MANCHESTER 1824

The University of Manchester

Advance Reservation and Co-Allocation Protocol

Dean Kuo
School of Computer Science
The e-Science North West Centre
The University of Manchester
dkuo@cs.man.ac.uk

MANCHESTER 1824

The University of Manchester

Assumptions

- Message latency is unbounded
  - Reliable messaging will only guarantee that a message will eventually be delivered in FIFO order
  - A service can not tell if another service has failed or it is a slow network or processor
- Consumer and providers can belong in different administrative domains
  - A consumer must not be able to “lock” a slot
    - Providers must have full control ***AT ALL TIMES***
    - Locking introduces the risk of denial of service attacks

MANCHESTER 1824

The University of Manchester

Cancellation

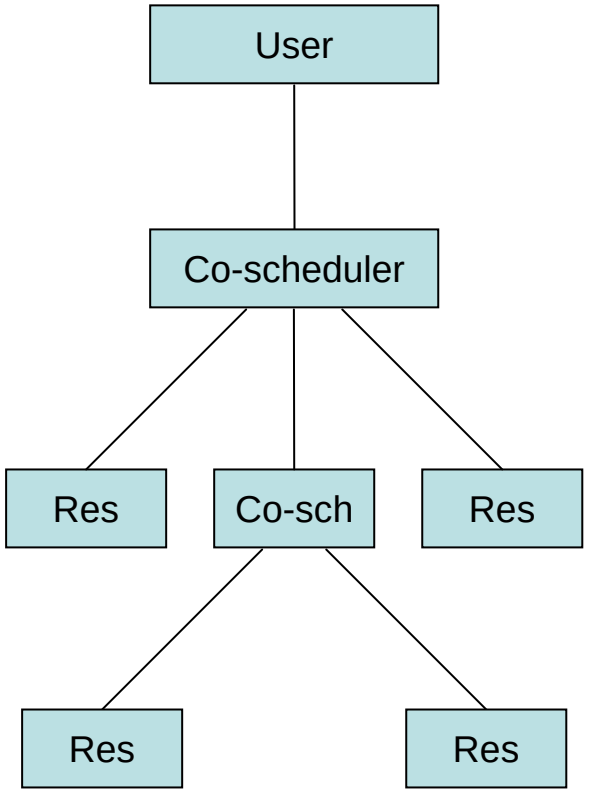
- Advance reservation needs to support cancellation
  - Providers must be able to cancel a reservation at anytime
    - Required for unscheduled downtime
    - It could cancel straight after it has agreed to a reservation request or seconds before the scheduled start time
    - Cancel if the consumer does not confirm before timeout period has expired
  - Consumer may also cancel a reservation at anytime
    - Consumer may be required to pay a cancellation fee if the reservation has been confirmed
  - Two-phase commit and Paxos commit is an agreement protocol not intended to support cancellation

MANCHESTER 1824

The University of Manchester

Aim

- Specification of an *advance reservation protocol*
- Specify a *co-allocation protocol* using the *advance reservation protocol*
  - The following pairwise interactions are use the advance reservation protocol
    - User and co-scheduler
    - Each resource and co-scheduler
  - Support for nested configuration
  - Resource can not distinguish between a reservation directly from a user or a co-scheduler



MANCHESTER 1824

The University of Manchester

Unit of Work

- Unit of work (conversation) begins when user sends its first message
  - Terminates when the job execution completes
- The co-allocation protocol does not provide atomicity
  - It is ***impossible*** to support ***ALL*** or ***NOTHING*** property

