Models for Web services transactions

Mark Little,

Arjuna Technologies Ltd

SIGMOD, June 13th-18th 2004, Paris, France

arjuna middleware for reliability

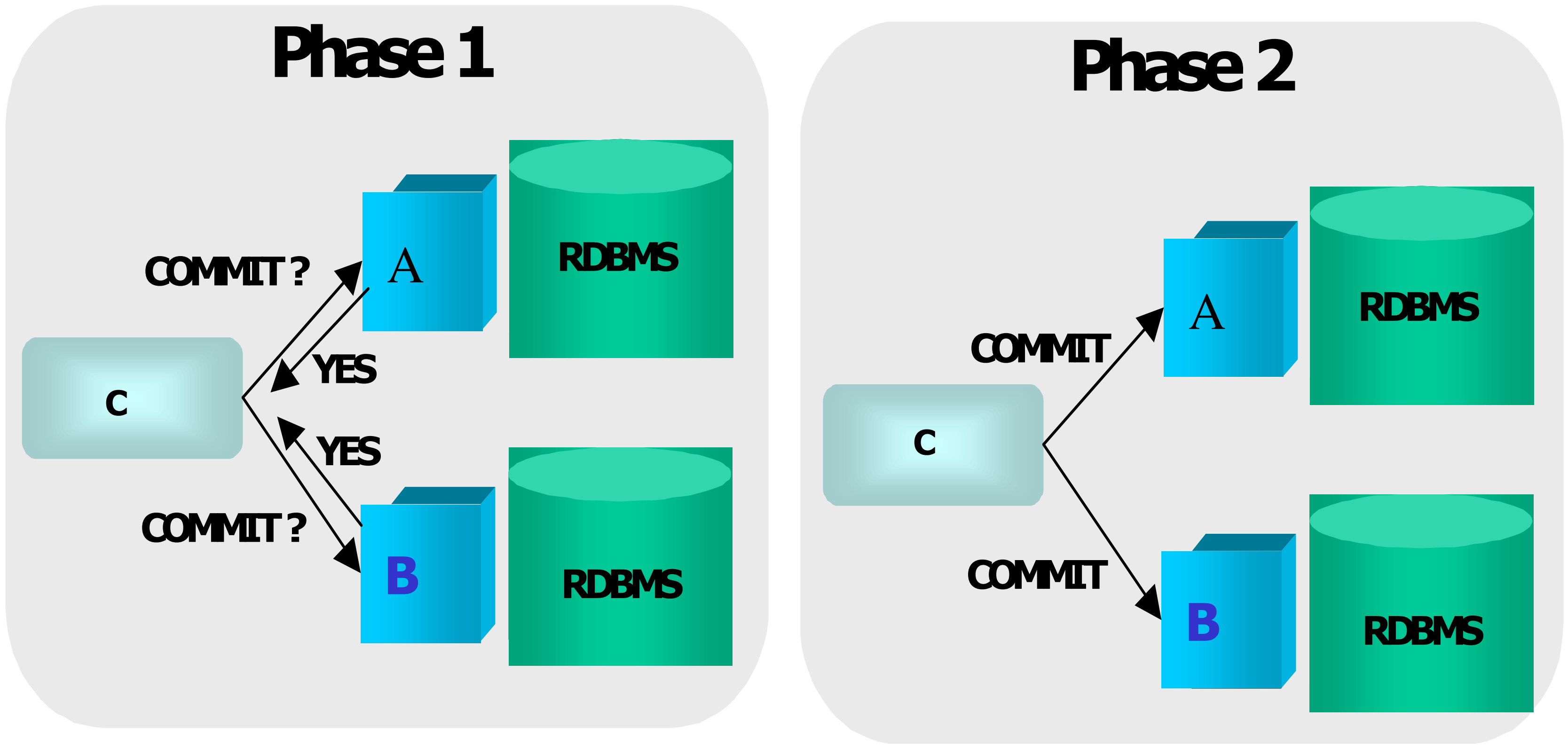
Overview

- Transactions and why they are important
- Web services and the problems they present
- Web services transactions specifications
  - – OASIS BTP
  - – WS-AtomicTransaction/WS-BusinessActivity
  - – OASIS WS-TXM

