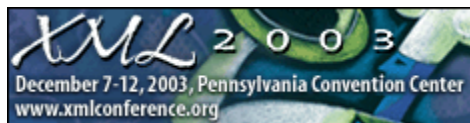


Web services transactions: past, present and future

Mark Little,
Arjuna Technologies Ltd



Overview



- ACID transactions and why they don't cut it in the world of Web Services
 - Consider long-duration activities
- Where are we?
 - OASIS BTP
 - WS-C/T
 - OASIS WS-CAF
- The future

ACID transactions

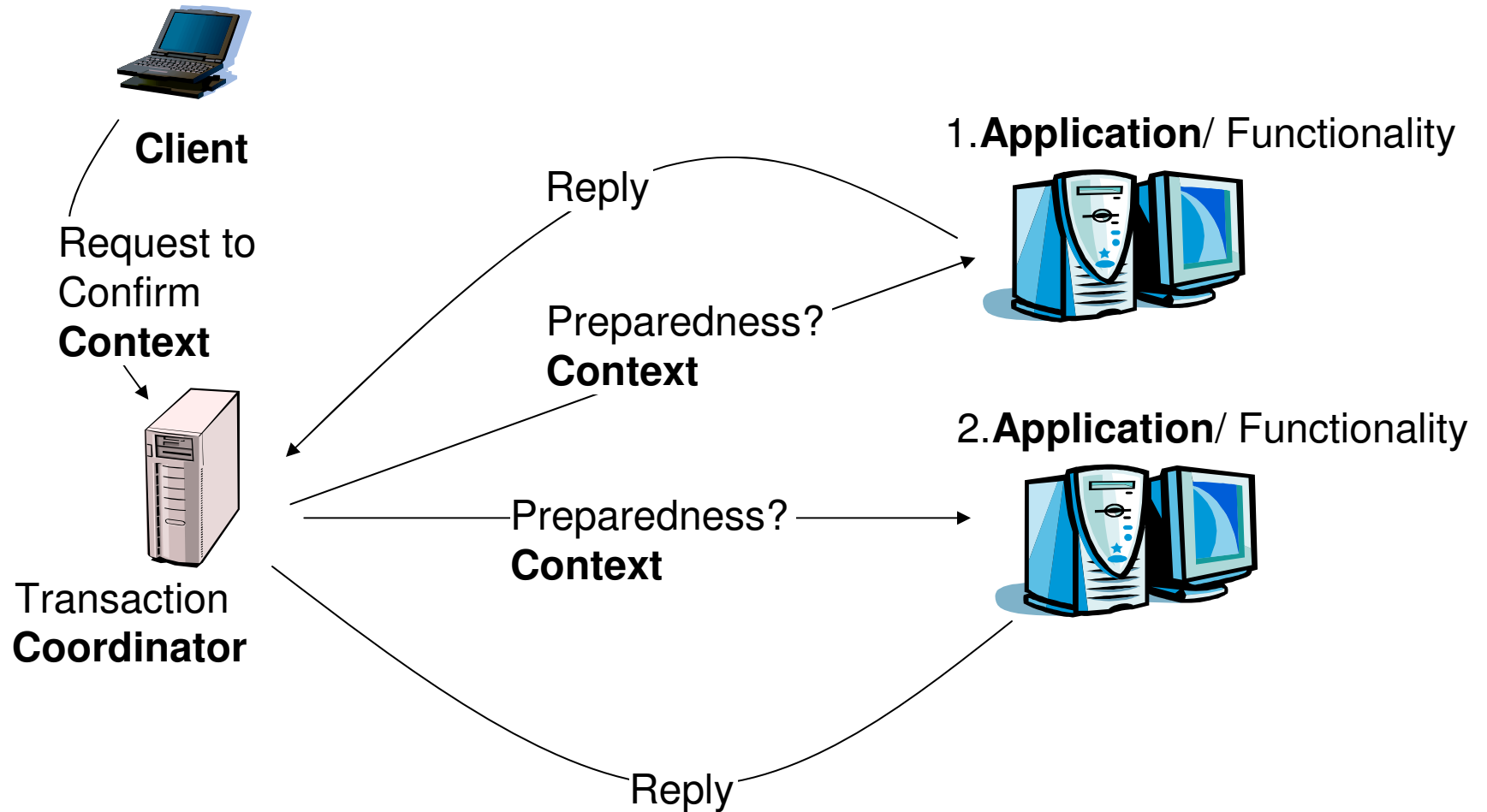


- *ACID guarantees*
 - Atomic
 - Consistent
 - Isolated
 - Durable
- Implicit contract that exists between
 - Transaction coordinator
 - E.g., HPTS, CICS, ...
 - Participants
 - E.g., XAResource