MANCHESTER 1824

The University of Manchester

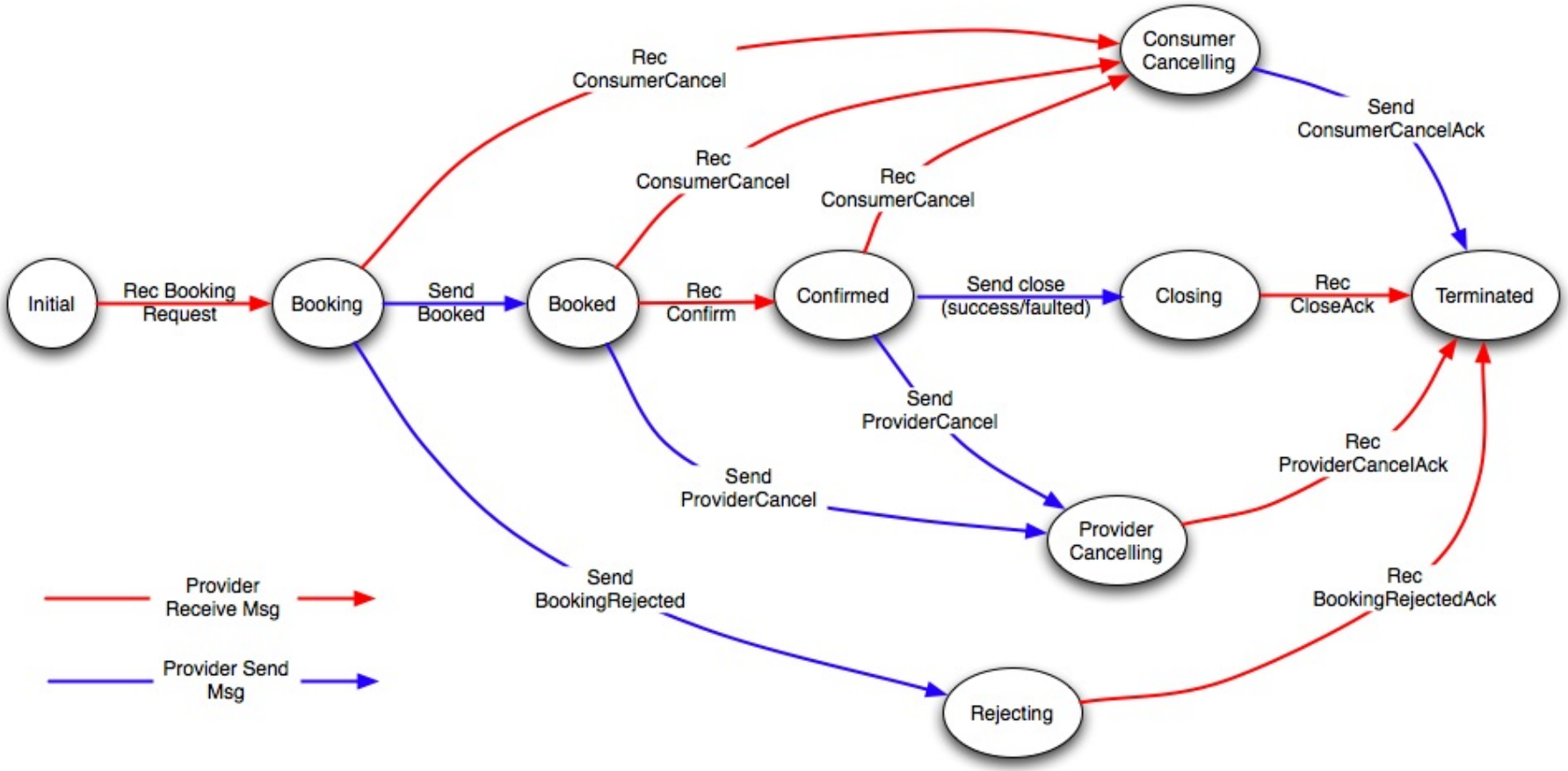
Advance Reservation Protocol

- 11 Messages between consumer and provider
  - Completed execution
    - BookingReq, Booked, Confirm, Close (successful/failed), CloseAck
    - Booked message include details of pricing and cancellation policy
  - Consumer Cancellation
    - ConsumerCancel, ConsumerCancelAck
  - Provider Cancellation
    - ProviderCancel, ProviderCancelAck
  - Booking rejected
    - Reject, RejectAck

MANCHESTER 1824

The University of Manchester

Provider Protocol



MANCHESTER 1824

The University of Manchester

Agreed Outcomes

- Booking request rejected
- Scheduled job executed
  - Successful or faulted
- Consumer cancelled
  - Consumer cancelled before confirmation then there is no fee
  - Consumer cancelled after confirmation then fees specified in the cancellation fee applies
- Provider cancelled
- Define charging model in terms of on final agreed outcome

