BPM 2017]
  - Maps BPMN onto Ethereum blockchain
- Both levels are fundamentally ***event-driven transition systems***

Agenda

- Blockchain enables a new level of trust & communication
- What is Blockchain, and why is it useful for Business Collaborations?
- Logical separation between Blockchain mechanics and Biz-level programming
- Artifact-centric paradigm as starting point for Business Collaboration Language
  - Business Artifacts and related models
  - ACSI – Artifact-Centric Service Interoperation
- Selected research challenge areas
- Conclusions

Business Artifacts with Lifecycles: A way to factor Business Processes and their data that gives unifying, end-to-end view

A logical view that is natural to biz-level stakeholders



Order (between Importer & Exporter)

Letter of Credit

Shipment of Physical Goods

Export Documents

Draft (request for payment)

. . .

Manages/tracks overall operation of the Order, from creation to delivery

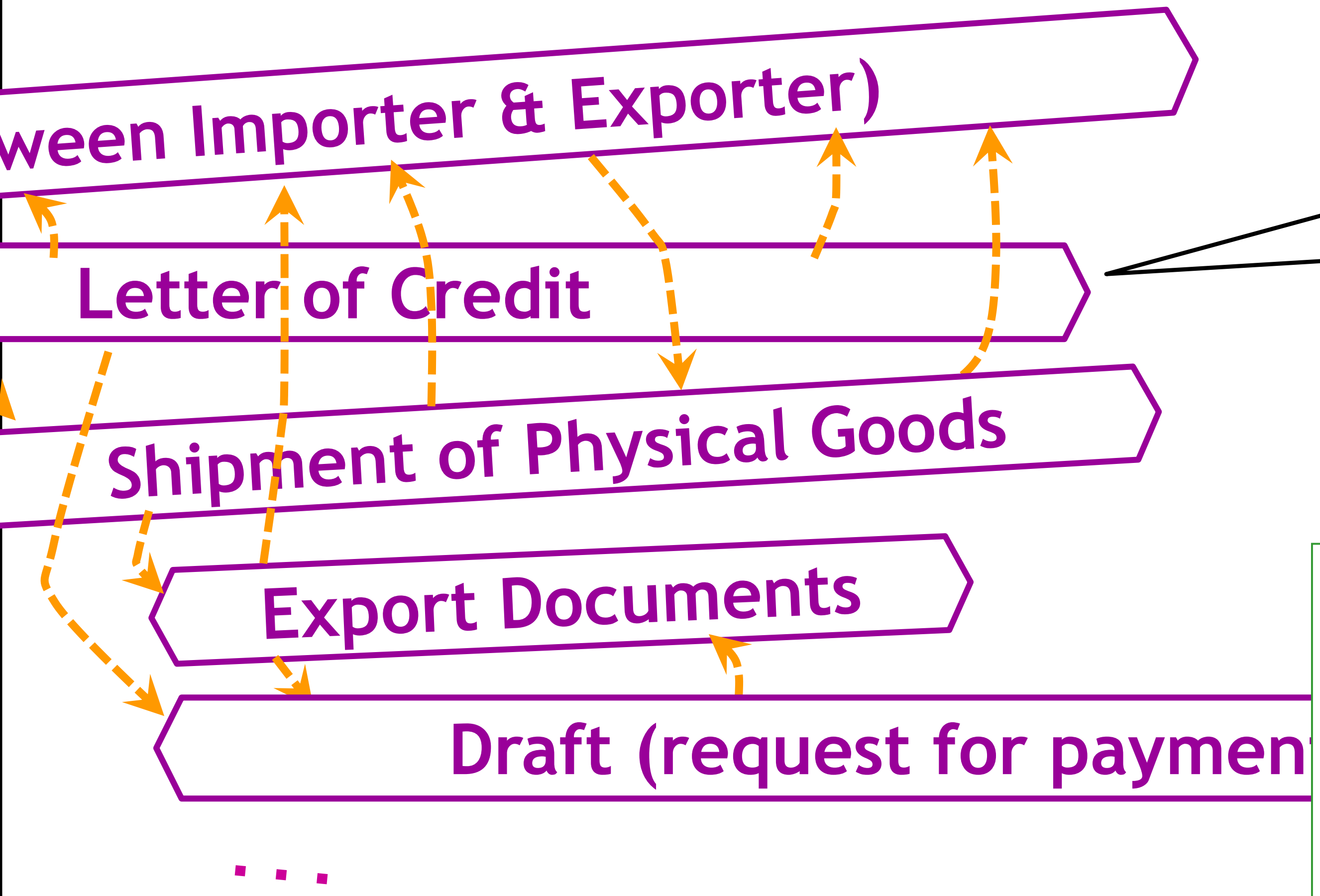
Establishes trust between Importer (Bank) & Exporter (Bank)

Tracking physical shipment

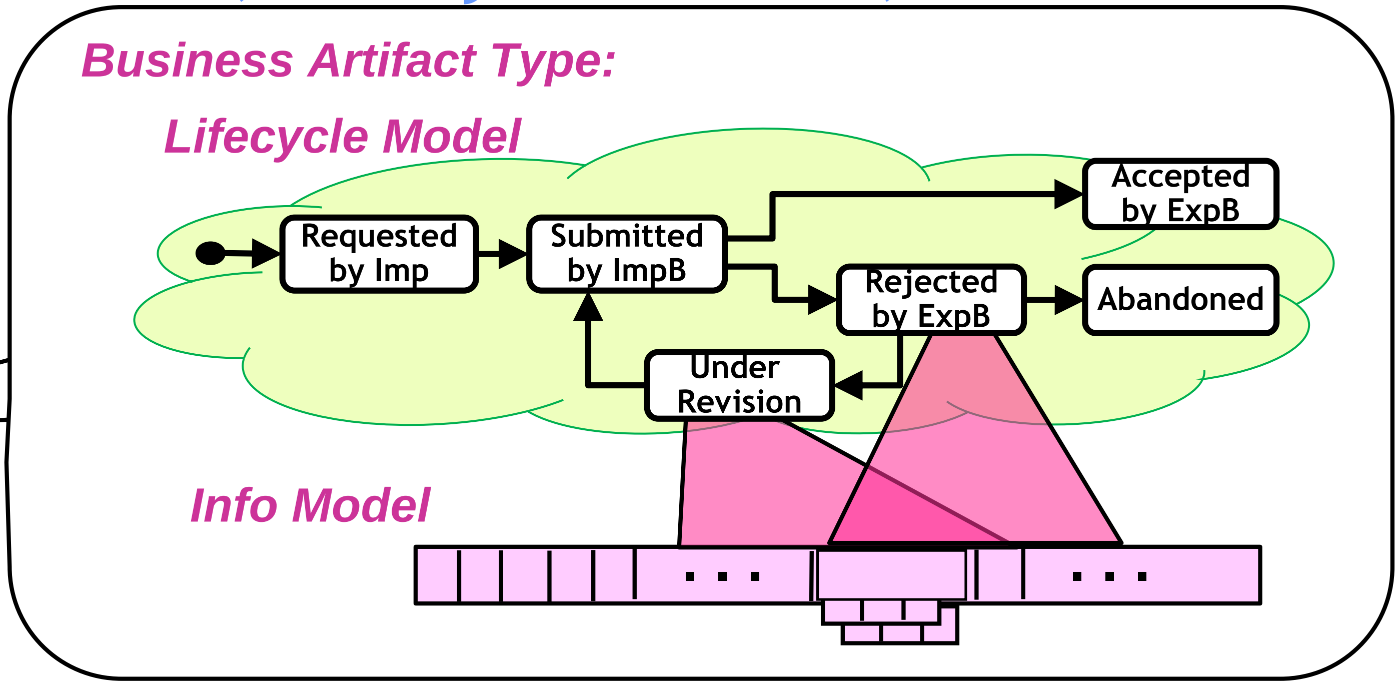
Legal documents holding information about the shipment

Financial contract between Exporter Bank and Importer Bank (may be transferred)