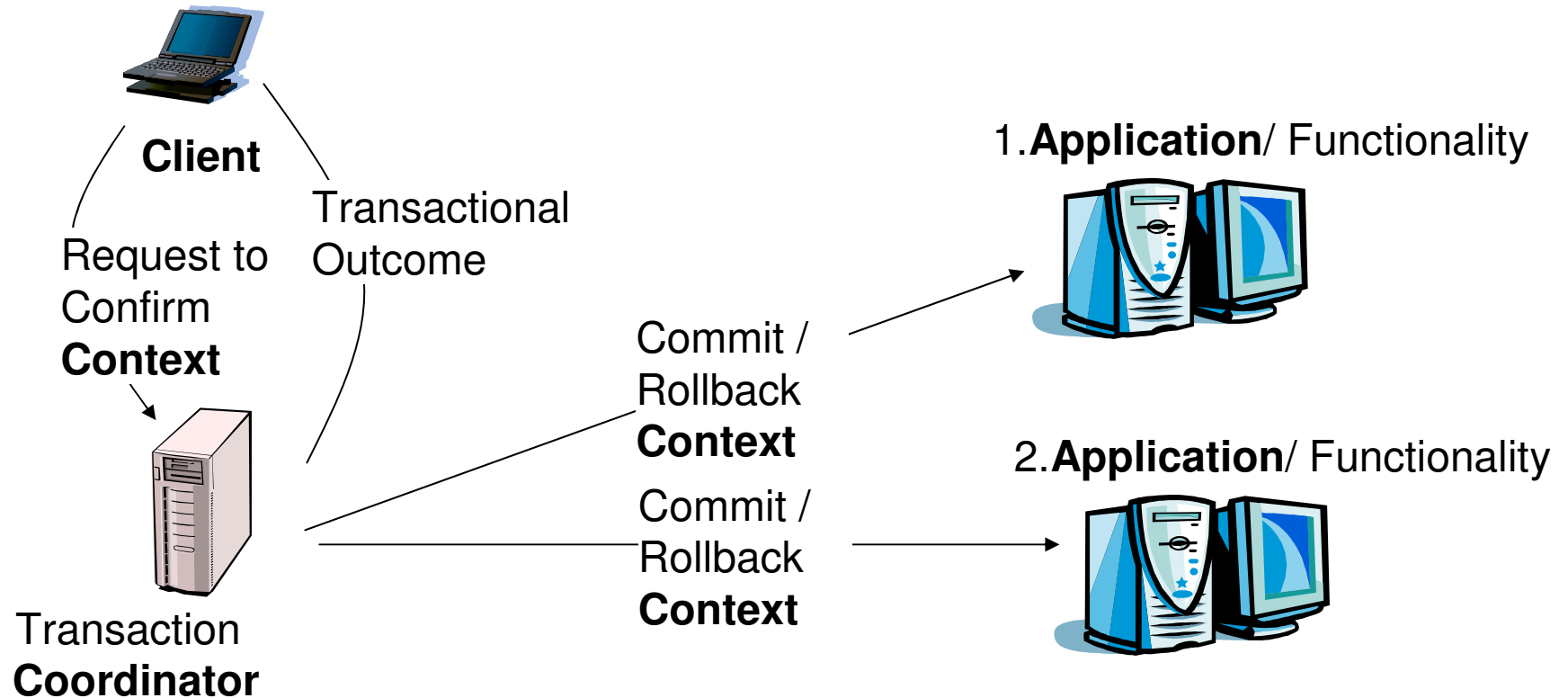
Termination protocol arjuna middleware for reliability

- Typically use a two-phase commit protocol
 - Prepare phase
 - Participants that can commit are required to record sufficient information to allow completion if failure
 - Either Commit phase
 - Coordinator records sufficient information to complete in case of failure
 - Or, Rollback phase

Phase one



Phase two



2PC is a protocol and does not define transaction qualities - i.e., ACID or isolation levels i.e., two phase locking.

Assumptions



- ACID transactions implicitly assume
 - Closely coupled environment
 - Short-duration activities
 - Must be able to cope with resources being locked for periods
- Therefore, do not work well for
 - Loosely coupled environments!
 - Long duration activities!

Web services



- Business-to-business interactions may be complex
 - involving many parties
 - spanning many different organisations
 - potentially lasting for hours or days
- B2B participants cannot afford to lock resources exclusively on behalf of an individual indefinitely
 - rules out the use of atomic transactions for many use cases

But ...



- Web Services are as much about interoperability as they are about the Web
- In the short term Web Services transactions will be about interoperability between existing TP systems rather than running transactions over the Web

Overall goals



- Transaction information must leverage the existing WS standards and initiatives
- ACIDity, specifically isolation needs to be relaxed such that parties can negotiate the transactional commitments at runtime.
 - Should also support ACID
 - consensus between participants, as illustrated in an atomic transaction, is extremely useful

OASIS BTP



- Developed by HP, Oracle, Sun, BEA and others
- First real standards attempt
- Defines two transaction models
 - Atoms
 - Cohesions

Atom



- Uses a two-phase termination protocol
 - prepare, confirm and cancel
 - There is an implicit contract between Atom and participant that work must be atomic
 - *All* participants will do the same thing
 - Does not mandate how to implement prepare, confirm and cancel
 - More flexibility than in ACID
 - Does not say anything about isolation