Atomic transactions

- Scoping mechanism that provides “all-or-nothing” semantics
- Enables shared resources to be protected from concurrent users
- ACID properties
  - Atomic
  - Consistent
  - Isolated
  - Durable

Two-phase commit



B2B interactions

- Business-to-business interactions may be complex
  - involving many parties
  - spanning many different organisations
  - potentially lasting for hours or days
- Cannot afford to lock resources on behalf of an individual indefinitely
- May need to undo only a subset of work

Relaxing isolation

- Internal isolation or resources should be a decision for the service provider
  - E.g., commit early and define compensation activities
  - However, it does impact applications
    - Some users may want to know a priori what isolation policies are used
- Undo can be whatever is required
  - Before and after image
  - Entirely new business processes

Relaxing atomicity

- Sometimes it may be desirable to cancel some work without affecting the remainder
  - E.g., prefer to get airline seat now even without travel insurance
- Similar to nested transactions
  - Work performed within scope of a nested transaction is provisional
  - Failure does not affect enclosing transaction
- However, nested transactions may be too restrictive
  - Relaxing isolation

OASIS BTP

- Developed by BEA, HP, Oracle, Sun and others
- First real standards attempt
  - Not Web services specific
- Defines two transaction models
  - Atoms
  - Cohesions

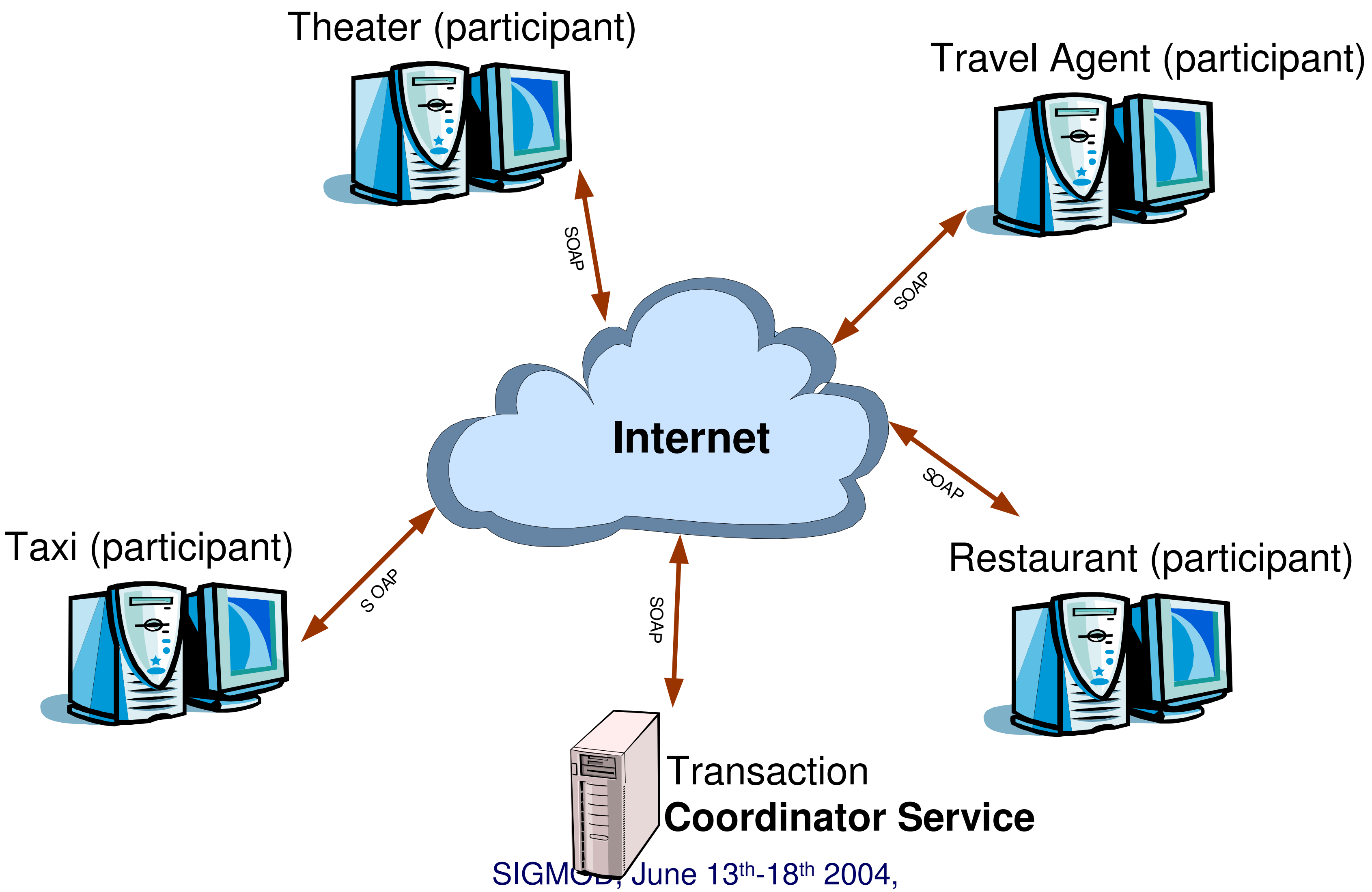
Atom

- Uses two-phase termination protocol
  - prepare, confirm and cancel
  - Termination is atomic
    - *All* participants will do the same thing
    - Does not mandate how to implement prepare, confirm and cancel
      - E.g., prepare could be “charge credit card”
- Participants can signal upstream
- Does not say anything about isolation
  - Services cannot define isolation within the protocol

Cohesion

- prepare, confirm and cancel are parameterized
  - prepare and cancel can be called multiple times
- Work on a set of Atoms
  - Allows the confirm of a specific subset of work
    - Superset of Atom functionality
- Once subset is determined by business logic, the outcome will be atomic
- Does not talk about isolation

Example interaction



WS-AT/WS-BA

- Specifications released by BEA, IBM and Microsoft
- Separate coordination from transactions
- Define two transaction models
  - AtomicTransaction
    - Closely coupled, interoperability
  - Business Activities
    - Compensation based, for long duration activities