Each Artifact type includes info model, lifecycle model, and roles



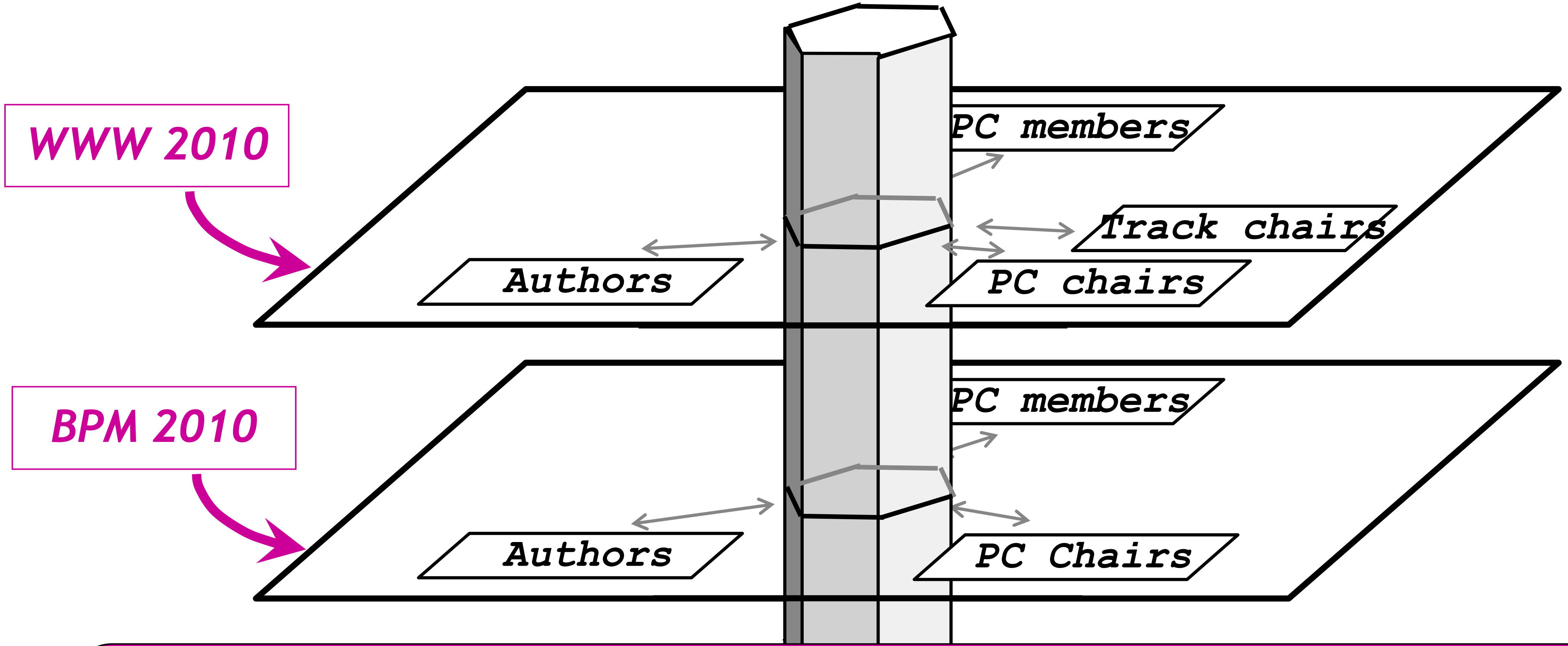
ween Importer & Exporter)
Letter of Credit
Shipment of Physical Goods
Export Documents
Draft (request for paymen
. . .



- Info model brings together all biz-relevant data about a given artifact type
  - These cut across parties, organizational silos, etc.
  - Provide a common vocabulary across parties, silos
- Lifecycle model shows possible progressions of artifact instance through the business operations
  - Status of Lifecycle is stored in the info model
- Roles have access rights to data & operations
- Biz-level stakeholders can easily query, monitor, use dashboards, and specify rules/policies

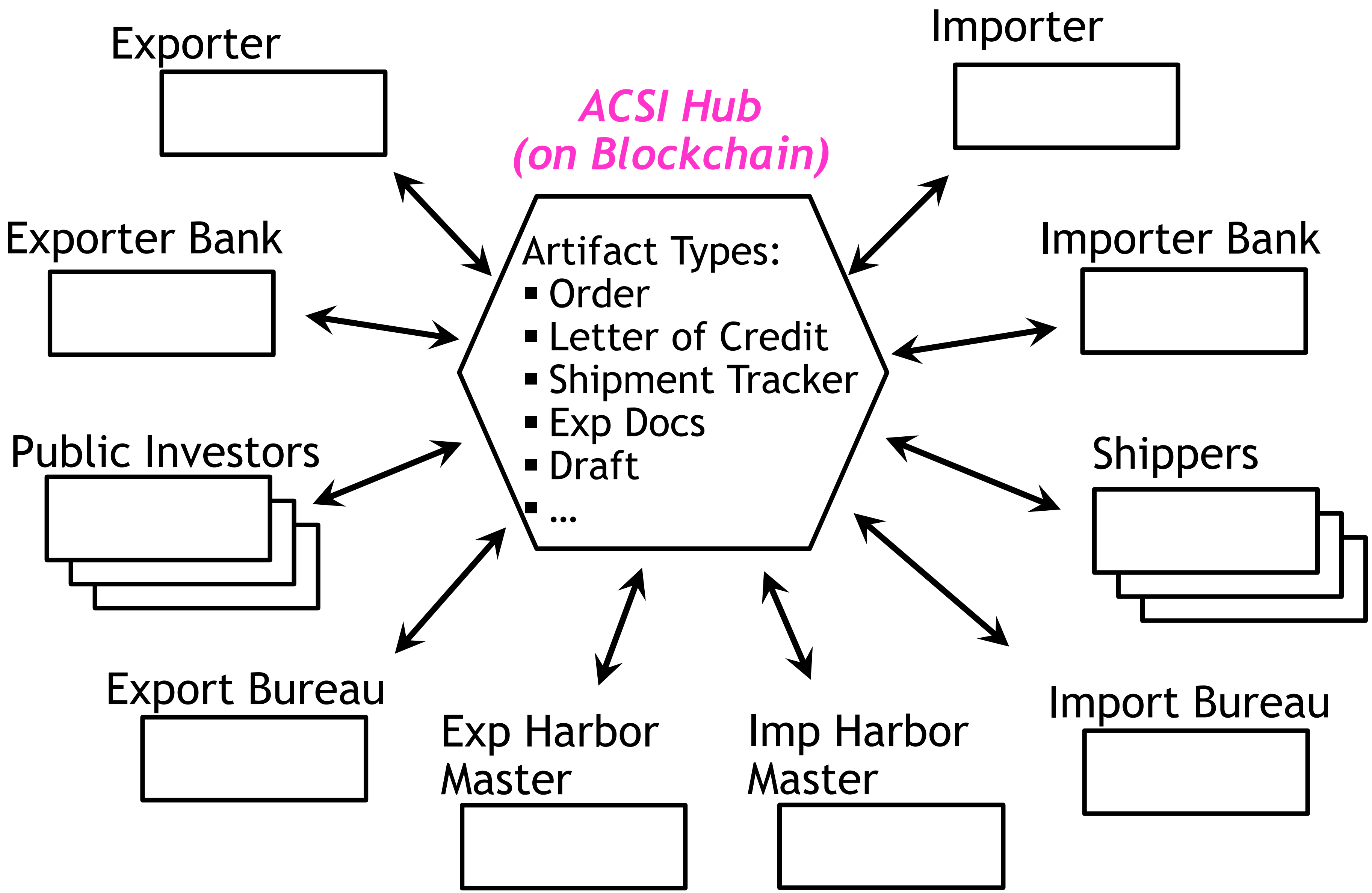
Artifact-Centric Service Interoperation (ACSI) [_, Narendra, Nigam, ICSOC 2009]