Cohesion

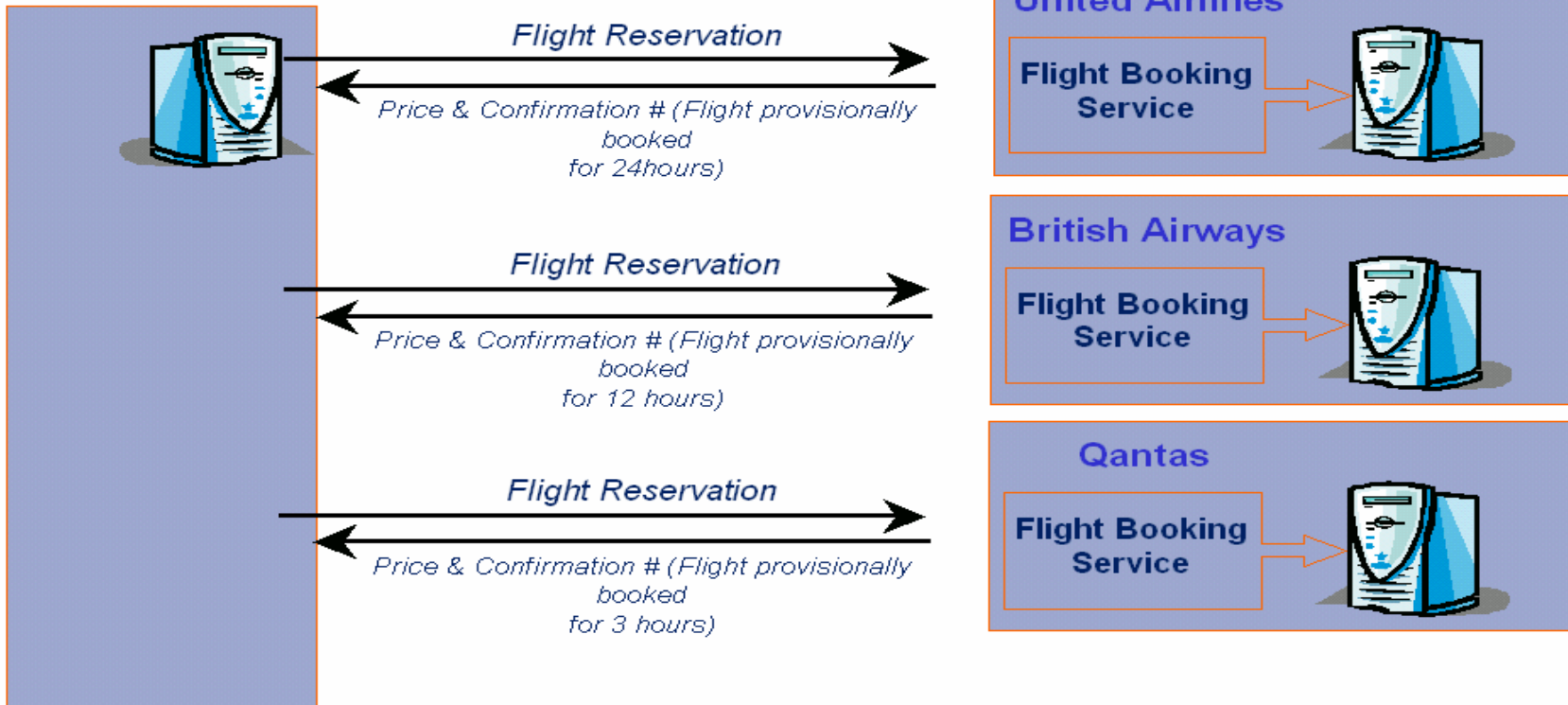


- prepare, confirm and cancel are parameterized
 - Work on (set of) Atom id(s)
 - Allows the confirm of a specific subset of work
 - Once subset is determined by business logic, it will be atomic

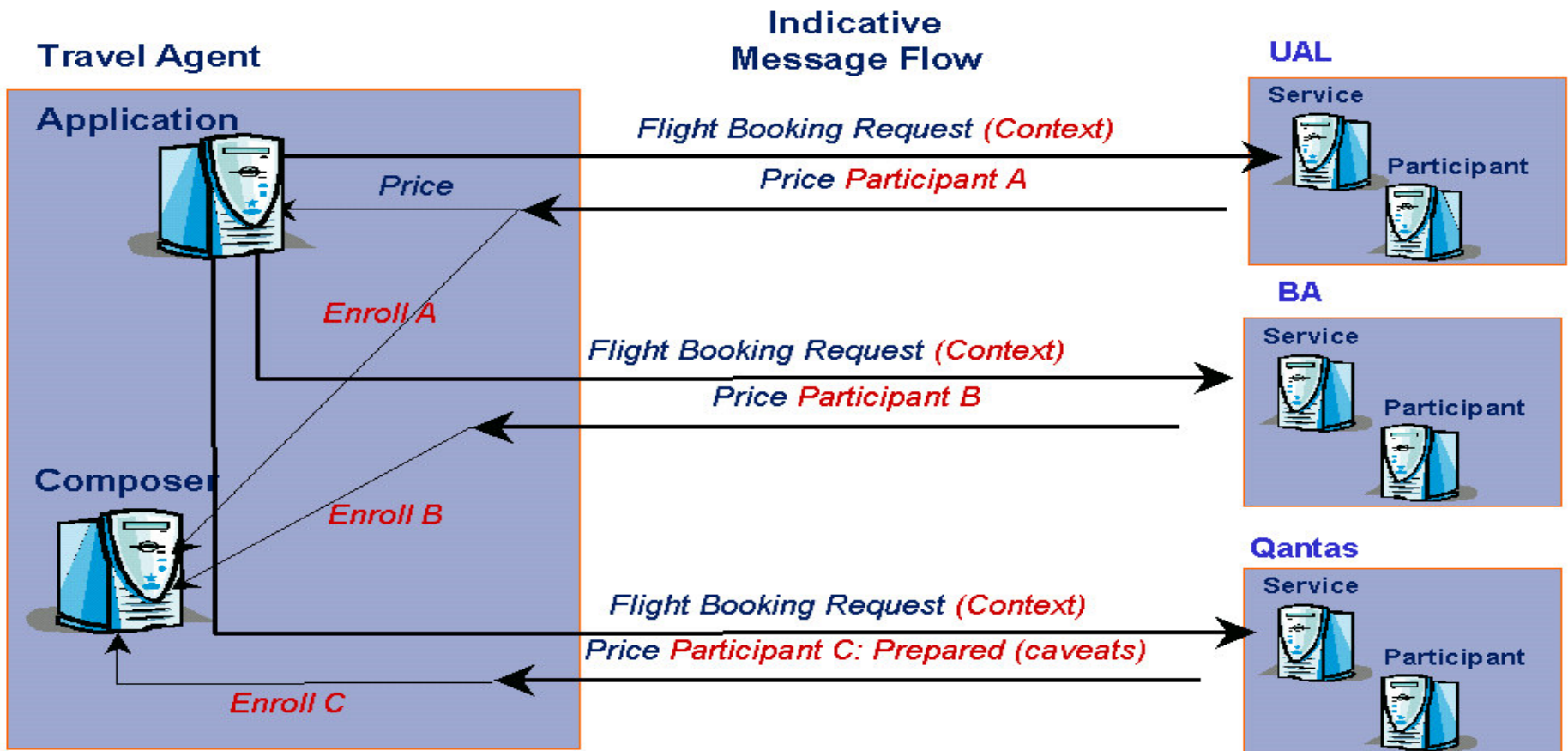
BTP | Single Service Type Cohesion

Travel Agent
(Consumer)

