A new characterization of atomicity of business interactions (or How to Construct long running transactions without coordinators)

Santosh Shrivastava
School of Computing Science
Newcastle University, UK

(joint work with Carlos Molina-jimenez, Mark Little)

1

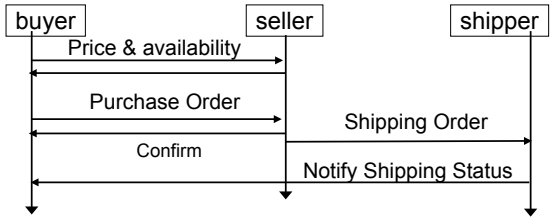
Background

- Domain of interest: business-to-business (B2B) message-based interactions
  - two or more interacting parties collaboratively executing a loosely coupled cross-organizational activity (a *business function*),
    - travel booking, order fulfilment, …
    - can be viewed as the business partners taking part in the execution of a *shared business process* (also called *public* or *cross organizational business process*)
  - These are peer-to-peer interactions
    - any peer can initiate the transfer of a message
    - each peer can locally and unilaterally decide (at any time) on the correctness of a received message and on the final outcome of the interaction

2

Background

- Example Business function: Order fulfilment



Underlying technologies/concepts:

Web services, Service oriented architectures, business process management, cross-organisation workflows, service orchestration, service choreography, ACID/non-ACID transactions, electronic contracts, service agreements,……

3

Consistency

- Want to nail the concept of 'consistency' for such message-based interactions in precise terms

– in a manner that enables:
  - their automated correctness analysis, and
  - development of appropriate mechanisms for consistency maintenance.

- Large body of literature on transactions: ACID, non-ACID

– Solutions based on distributed ACID as well as their non-ACID counterparts for long running activities (WS-coordination, WS-BusinessActivity and the like) do exist
  - but hardly ever used in this domain.
  - Why?

4

Consistency

- Reluctance of industry to accept heavy-weight protocols that require cross-organization coordination
  - which could imply loss of autonomy for substantial durations.
  - not even ‘credit-debit’ transactions across banks are run as distributed ACID transactions
    - Rather they are run as local ACID transactions with messaging and compensation

5

Consistency

- ACID consistency has been about databases (stored data)
- Non-ACID models do minor tinkering…..
- Given loose coupling and the desire of organizations to maintain autonomy, making any consistency assertions about the data stored in organizational databases does not seem very practical, or make any sense
- Time to say goodbye to database centric consistency approaches for B2B message-based interactions

6

New Approach

- Focus on the business objectives of what the interacting parties *ought to be doing* in a given interaction.
  - Objectives encoded explicitly or implicitly in a contract
- Our approach is *choreography centric*
  - Externally observable interaction between business partners, encoded as a choreography specification
  - Use some magic to determine whether this specification meets the business objectives under normal/exceptional/failure scenarios
    - If so, the choreography is atomic
    - Use it to implement individual business processes

7

Contracts and choreographies

- Partner interactions are underpinned by a *business contract* (a service agreement)
  - A contract defines (either explicitly or implicitly), what operations the parties have the Rights (R), Obligations (O) or Prohibitions (P) to execute as well as when the operations are to be executed (time constraints) and in which order.
    - in a buyer–seller interaction the service agreement might stipulate that the buyer is obliged to submit payment within five days of receiving acceptance of a submitted purchase order.
- A *choreography* specification describes, from a global perspective, all permissible message exchanges between the partners.
  - Useful for implementing individual business processes

8

Contracts and choreographies

- Contract and choreography specifications describe permissible interactions between partners from different view points, emphasising different aspects.
  - Contract specification: notations drawn from the legal business domain, concerned with rights (permissions), obligations etc.
    - "*The buyer's right to submit purchase orders is suspended until he fulfils all his pending obligations"*
    - When an interaction completes, "has the partner met all the obligations?"
  - Choreography specification: focus is on specifying the business interactions at message level
    - reasoning about safety and liveness properties;
    - *E.g., safety property: "never deliver the goods before payment"*
    - is the Choreography realisable?