- Our Inspiration: EasyChair
- A “hub” that supports numerous conferences



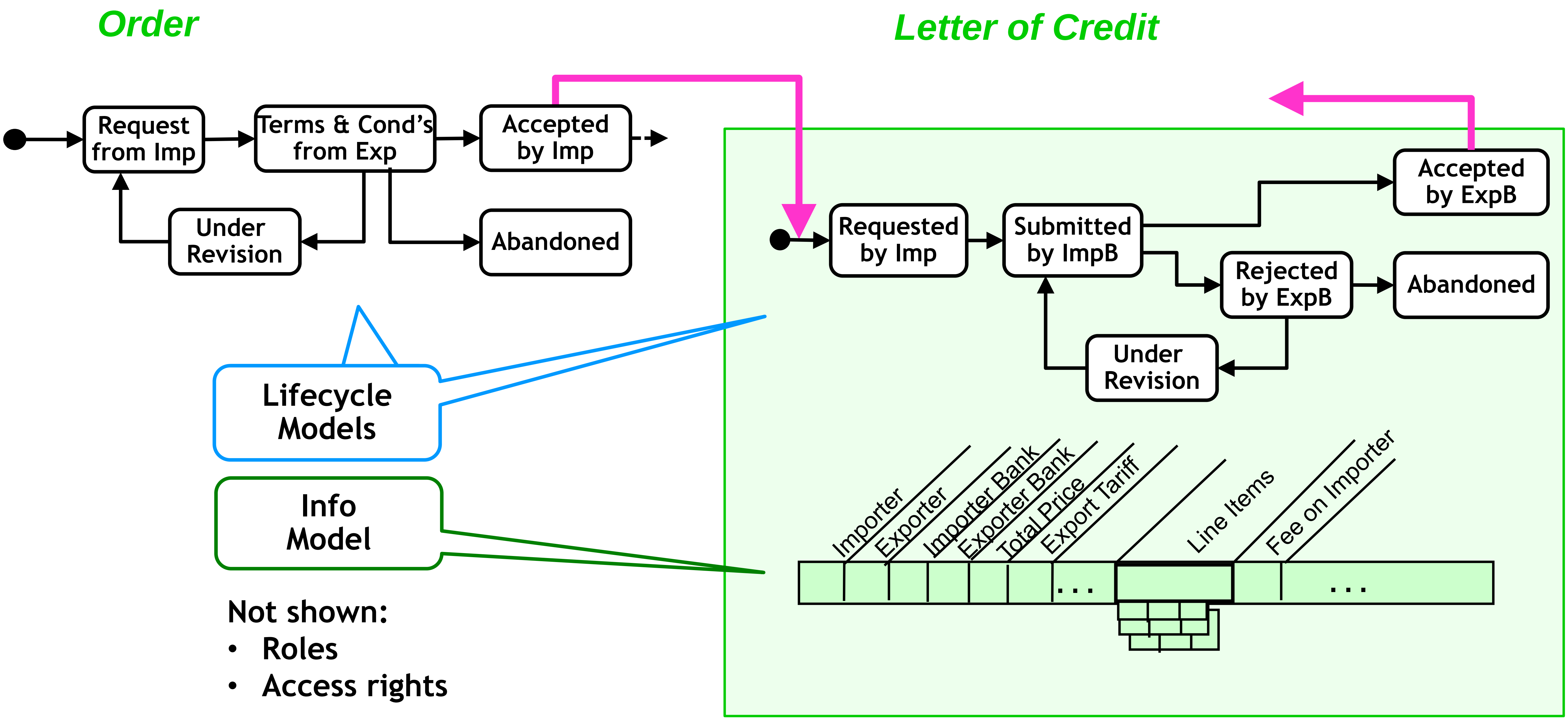
- **Events from participants lead to transitions in artifact lifecycles**
- **Hub establishes an intuitive “pseudo-standard” that participants will follow**
- **Hub is primarily re-active (unlike traditional orchestration)**
- **Hub maintains shared repository of collaboration-relevant info**

Example ACSI Hub for Trade Finance



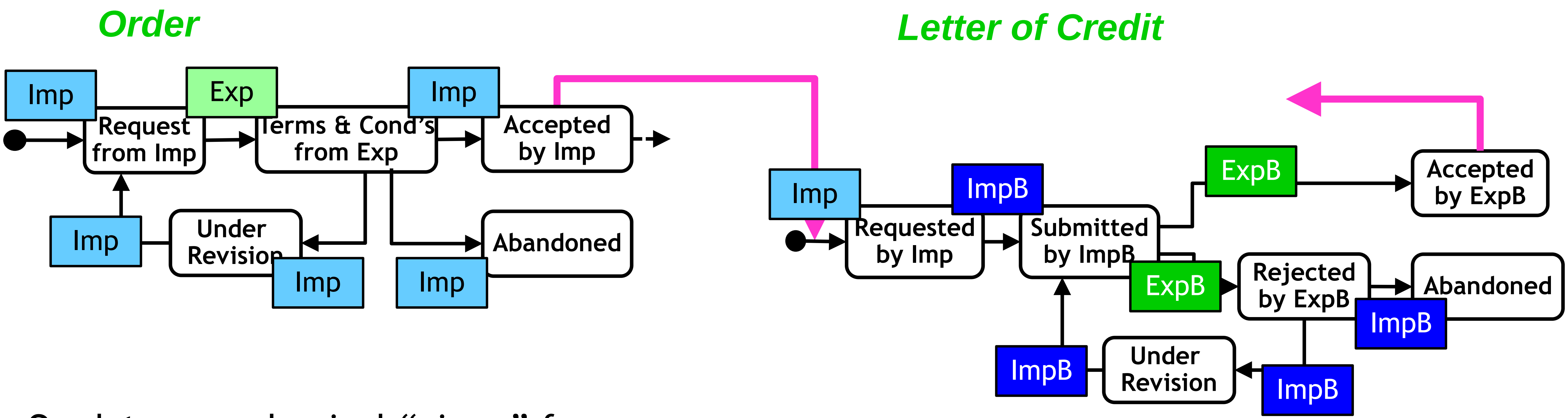
- The participating services do not have to be artifact-centric

Illustration of Lifecycle & Info Models



Not shown:
- Roles
- Access rights

Artifact structure provides natural framework for access controls



- On data: use classical "views" from databases
- On services: see illustration here
  - Can refine to create, read, update, etc.
- On instances: Restrict access to "right to know"

Imp	Importer	Exp	Exporter
ImpB	Importer Bank	ExpB	Exporter Bank

Artifacts and ACSI: Providing a robust starting point for a Business-Level Collaboration Framework for Blockchain

- There is also substantial research on Business Artifacts and the ACSI paradigm
  - Cf. EU-supported ACSI project (2010 to 2013)
- Systems – Biz Artifact (open source)
- Foundations

But … We cannot apply them "out of the box"

- Conceptual models: Blockchain restrictions – e.g., synchronous service calls
  - Operational perspective – specific notion of transaction
  - Contractual perspective, including legal and natural language
- Systems: Mapping onto Hyperledger, Ethereum, etc.
- Collaboration/Choreography: Very relevant
- Verification: Brings certain questions into focus

Agenda

- Blockchain enables a new level of trust & communication
- What is Blockchain, and why is it useful for Business Collaborations?
- Logical separation between Blockchain mechanics and Biz-level programming
- Artifact-centric paradigm as starting point for Business Collaboration Language
- Selected research challenge areas
  - Language design
  - Reasoning about artifacts
  - Relationship to natural language contracts
- Conclusions

Requirements on Business-level Smart Contracts Framework

Solution Language

- ***Intuitive*** for Business-level users to create and understand smart contracts
  - Example users: Business Analysts, Trade Specialists, Financial Analysts, Supply Chain Specialists, …
  - Holistic way of representing key business objects, including data, lifecycles, rules, roles
- Linkage between ***Legal Contractual perspective*** and Operational perspective
- Linkage to, and patterns from, existing standards, e.g., UBL, SWIFT, …
- Intuitive support for adding variations into existing smart contract specifications
  - Including modifications to business object data, lifecycles, rules
- ***Modularity & Composability***
  - Intuitively natural ways to do “plug and play”, and to substitute portions of a smart contract
  - Note: in the future, smart contracts will be created by different organizations and mashed together
- Access Control & Privacy features – specified at business level
  - For data
  - For invocable operations

- Ricardian contracts appear relevant
- Emerging CLACK language [Clack et al 2016] aimed at this challenge