Message Flow

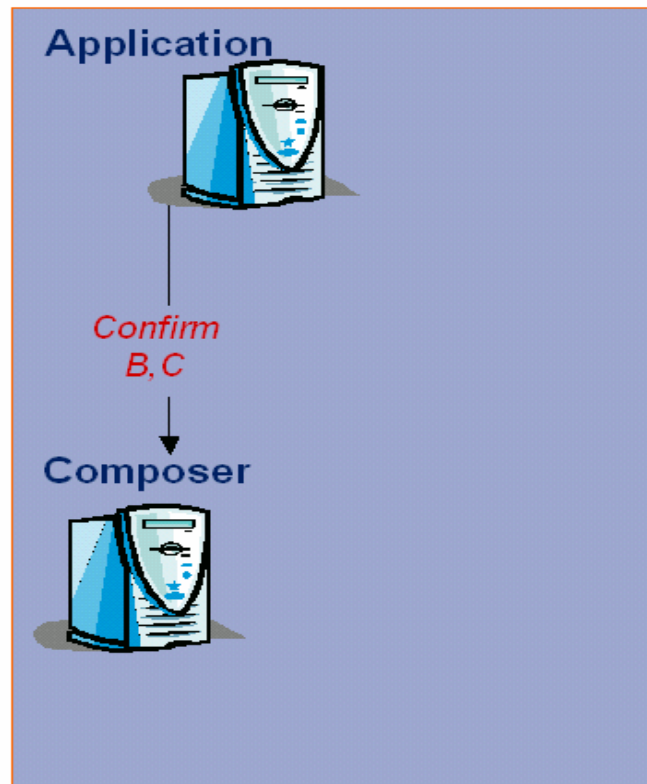


BTP | Single Service Type Cohesion



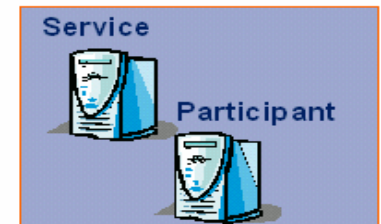
BTP | Single Service Type Cohesion

Travel Agent



Indicative
Message Flow

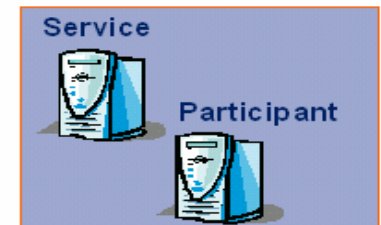
UAL



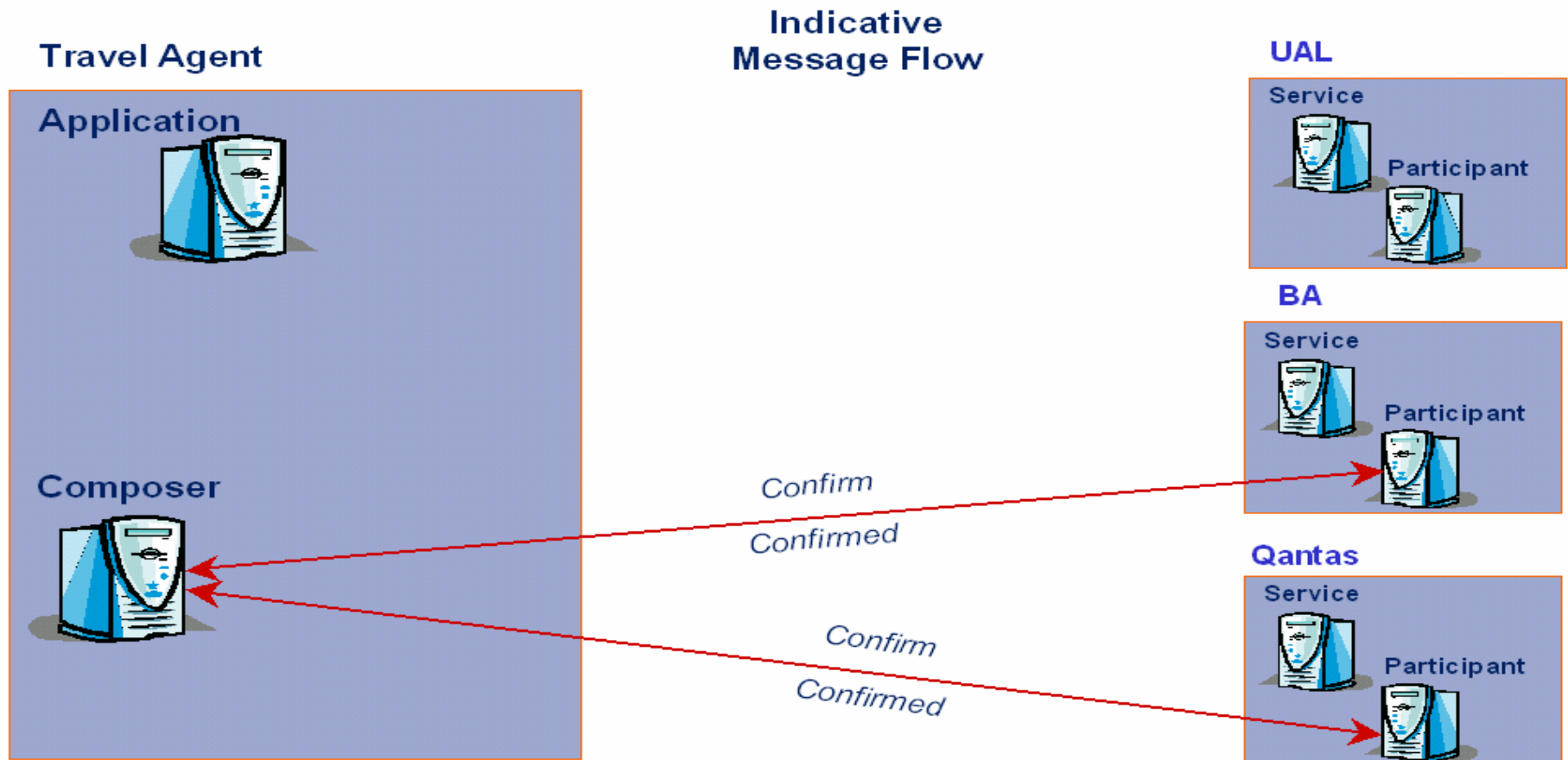
BA



Qantas



BTP | Single Service Type Cohesion



Relationship to Web Services



- Designed not to be Web Services specific
- Contexts and entire message set has been designed to be interoperable
 - Does not mandate a specific carried protocol
 - Could be SOAP, IIOP, carrier pigeon
 - Only mandates XML format for messages

Pros and Cons



- Pros
 - Well formed and complete
- Cons
 - 200+ pages!
 - Over complexity
 - Doesn't fit well in Web services architecture
 - Have to expose participants to end users
 - Business logic is encoded within transaction protocol
 - Really only one protocol that has to work for all use cases
 - Poor integration with existing TP infrastructures

WS-C/T



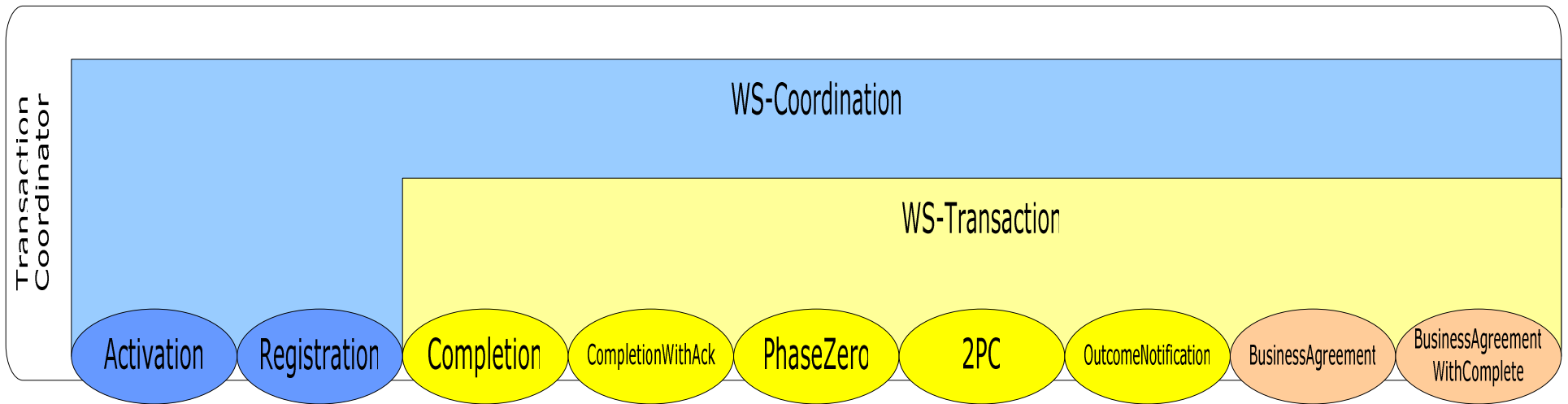
- Proprietary specifications released by IBM, Microsoft and BEA
- Separate coordination from transactions
- Define two transaction models
 - AtomicTransaction
 - Closely coupled, interoperability
 - Business Activities
 - Compensation based, for long duration activities

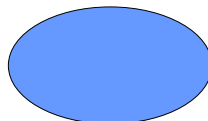
WS Coordination



- Coordination is more fundamental than transactions
 - Transactions, security, workflow, ...
 - But each use may require different protocol
 - Two-phase, three-phase, ...
- Define separate coordination service
 - Allow customisation for different protocols