9

Atomic Choreography

- We want to make sure that message exchanges as encoded in a given choreography conform to (are in accordance with) the contract.
  - make sure that any message interaction permitted in the choreography are 'contract compliant': will not cause a breach of the contract.
  - We say that a given choreography execution sequence is *consistent* if in the terminated state there are *no pending obligations*
    - all the partners have performed their duties! Its contract compliant.
  - A chorography is *atomic* if all of its execution sequences are consistent.

10

Example

- Textbook example: transferring money from account A from one bank to account B at some other bank.
  - Let ‘d(A)’, ‘c(B)’ stand for messages for requesting debit from A and credit from B, and so forth;
  - subscript ‘s’ (eg, $d(A)_s$) indicates the message has been accepted by the receiver for processing, subscript ‘f’ (eg, $c(B)_f$) indicates that the message was not delivered for processing at the receiver
  - Finer failure classification considered later

11

Example

- We can see intuitively that execution sequences such as $\{d(A)_s, c(B)_s\}$ (indicates successful transfer), $\{d(A)_f\}$ (no effect), $\{d(A)_s, c(B)_f, c(B)_s\}$ (successful transfer after retrying credit), $\{d(A)_s, c(B)_f, c(B)_f, c(A)_s\}$ (credit at B does not succeed even after a retry, so money is credited back to A) are ‘consistent’,
- whereas sequences such as $\{d(A)_s\}$ or $\{d(A)_s, c(B)_f\}$ are not consistent.

12

Building Blocks for Atomic Chorography

- Assume a business function (travel booking, order fulfilment etc) is composed of well defined primitive operations
- Such an operation represents a primitive interaction between two partners, involving exchange of one business message
  - *"buy request"*, "*invoice notification"*, "*verify that a customer credit card is valid and can be used as a form of payment for the amount requested"*, etc.
  - Practical examples: RosettaNet partner Interface Processes (PIPs), OpenTravel Alliance business messages for travel industry

13

Building Blocks for Atomic Chorography

- Outcome event of a business operation:
  - *Success (S)*: message accepted by the receiver for processing
  - *Technical failure (TF)*: message not accepted by the receiver for processing (TF models protocol related failures, e.g., syntactically incorrect message)
  - *Business failure (BF)*: message not accepted by the receiver for processing (BF models business level failures, e.g., semantically incorrect message)
  - Both parties know of the outcome: a synch protocol is required (very similar to the TCP connection management)

14

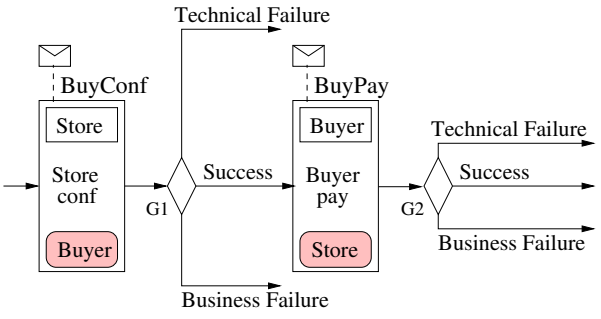
Building Blocks for Atomic Chorography

- Note the messaging abstraction with bi-lateral consistency guarantee:
  - The sender needs a timely assurance that the sent document will be processed by the receiver, and the receiver needs the assurance that if it accepts the document for processing, the sender will be informed of the acceptance in a timely manner; in all other cases, the interaction returns failure exceptions to both the parties.
  - Ensures that business processes do not get misaligned

15

Building Blocks for Atomic Chorography

- With failures, choreographies get very complex
  - long and subtle execution sequences.
  - extremely hard to generate and reason about them without tools.