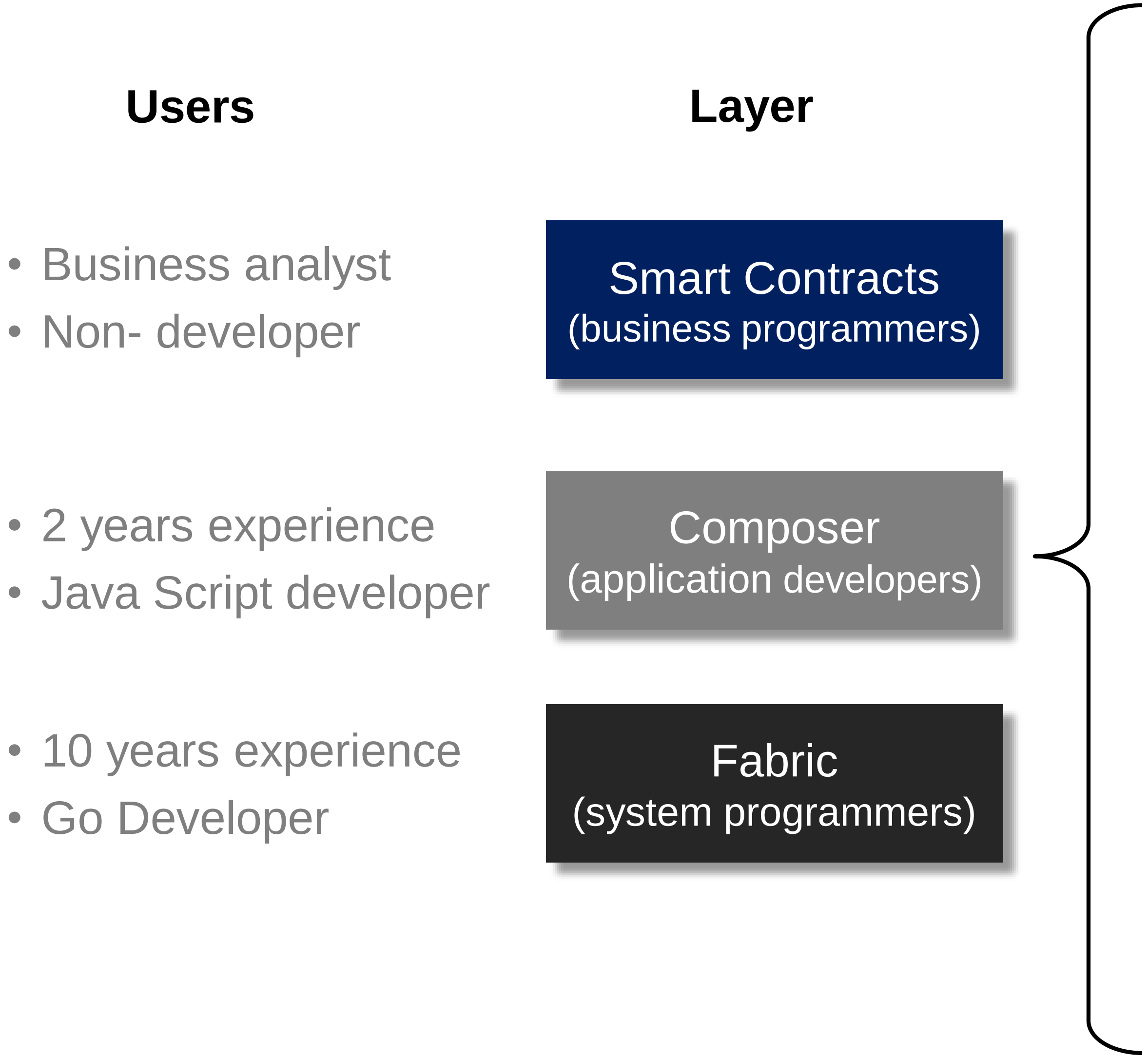
- Artifact types can serve as natural composable modules
- Data & lifecycles provide further modularity

Solution Development & Administration

- Visual editor
- Enable ***rapid development & modification of production-level solutions***
  - Use a fully interpreted paradigm for execution of smart contracts
- Design, develop, deploy, test, refine
- Version management

The BizArtifact system [Boaz et al 2013] for artifacts included
- Visual editor
- Fully interpreted implementation of artifacts
- Administration framework

The Hyperledger Composer: A first step towards artifact perspective

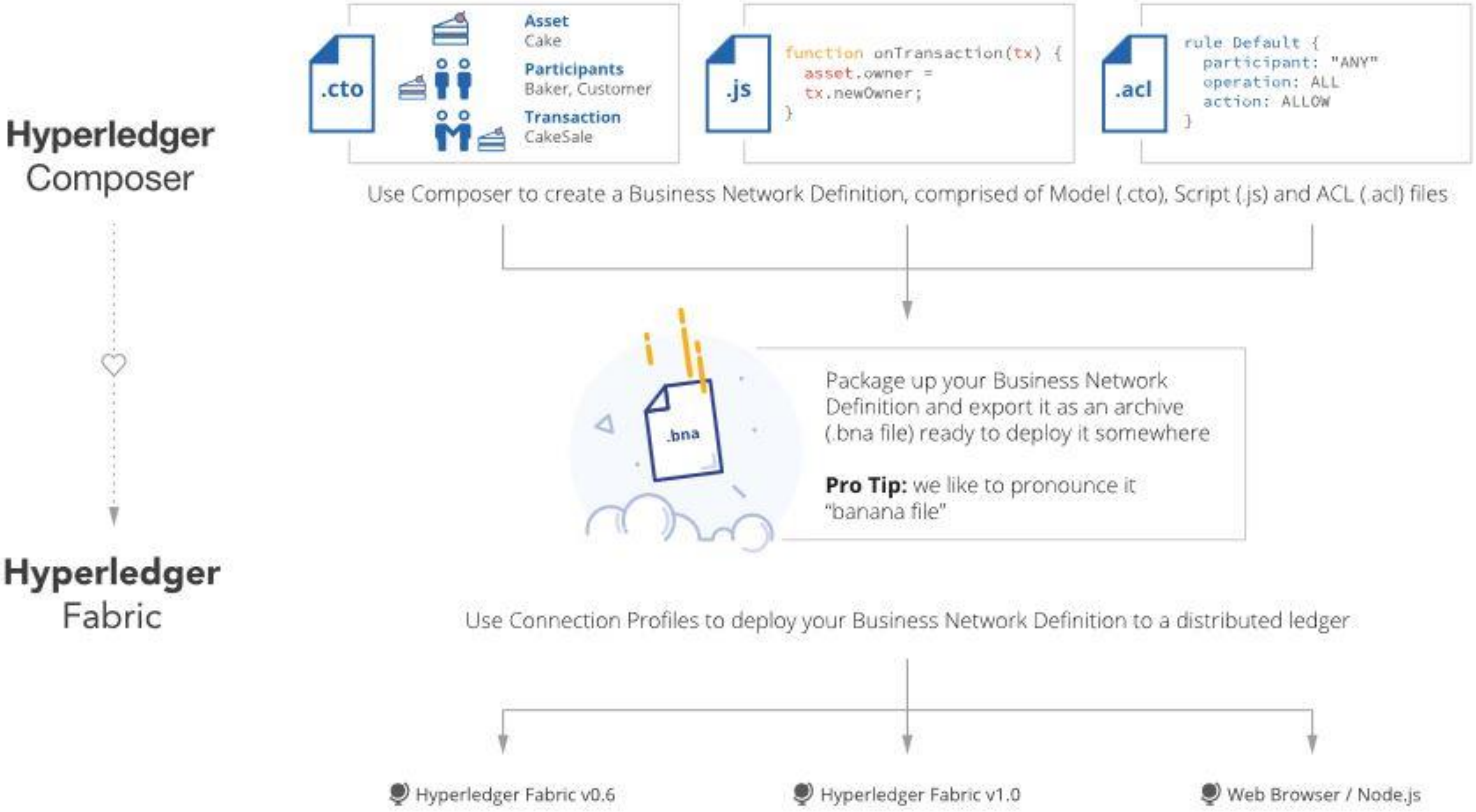


Users	Layer
• Business analyst • Non- developer	Smart Contracts (business programmers)
• 2 years experience • Java Script developer	Composer (application developers)
• 10 years experience • Go Developer	Fabric (system programmers)