AtomicTransaction

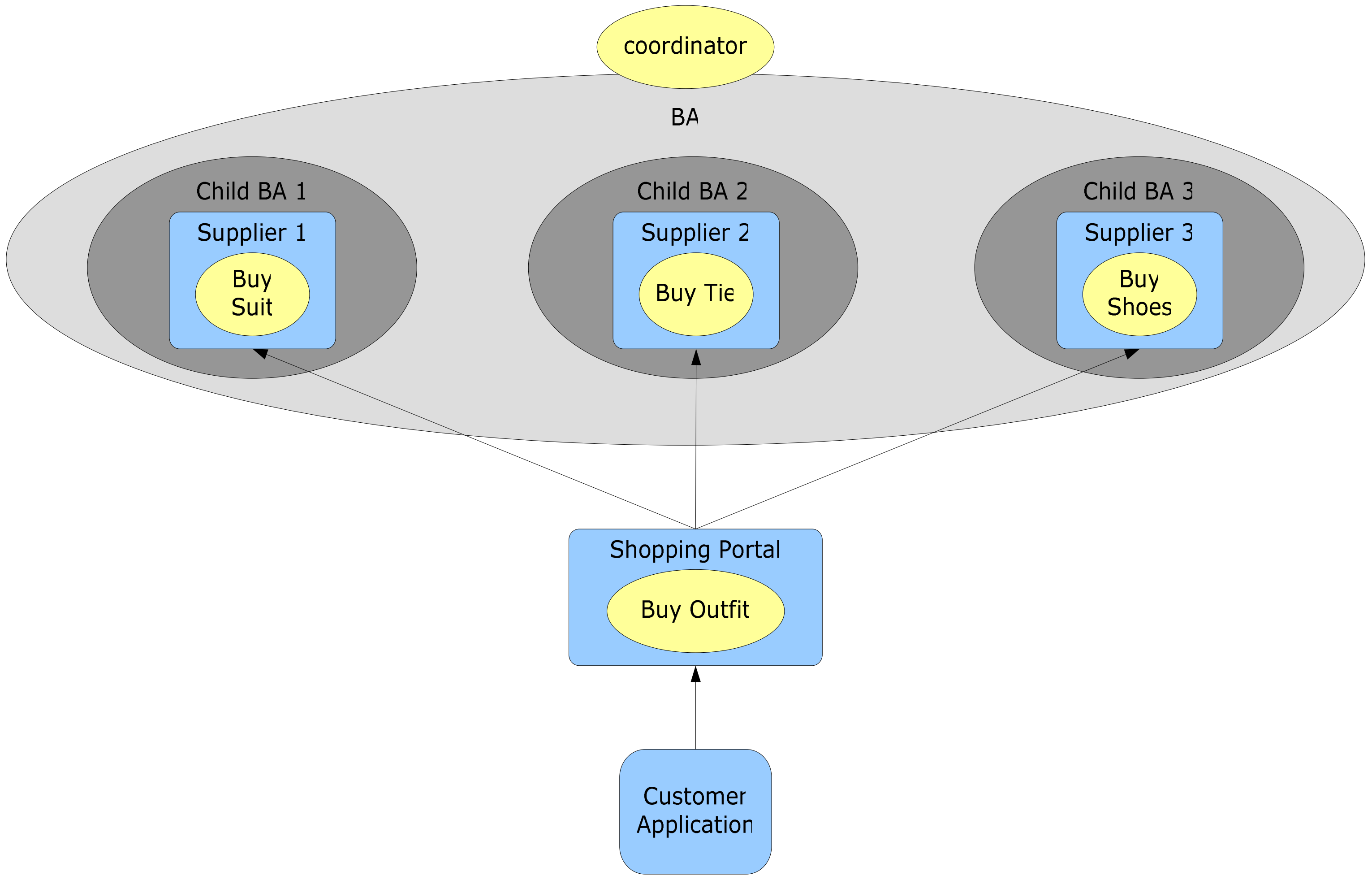
arjuna
middleware for reliability

- Assume ACID transactions
  - High degree of trust
  - Isolation for duration of transaction
  - Backward compensation techniques
  - Does not allow heuristic outcomes
- Integration with existing transaction systems
  - Important to leverage investments
- Interoperability between transaction systems
  - Something of a holy grail to date

Business Activities