MANCHESTER 1824

The University of Manchester

Race Situations

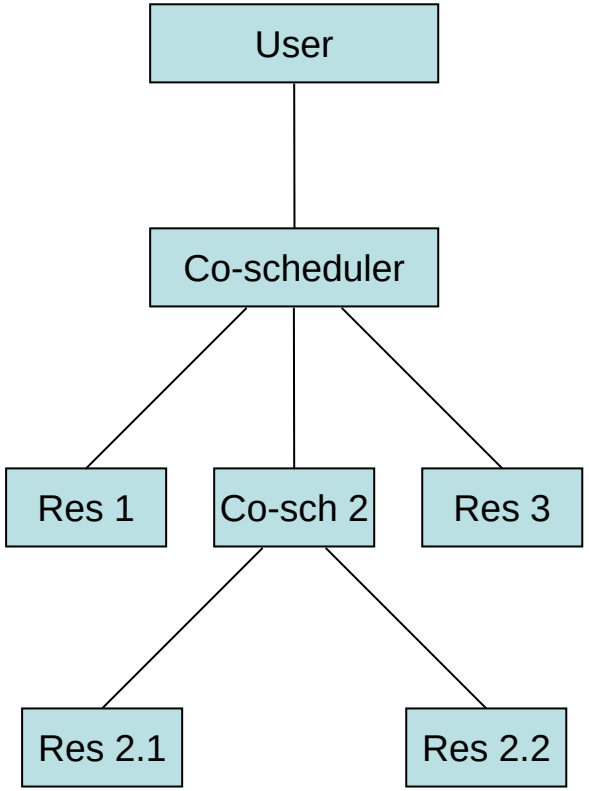
- What if consumer and provider cancel “simultaneously”?
  - Agreed outcome is “consumer cancelled”
- What if consumer cancels but when the provider receives the message, the job has already completed
  - Agreed outcome is “job completed”
- All possible race situations dealt with in the protocol

MANCHESTER 1824

The University of Manchester

Co-Allocation

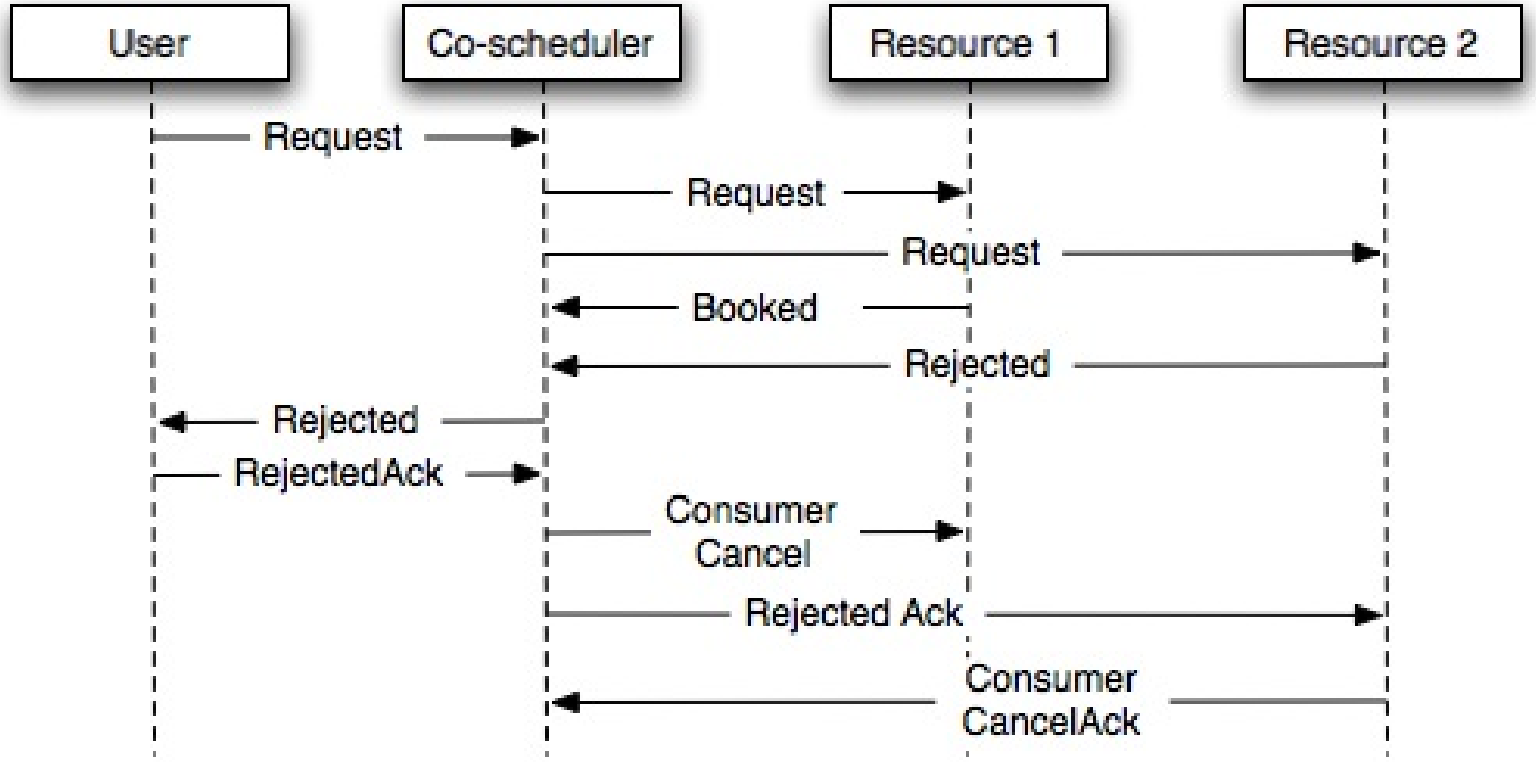
- Reservation of multiple resources
- Run the advance reservation protocol between
  - User and co-scheduler
  - Co-scheduler and Res 1
  - Co-scheduler and co-sch 2
  - Co-scheduler and Res 3
  - …
- Main benefit
  - Simplicity and nesting