Composer

- Open sourced in February 2017, as a layer above Hyperledger
- Intended to simplify development of smart contracts that support Business Collaborations
- ***Business Network Definition*** models:
  - Assets, e.g., Trade agreements, shipments, Letters of Credit
  - Participants, e.g., Importer, Exporter, Imp Bank, Exp Bank
  - (Atomic) Transaction, e.g., accept Letter of Credit, send Shipment

Creating smart contracts with Composer



```
function onTransaction(tx) {
  asset.owner =
  tx.newOwner;
}
```

```
rule Default {
  participant: "ANY"
  operation: ALL
  action: ALLOW
}
```

Use Composer to create a Business Network Definition, comprised of Model (.cto), Script (.js) and ACL (.acl) files

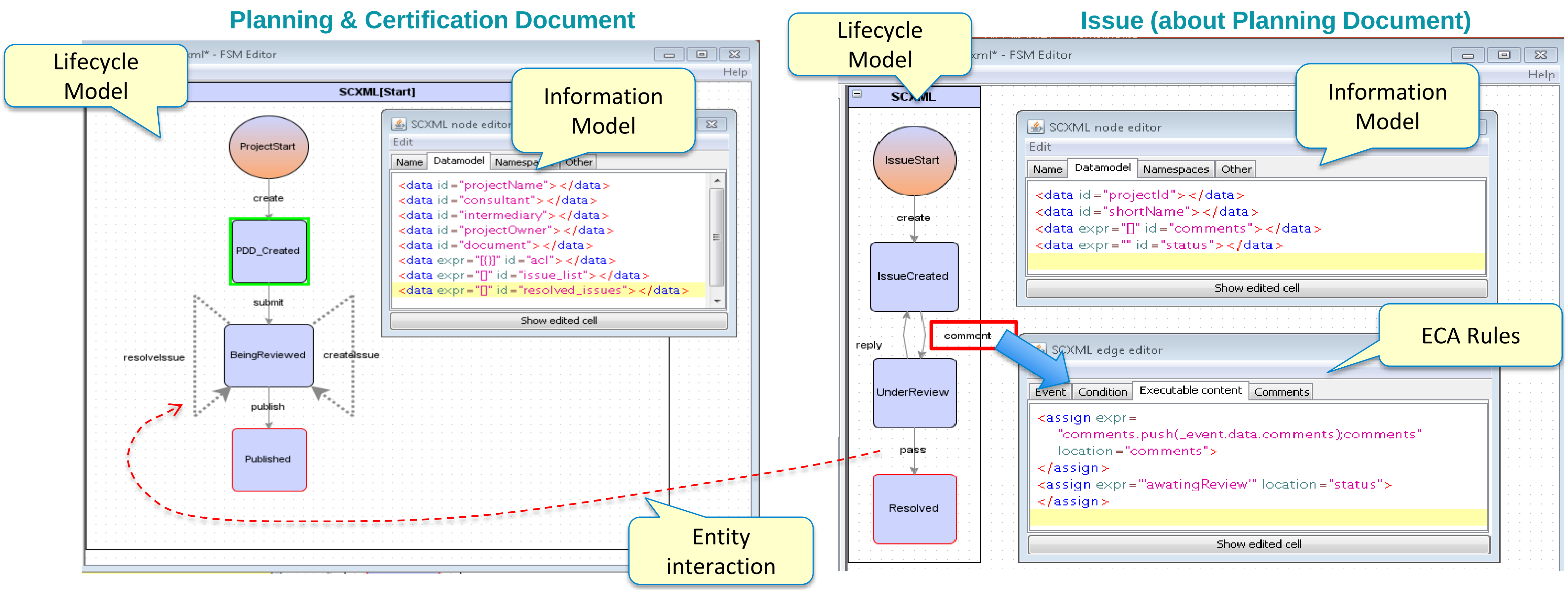
Package up your Business Network Definition and export it as an archive (.bna file) ready to deploy it somewhere

Pro Tip: we like to pronounce it "banana file"

Use Connection Profiles to deploy your Business Network Definition to a distributed ledger

Bartok: Smart Contracts based on FSM-based artifacts

- Preliminary prototype: Rests above Hyperledger and Composer
- Leverages open source SCXML tooling; extended to support inter-FSM messaging
- Access rights managed as part of the ECA rules that govern state transitions
- Exploring use of biz-level rules language for the conditions and actions, e.g., subset of SBVR



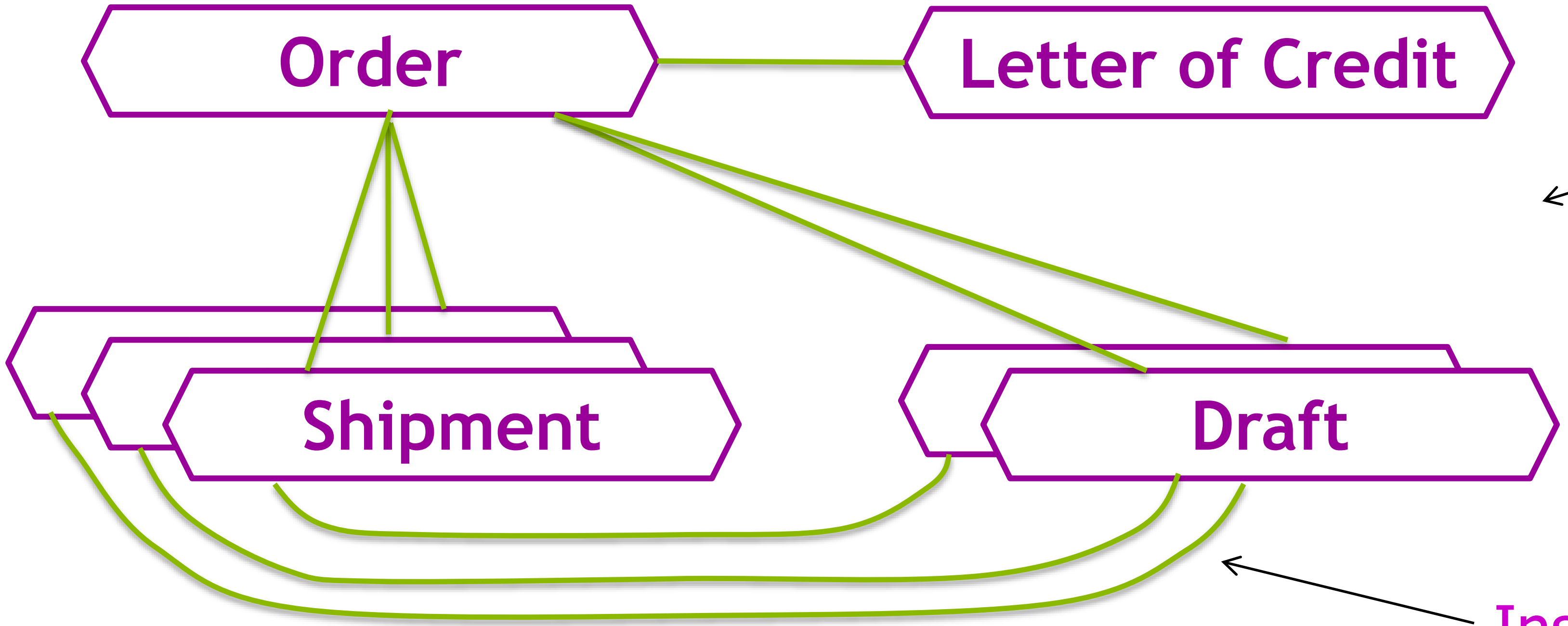
```
<data id="projectName"></data>
<data id="consultant"></data>
<data id="intermediary"></data>
<data id="projectOwner"></data>
<data id="document"></data>
<data expr="[{}]" id="acl"></data>
<data expr="[]" id="issue_list"></data>
<data expr="[]" id="resolved_issues"></data>
```

```
<data id="projectId"></data>
<data id="shortName"></data>
<data expr="[]" id="comments"></data>
<data expr="" id="status"></data>
```

```
<assign expr=
   "comments.push(_event.data.comments);comments"
   location="comments">
</assign>
<assign expr="'awatingReview'" location="status">
</assign>
```