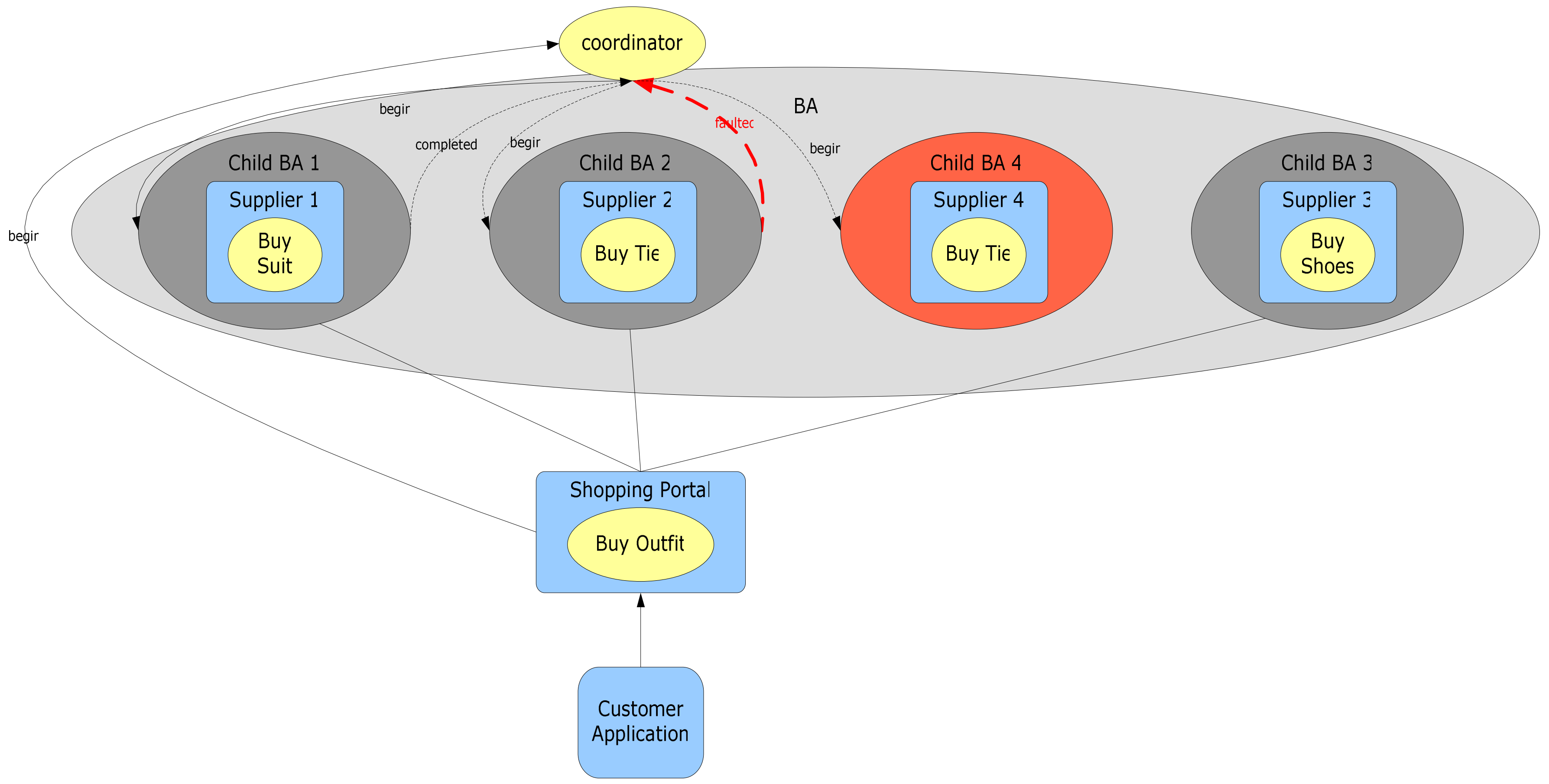
- Workflow-like coordination and management
- Business activity can be partitioned into tasks
  - Parent and child relationships
    - Select subset of children to complete
    - Parent can deal with child failures without compromising forward progress
- Tasks can dynamically exist a business activity
- Tasks can indicate outcome earlier than termination
  - Up-calls rather than just down-calls