WS-T and WS-C



 WS-Coordination Protocol Endpoint

 WS-Transaction Atomic Transaction Protocol Endpoint

 WS-Transaction Business Activity Protocol Endpoint

AtomicTransaction



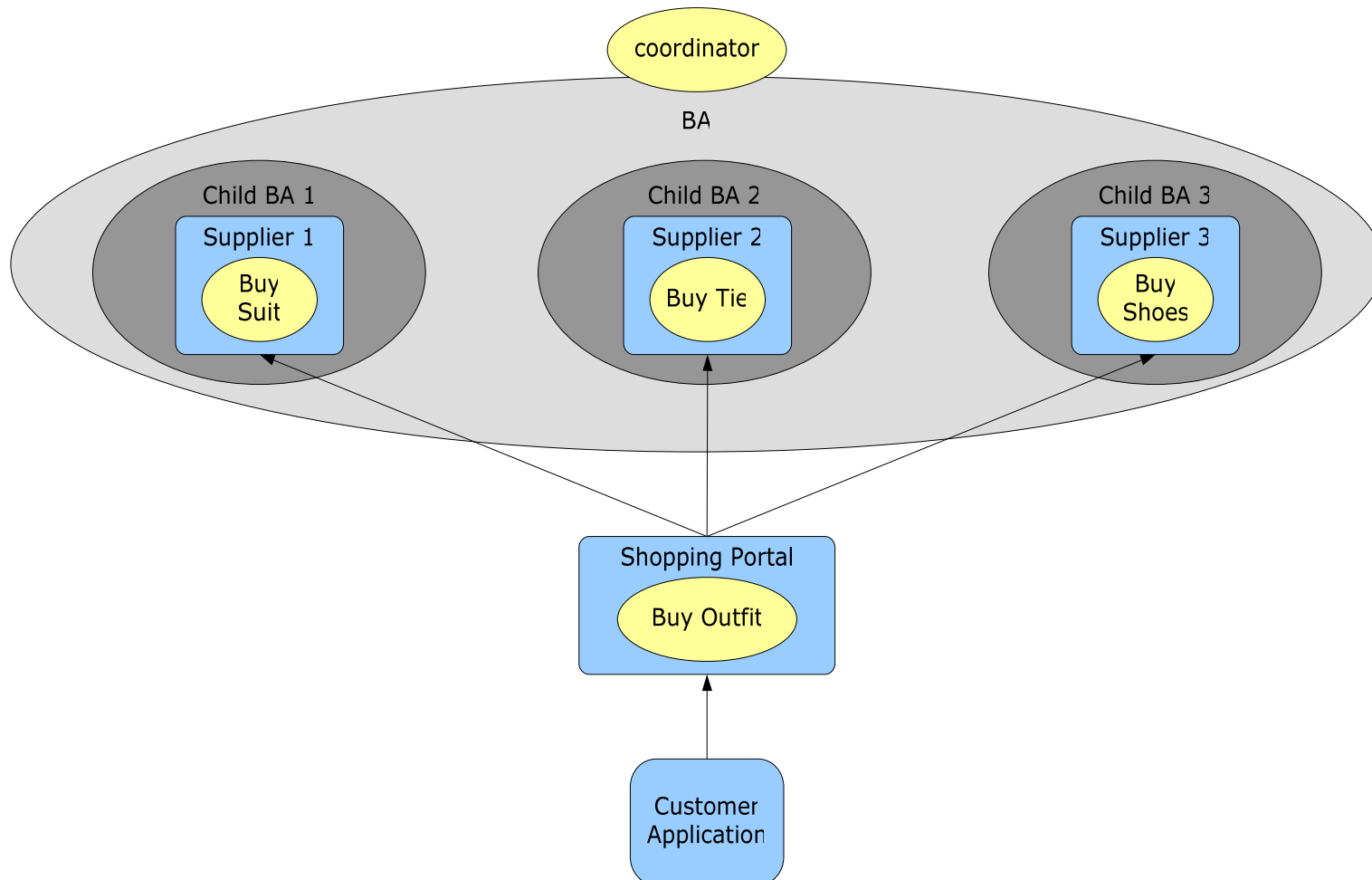
- Assumed ACID transactions
 - High degree of trust
 - Isolation for duration of transaction
 - Backward compensation techniques
- Integration with existing transaction systems
 - Should be possible to layer Web Services abstraction on them
- Interoperability between transaction systems