*Bartok developed by Yao Liang Chen, Yunjie Qiu, IBM China Research Lab

Artifact types as basis for Smart Contract modularity: Behavioral constraints may work across artifacts



Conditions involving Data:

For each shipment, if Export Docs obtained by Exporter, then a Draft including that Shipment is generated by Importer Bank

If the amount is less $1000 and additional Shipments pending, then delay Draft

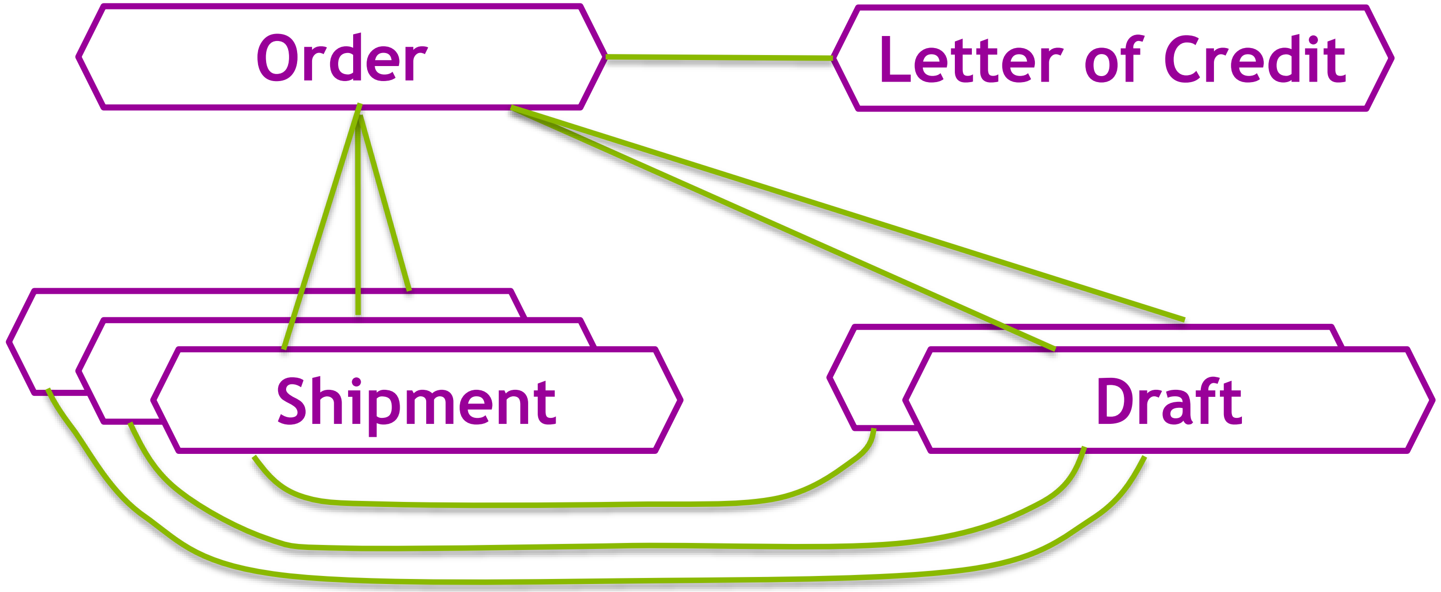
Instance-level correlation:

If a Draft is used to pay for a Shipment, both must come from same Order

- In Bartok and elsewhere, message passing is used to support interaction between artifacts
- ***Would an approach based on Complex Event Processing have advantages?***
  - This can provide a more declarative approach
  - The data from the artifact instances is available for the conditions
  - Artifact information models could hold data to help with tracking of complex conditions

Artifact types may be distributed across fabrics

A single collaboration will involve numerous artifact instances, with multiple 1-to-many relationships



Different artifact types designed & maintained by different organizations

Why?

- Making artifact types similar to existing standards, e.g., UBL, SWIFT, …
- Different kinds of concerns for logistics vs. finance

Benefits:

- This can enable “plug & play” of artifact types
- Can break contract testing & verification into manageable chunks

How can messages, events, conditions be modeled across blockchain fabrics?

Comparison with BPMN-based approaches

<< cf work of Weber, Dumas, et al, who are layering BPMN on top of Ethereum >>

- BPMN Conversation*

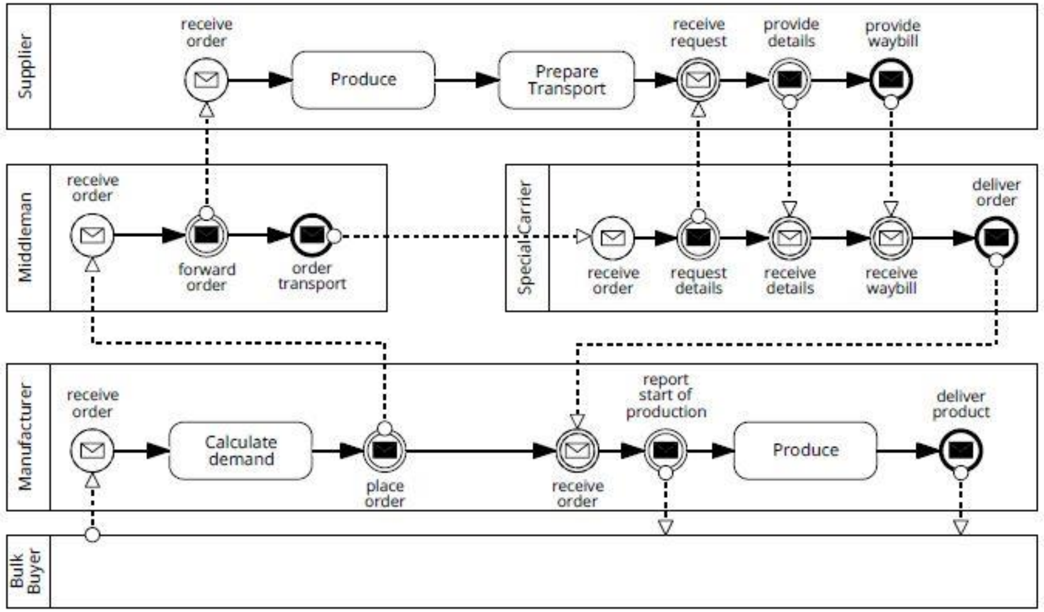


Figure 1: Supply Chain Scenario from [3] (simplified)

  - Focus on “pools”, one per participant
  - Communication via messages between participants

- BPMN Collaboration*

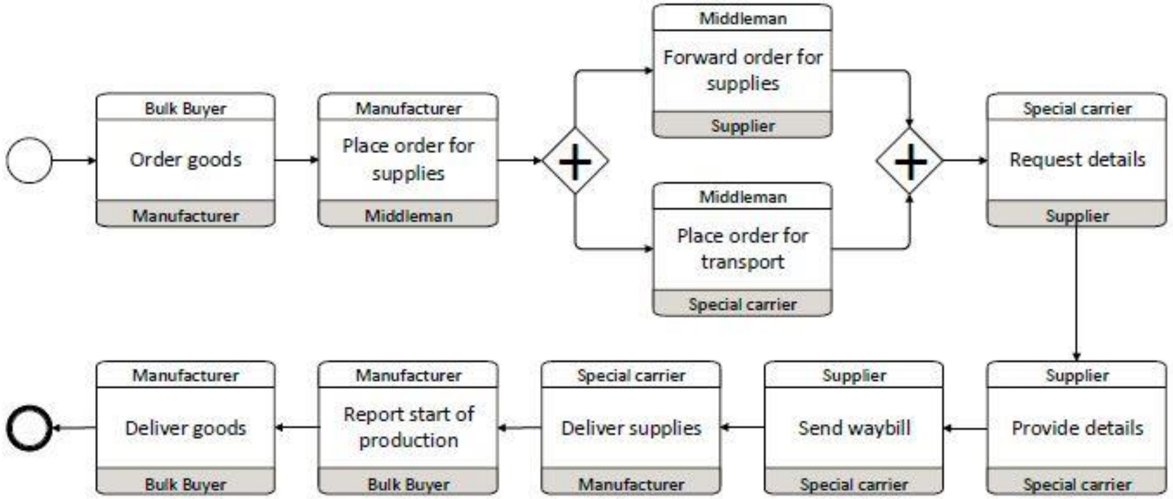


Figure 3: BPMN choreography diagram of the process in Fig. 1

  - Focus is on interactions between a set of participants
    - Each interaction has a single “initiator”
  - Sequencing by traditional flow constructs