BA example


coordinator
BA
Child BA 1
Supplier 1
Buy Suit
Child BA 2
Supplier 2
Buy Tie
Child BA 3
Supplier 3
Buy Shoes
Shopping Portal
Buy Outfit
Customer Application

Compensating BA

arjuna
middleware for reliability



OASIS WS-TXM

- Specification from Arjuna, Fujitsu, IONA, Oracle, Sun and others
  - Part of WS-CAF
    - Three specifications
      - WS-Context
      - WS-Coordination Framework
      - WS-Transaction Management

Transaction models

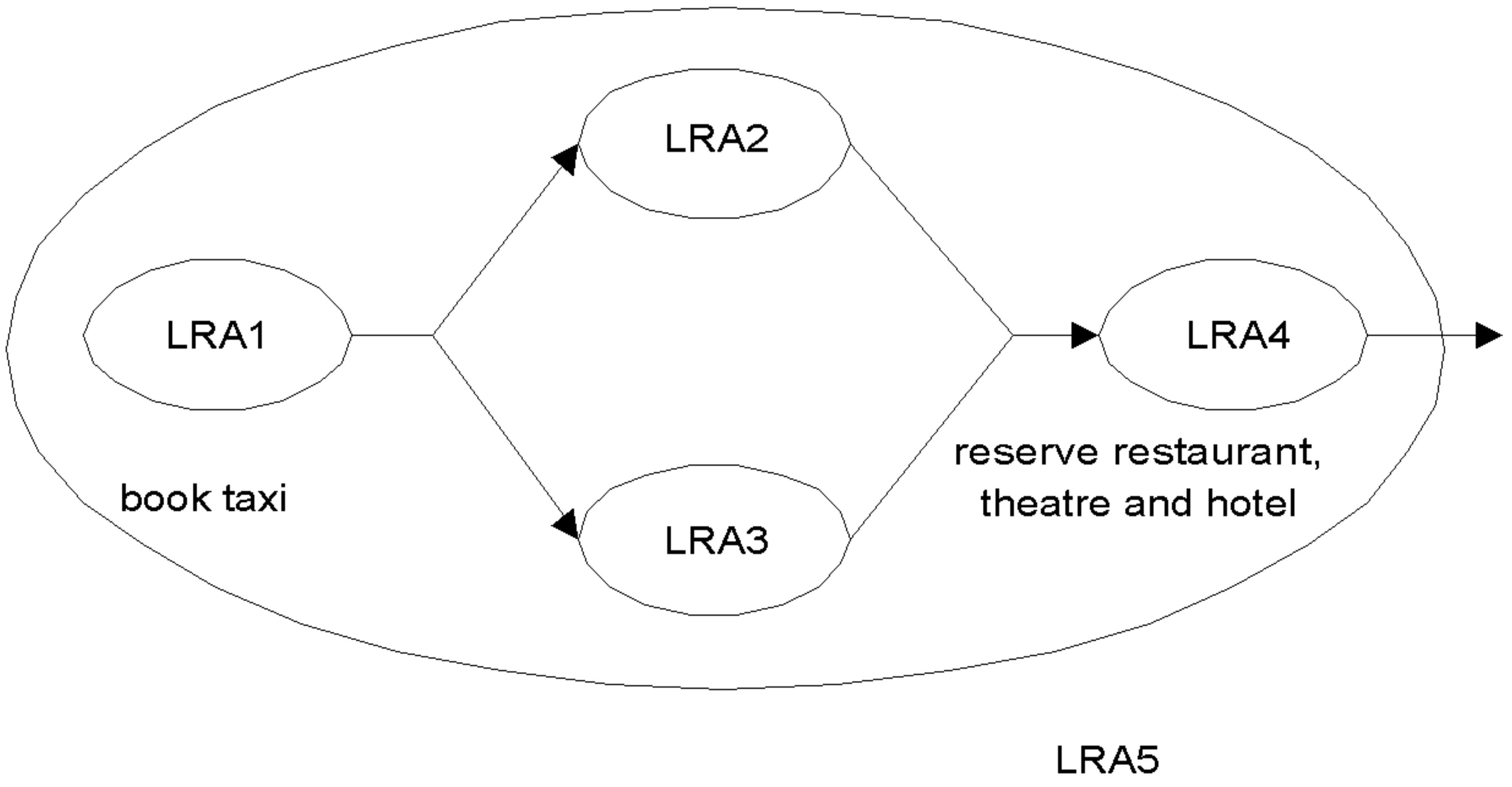
- Three models
  - ACID transaction
    - For interoperability and high-cost services where ACID transactions are a requirement
      - Heuristics allowed
  - Long running action
    - Loosely coupled, long duration work that uses compensations
  - Business process
    - For gluing together different transaction models, with different implementations, into a single global transaction

Long running action

arjuna middleware for reliability

- Specifically for long duration interactions
  - Could be used for short duration
- Spheres of compensation
  - Can be nested (parent-child relationship)
- Compensation actions
  - Return the business state to consistency
    - E.g., credit your credit card and give you back interest payments

Example


LRA1
LRA2
LRA3
LRA4
LRA5
book taxi
reserve restaurant, theatre and hotel

Business process

- All parties reside within *business domains*
  - May represent a different transaction model and implementation
    - ACID, compensation, message-oriented, …
- Business process is split into *business tasks*
  - Compensatable units of work
    - Forward compensation during activity is allowed
  - Can be check-pointed and restarted by business process during flow of activity
    - Support for manual intervention

Commonality

- ACID transaction model
  - Interoperability with existing infrastructures
  - Well understood model and semantics
    - Lots of tool support
- Compensation model
  - Forward recovery
  - Better model for long duration interactions
    - Requires more work from applications and users
      - Potentially more complex model
    - Compensations specific to requirements
      - Requires tool support

Are they sufficient?

arjuna
middleware for reliability

- WS-AT/WS-BA and WS-TXM have similar roots
  - Micro-protocol approach
    - One-size does not fit all
    - Tailor transaction model to specific requirements
- The models suit current use cases
  - Further expansion is allowed
  - Web services evolution

Conclusions

- Very active subject
- Two models are common
  - Backward compensation
    - ACID for interoperability
  - Forward compensation
    - Matches business models and allows independent structuring
- Try them out!
  - Should there be additional protocols?