Business Activities

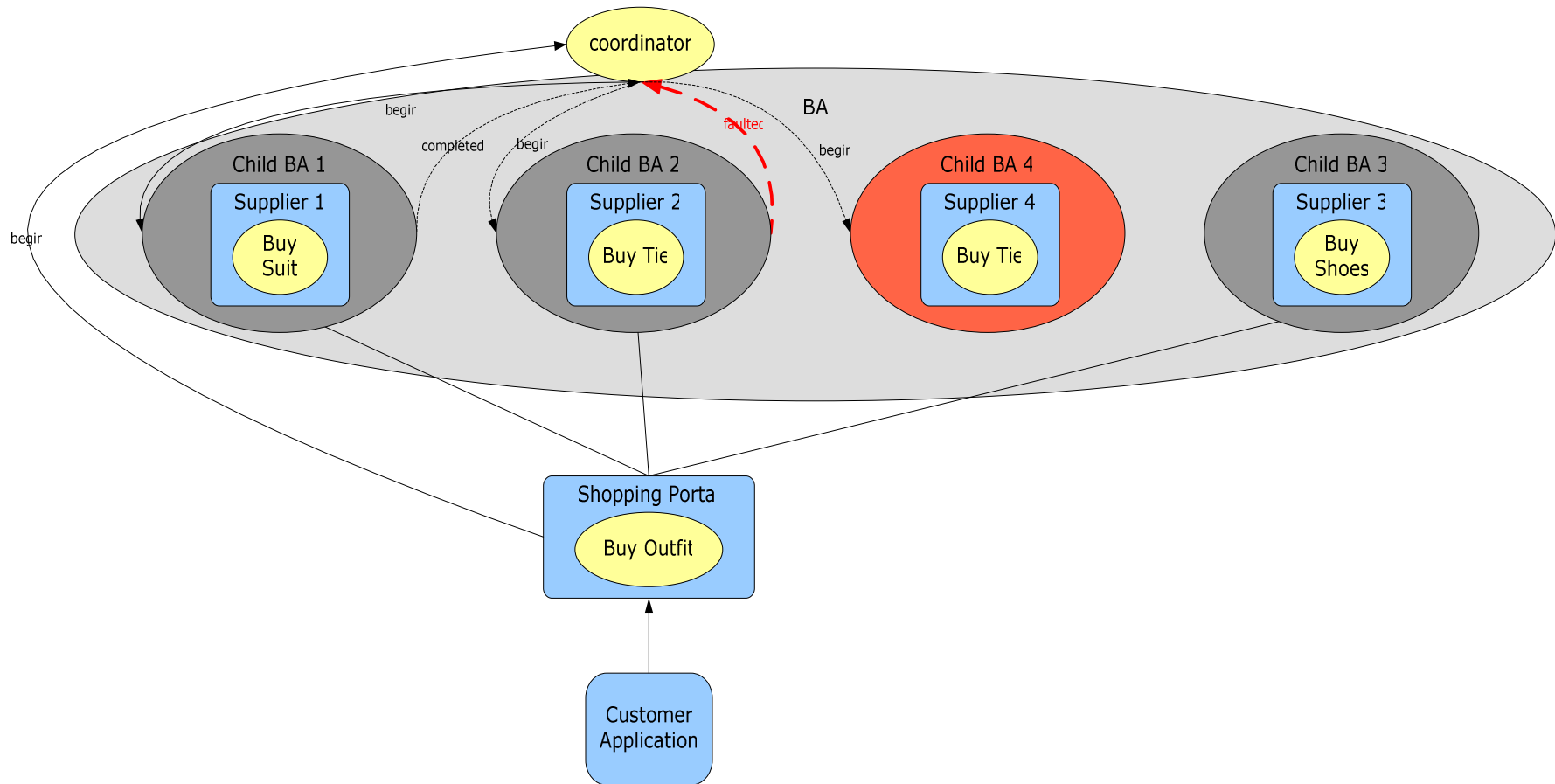


- Workflow-like coordination and management
 - Business activity can be partitioned into scopes (tasks)
 - Parent and child tasks
 - Select subset of children to complete
 - Parent can deal with child failures without affecting forward progress
 - Tasks can dynamically exist a business activity
 - Not interested in final outcome
 - Tasks can indicate outcome earlier than termination
 - Up-calls rather than just down-calls

BA example



Compensating BA



Pros and Cons



- Pros
 - Good separation of coordination from transactions
 - TP interoperability
 - The supporters!
- Cons
 - Incomplete specifications
 - Error conditions are poorly defined
 - Adversely affects interoperability
 - IPR

OASIS WS-CAF



- Supported by Oracle, Sun, IONA, Arjuna, Fujitsu, HP and others
 - Royalty free specifications
- Three specifications
 - WS-Context
 - WS-Coordination Framework
 - WS-Transaction Management
 - Three transaction models for Web services
 - Interoperability with existing implementations is important



WS-Context



- Context service
 - Fundamental aspect of WS architecture