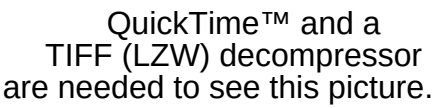
MANCHESTER 1824

The University of Manchester

Booking Rejected



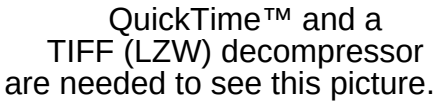
Co-Allocation Protocol - Completed



MANCHESTER 1824

The University of Manchester

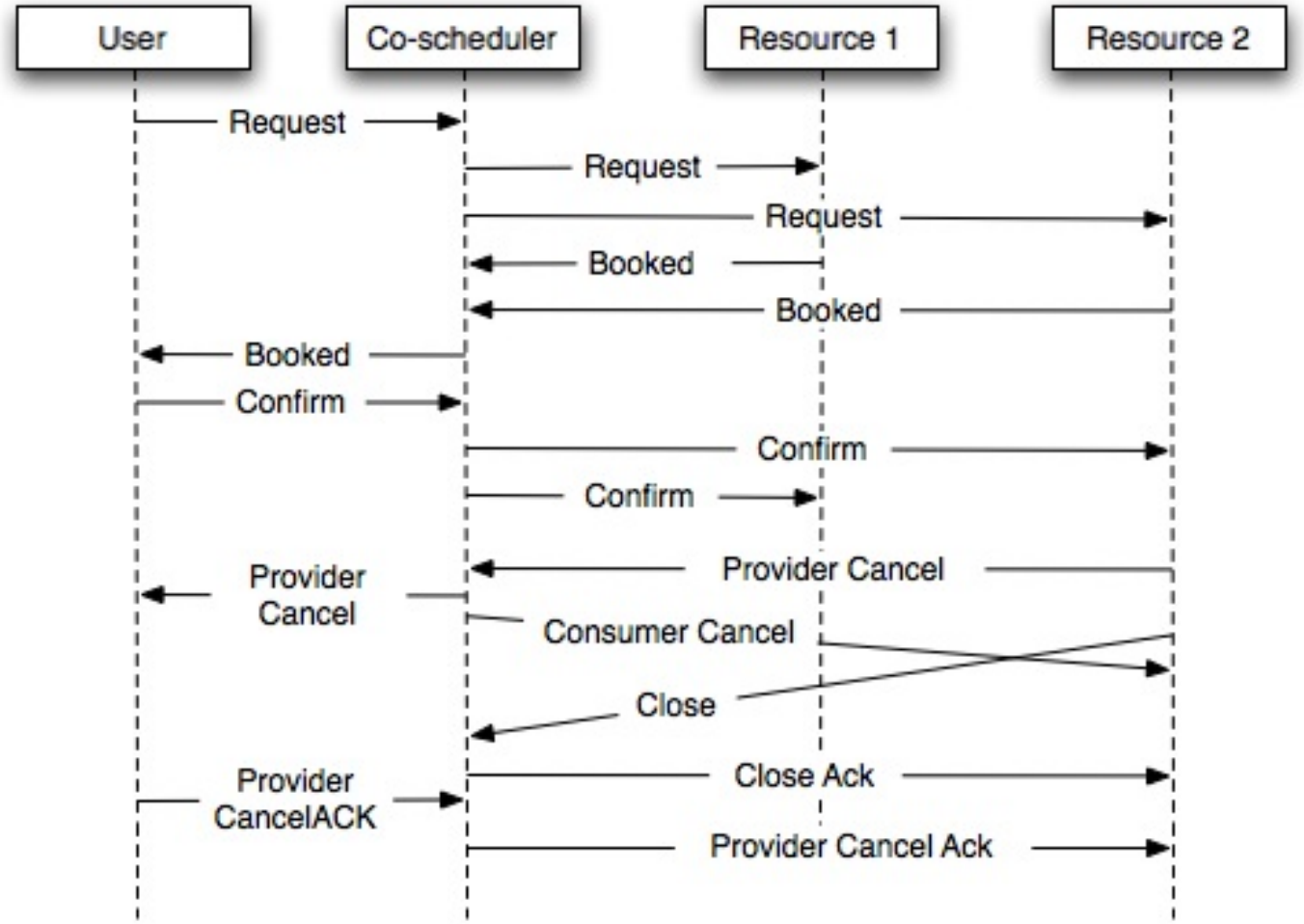
Co-Scheduler Cancels I



MANCHESTER 1824

The University of Manchester

Co-Scheduler Cancels



MANCHESTER 1824

The University of Manchester

Co-Allocation Protocol I

- Two phase to confirm a reservation
  - Cancellation can be initiated at anytime by any party
  - If consumer does not confirm a reservation before a timeout period then the provider cancel
    - Prevents denial of service attacks
- Atomicity is ***not*** supported
  - It is possible reservations with some resources terminated in the *completed* state while others terminate in the *provider cancelled or consumer cancelled* state
  - Co-allocation is not a consensus protocol

MANCHESTER 1824

The University of Manchester

Co-Allocation Protocol II

- Other co-allocation strategies can also be supported besides the two phase protocol
  - Co-scheduler reserve each resource sequentially
  - Greater chance where co-scheduler needs to cancel a confirmed reservation with a resource

MANCHESTER 1824

The University of Manchester

Correctness of Protocol

- Advance reservation protocol and co-allocation protocol have been modeled and verified using the SPIN model checker
  - Co-allocation protocol verification
    - Model contains only two resources

MANCHESTER 1824

The University of Manchester

Charging Framework I

- Defined in terms of the agreed outcome between the user and the co-scheduler