16

Design Approach

- Contracts and choreographies are modelled by finite state machines (finite automata) that accept languages over the same alphabet.
- We show how by carefully defining the alphabet and specification approaches, we can reason about contracts and choreographies
  - Model checking techniques can be used for their verification
  - their input-output behaviours can be compared for conformance
  - we are able to cope with failures and exceptions that any practical specification technique —for contract or for choreography— must take into account

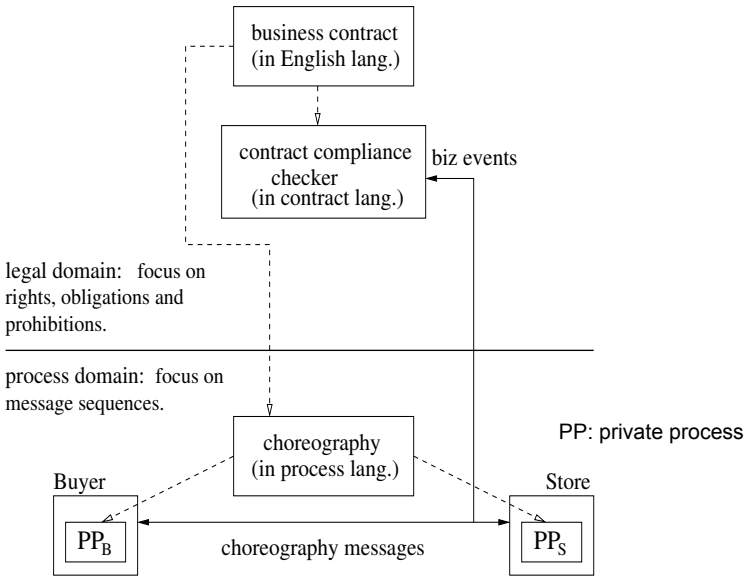
17

Approach

- Alphabet consists of *business events* derived from business messages
- Contract state machine specified using ‘event-condition-action’ rules concerning rights/ obligations of partners that check whether business events are ‘contract compliant’
- Choreography state machine specified using message passing protocol notation (a restricted version of BPMN)

18

Contracts and Choreographies



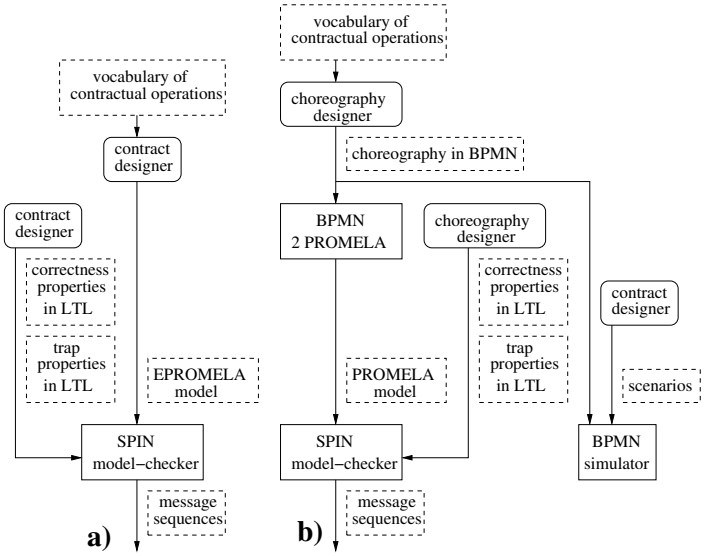
19

Our tools

- We have built BPMN2PROMELA
  - verifies logical correctness of **choreographies**.
  - mechanically generates all the execution sequences encoded in a BPMN choreography.
- We have built EPROMELA
  - verifies logical correctness of **contracts**.
  - mechanically generates all the execution sequences encoded in a contract.

20

A framework for validating logical consistency and conformance



21

Conclusions

- Developed the concept of atomic choreography
- Requires only message level synchronisation
  - Of same complexity as used in TCP
- We described a model checker based tool framework for conformance checking with contract that forms the basis for designing atomic choreographies

22