- Defines notion of an activity
 - Unit of work
 - Precise definition left up to higher level services/users
 - Basic context associated with activity
- Context Service maintains context for each activity

WS-CF



- Provide a general framework for coordination protocols
 - Existing implementations to be plugged in
 - New implementations can be supported
 - Defines coordinator and participant relationships
- Work with WS-Context
 - Define an appropriate ALS
 - Augment context
- Scope of activity becomes scope of coordination boundary

WS-TXM



- Transactions for Web services
- Builds on WS-CF and WS-Context
- Based on experience of using Web service transactions
- Intended as a live document
 - New models can be added as required
- Scope of activity becomes scope of transaction



Models



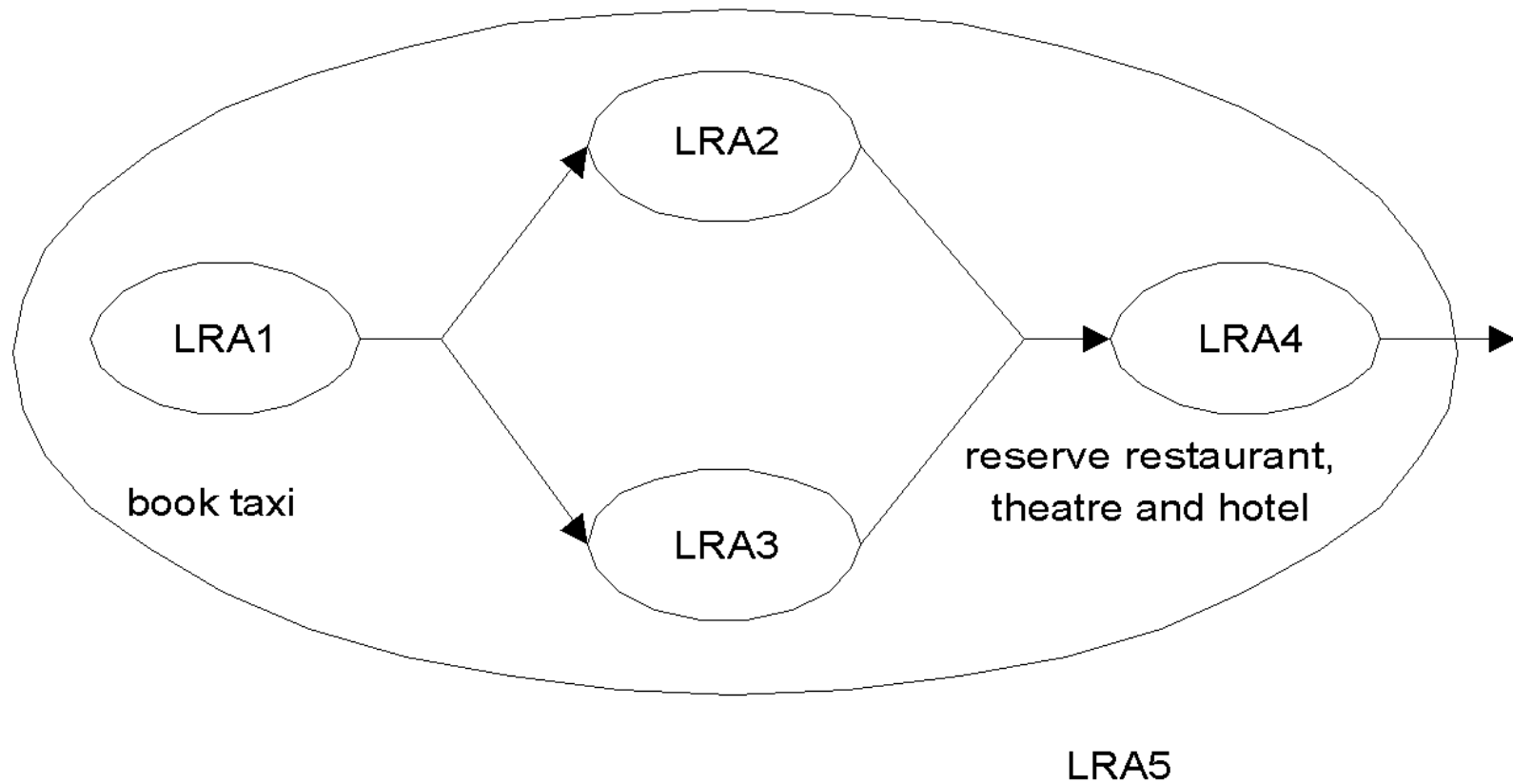
- Three transaction models
 - ACID transaction
 - For interoperability and high-cost services where ACID transactions are a requirement
 - Long running action
 - Loosely coupled, long duration work that uses compensations
 - Business process
 - For treating all steps in an automated business process as part of a single logical transaction