  - Rejected
    - No fee
  - Successful completion
    - Full price as specified in the booked message
  - Faulted execution
    - No fee or whatever is defined in the policy in the booked message

MANCHESTER 1824

The University of Manchester

Charging Framework II

- Consumer cancel
  - No fee if consumer cancels and has not sent a confirmation message
  - Fee as defined by the cancellation policy specified in the booked message
    - Co-scheduler may incur a financial loss in the co-allocation protocol
      - Less chance of incurring a loss for the two phase strategy
    - Co-scheduler can charge a premium to on all reservations to cover losses
- Provider Cancel
  - No fee or ?

MANCHESTER 1824

The University of Manchester

Last Minute Reservations

- The need for greater fault tolerance
  - Need to know if a reservation is confirmed or not once a provider sends a “booked” message
- Can we layer or merge Paxos commit with the advance reservation and co-allocation protocol?
  - Fault tolerance + support for cancellation

MANCHESTER 1824

The University of Manchester

Concluding Remarks

- Defined an advance reservation and co-allocation protocol based on a single protocol
  - Supports cancellation
  - A *unit of work* starts when user (consumer) initiates a reservation process and terminates when the job execution terminates
  - Includes a simple framework to support charging
  - Transaction atomicity is not possible as the protocol MUST be non-blocking and any party may cancel a reservation
    - Co-scheduler may incur a financial loss
  - Protocol provides a framework for explicitly defining charging models