- In BPMN approaches …
  - Process state focuses on what tasks are “in progress”, events launch new tasks
  - Data is buried in the interactions – Ability to use conditions to manage behaviors is limited
  - Modeling support for 1-many relationships limited by BPMN “multi-instance” construct
    - “Well-structured” requirement – the children have to finish before parent can

*Examples from [Weber et. al., BPM 2016]

* Examples from [Weber et. al., BPM 2016]

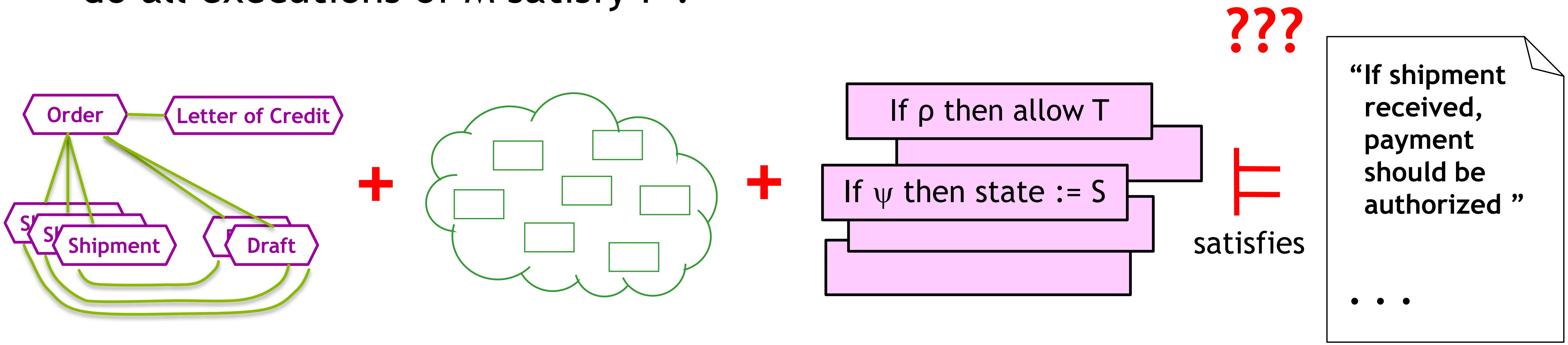
Agenda

- Blockchain enables a new level of trust & communication
- What is Blockchain, and why is it useful for Business Collaborations?
- Logical separation between Blockchain mechanics and Biz-level programming
- Artifact-centric paradigm as starting point for Business Collaboration Language
- Selected research challenge areas
  - Language design
  - Reasoning about artifacts
  - Relationship to natural language contracts
- Conclusions

Verification for artifact-centric models: a representative framework

Will be hugely important when Business Analysts are creating and modifying smart contracts

- Given an artifact-based model M and a property P, do all executions of M satisfy P ?



"If shipment received, payment should be authorized "

Artifact Info models (with messaging patterns)

Atomic Actions (e.g., specified using pre- and post-conditions)

Lifecycle (expressed using rules)

Goals / Constraints

Temporal + First-Order, e.g., LTL-FO

- The presence of data leads to an infinite state space
  - Verification in general is undecidable
  - Several different approaches to restrict expressive power have been developed
  - E.g., [Deutsch, Li, Vianu 2016] "VERIFAS: A practical verifier for artifact systems"

The Smart Contract synthesis problem

Given a family of terms & conditions for a contract

Generate

Executable smart contract, e.g., FSM-based artifacts with ECA rules

E.g.,

“ **During:** the month of July,
If: Importer sells > 100 cases per week,
Then: 5% discount on cost per case “

- Rule for processing payments should include discounts, if applicable
- May expand data recorded in artifacts about quantity sold per week, to simplify condition checking

- Theoretical approach: Explore space of Smart Contracts, and use verifier to pick one that satisfies constraints
- Pragmatically speaking: Need heuristics to dramatically narrow the search space

Agenda

- Blockchain enables a new level of trust & communication

- What is Blockchain, and why is it useful for Business Collaborations?

- Logical separation between Blockchain mechanics and Biz-level programming

- Artifact-centric paradigm as starting point for Business Collaboration Language

- Selected research challenge areas
  - Language design
  - Reasoning about artifacts
  - Relationship to natural language contracts

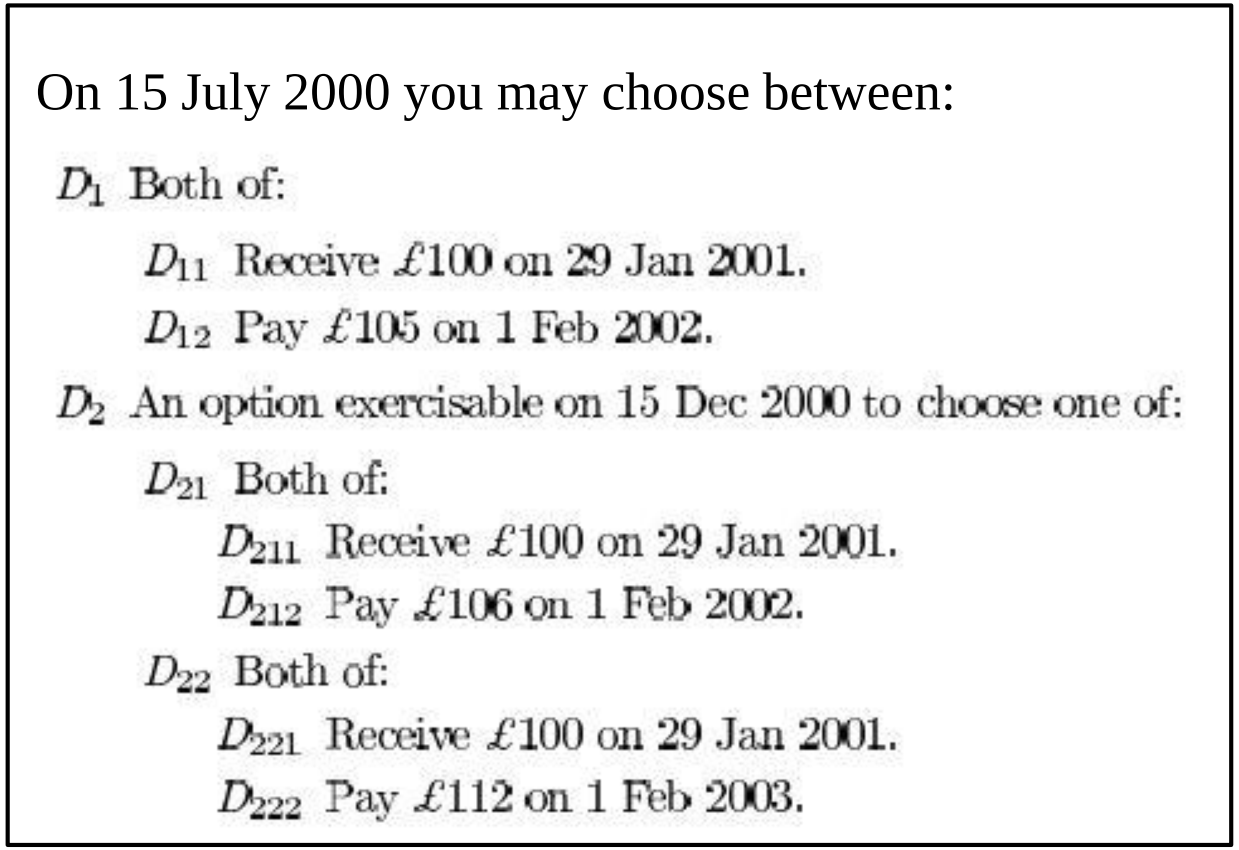
- Conclusions

Legal contracts: what makes them different?