LRA



- Specifically for long duration interactions
- Compensation actions used
 - Forward work to return the business state to consistency
 - E.g., credit your credit card and give you back interest payments

Example

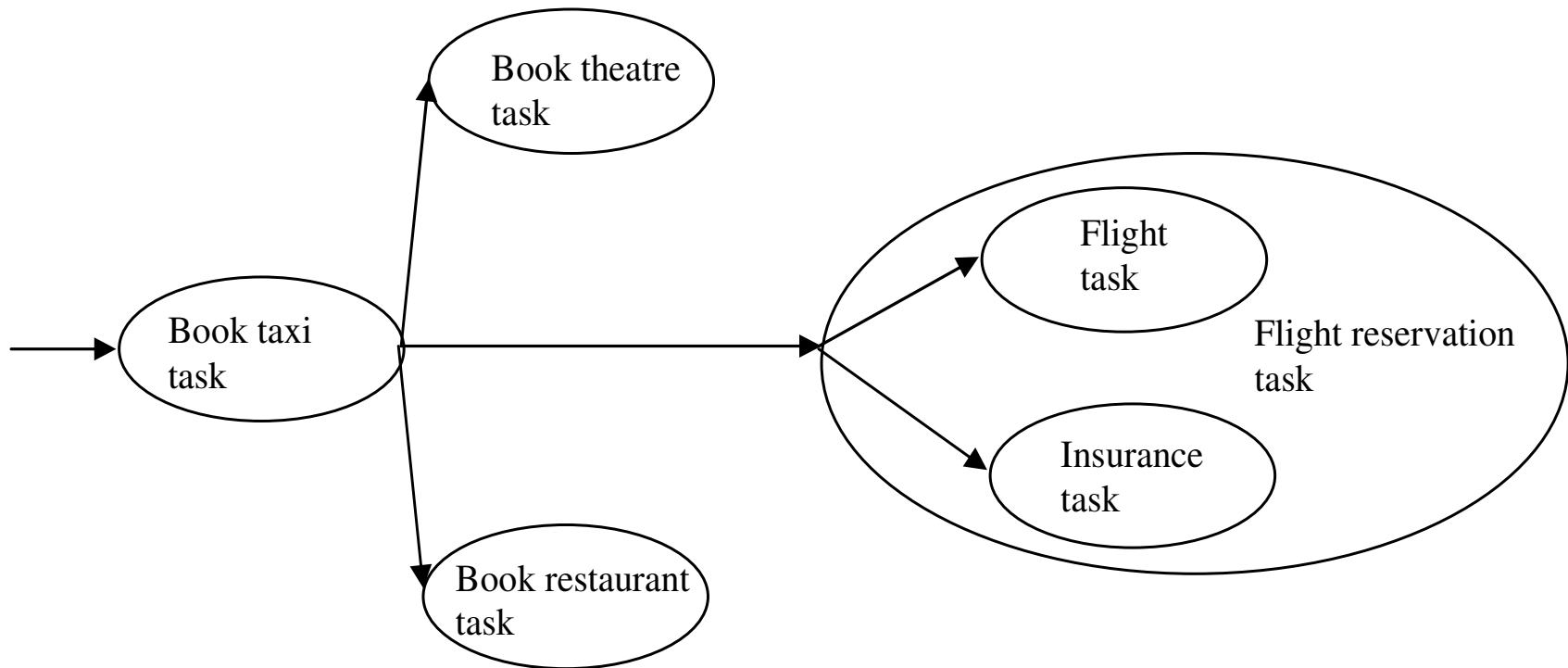


BP model



- All parties reside within *business domains*
 - Recursive structure is allowed
 - May represent a different transaction model
- Business process is split into *business tasks*
 - Execute within domains
 - Compensatable units of work
 - Forward compensation during activity is allowed
 - Keep business process making forward progress

Example



Pros and Cons



- Pros
 - Interoperability is important
 - Based on implementations
 - WS-Context
 - BP model
- Cons
 - Not backed by IBM and Microsoft
 - 18 months before it is a standard

Conclusions



- Very active subject!
 - Sometimes seems like we're going round in circles
- BTP was the first real attempt at a standard
 - Too complex
 - Not enough thought about leveraging existing infrastructures
 - Many existing TP systems couldn't be made BTP-aware
- WS-C/T and WS-CAF look promising
 - Leveraging existing investments is a priority
 - Similar enough to allow convergence
 - If all parties can agree!