- Binary relationships
  - “Holder”
  - “Counterparty”
- Contract based along time dimension
  - As they move through time …
  - … people make choices
  - … result is essentially a new contract
- Contracts are exchanged, combined, traded, …
- Contract may depend on external “random” variables
  - E.g., exchange rates, stock prices
- A focus of financial industry is

  What is the current value of this contract ?

  - Must incorporate uncertainties of future
  - Various statistical techniques available



On 15 July 2000 you may choose between:

D_1 Both of:

D_{11} Receive £100 on 29 Jan 2001.

D_{12} Pay £105 on 1 Feb 2002.

D_2 An option exercisable on 15 Dec 2000 to choose one of:

D_{21} Both of:

D_{211} Receive £100 on 29 Jan 2001.

D_{212} Pay £106 on 1 Feb 2002.

D_{22} Both of:

D_{221} Receive £100 on 29 Jan 2001.

D_{222} Pay £112 on 1 Feb 2003.

From “How to write a financial contract”,
S.L. Peyton Jones and J-M. Eber,
Proc. Intl. Conf. on Functional Programming, 2000

Functional programming can provide formal abstraction for finance-based contracts

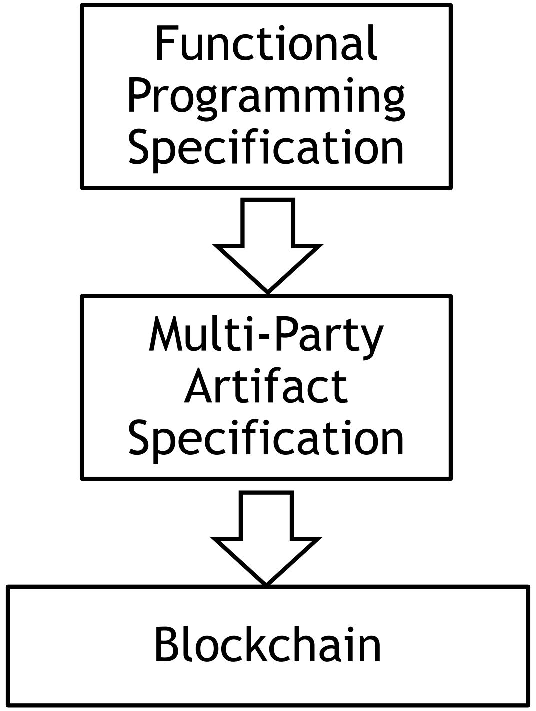
[Peyton Jones, Eber 2000] provides a family of 10 primitive combinators that can be used to formally define contracts

- "and": if you acquire "c1 and c2", then you immediately have both
- "or": if you acquire "c1 or c2", then you must immediately choose to retain one or the other
- "when": if you acquire "when <obs> c", where <obs> is a Boolean-valued observable, then c becomes available to you if/when <obs> becomes true
- "until": "until <obs> c" acts like c until <obs> becomes true. From that moment the contract becomes worthless
- …

This functional programming view enables

- Composability
- Formal reasoning about semantic equivalence

Conjecture: a family of inter-related binary contracts be can operationalized using an artifact-based Blockchain implementation


Functional Programming Specification
Multi-Party Artifact Specification
Blockchain

Reasoning about value

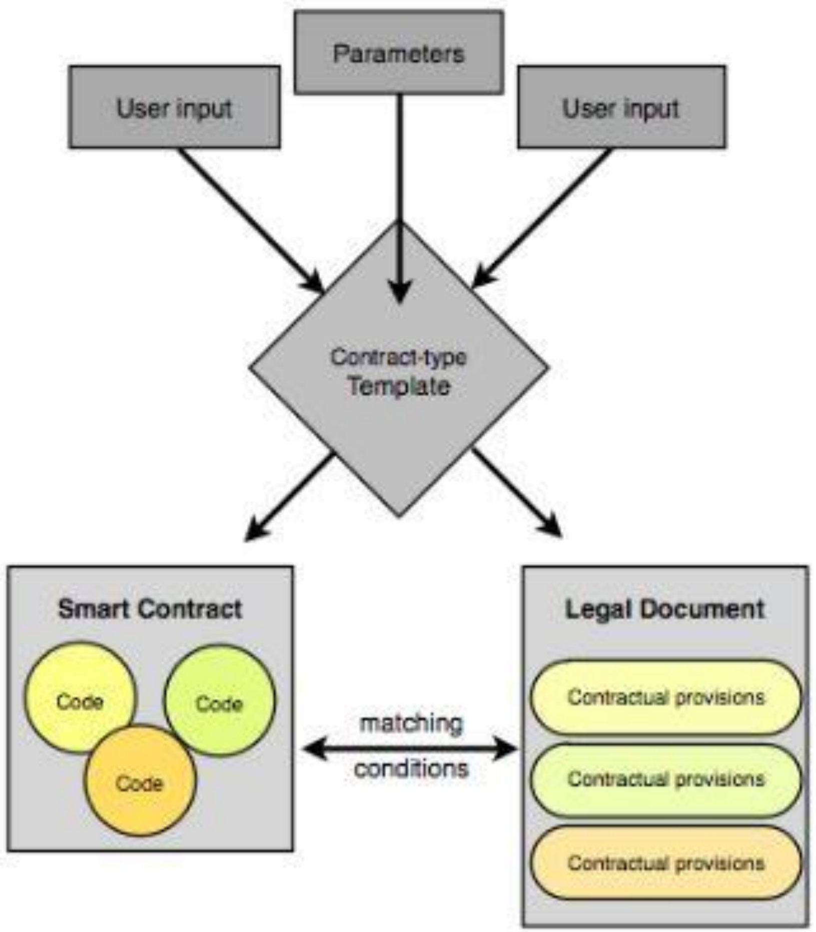
Operational semantics, Reasoning about operations

Reliable execution

Another perspective on mixing “legal” and “smart” contracts

- Distinction made in CoinDesk by Stark [2016]
  - *Smart Contract Code:* code that embodies how agents want to collaborate, running on a Blockchain
  - *Smart Legal Contract:* combination of legal wording and executable code that correspond to each other

- *Ricardian contracts*: an example of Smart Legal Contract
  - Invented by Ian Grigg [2004]
  - “A digital contract that defines the terms and conditions of an interaction between two or more peers, that is cryptographically signed and verified”
  - It is both **human and machine readable**
  - Has a unique and secure identifier



http://www.webfunds.org/guide/ricardian_implementations.html

Groups like CommonAccord are attempting to create a body of “universal contracts” that can handle essentially all useful kinds of collaborations

Agenda

- Blockchain enables a new level of trust & communication
- What is Blockchain, and why is it useful for Business Collaborations?
- Logical separation between Blockchain mechanics and Biz-level programming
- Artifact-centric paradigm as starting point for Business Collaboration Language
- Selected research challenge areas
- Conclusions

Blockchain: A new technology with growing adoption for Business Collaboration

This raises many of the classical questions from Services & DEBS communities . . .

. . . But with a twist:

A new way for managing distribution and data consistency at the core

- Allows, and forces, a re-thinking of basic Services & DEBs approaches, such as
  - Orchestration/choreography: is ACSI hub the right abstraction, or something else?
  - Service composition: It's not just about message/conversation compatibility anymore
    - Using Business Artifacts be the unit of composition provides unified basis for data and messaging
- This talk emphasized the abstraction layer above the distribution, encryption, consensus
  - Can a DEBS perspective teach us something about that boundary, e.g., for optimizations?

Blockchain: Operational vis-a-vis Legal/Financial perspectives

Two critical observations:

- The courts will always be the remedy of last resort → legal perspective is always present
- Almost every operational task has a financial aspect → financial perspective is always present

- Brings a new style of challenge to the Services & DEBS communities
  - Service composition: Legal and Financial contracts are interlocking, interdependent
    - Do our current paradigms adequately model this?
  - Event Management: How to map between (complex) event perspective & legal perspective
  - Formal reasoning/verification: We need to address Legal/Financial patterns (among others)
  - Design/Coding style: How will marriage of legal+code be structured, at macro- and micro- levels