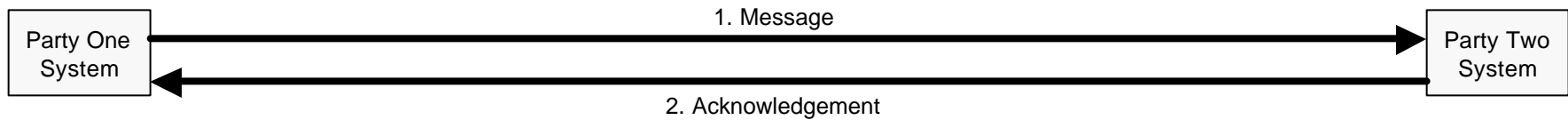


Multi-Hop Reliable Messaging Use Cases

Discussion paper for Tokyo F2F

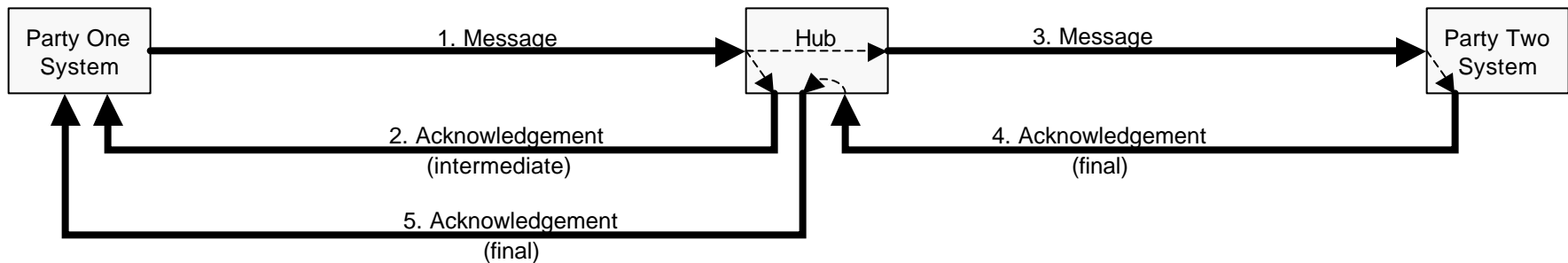
David Burdett
Commerce One
October 30, 2000

Use Cases – Explicit Acknowledgements

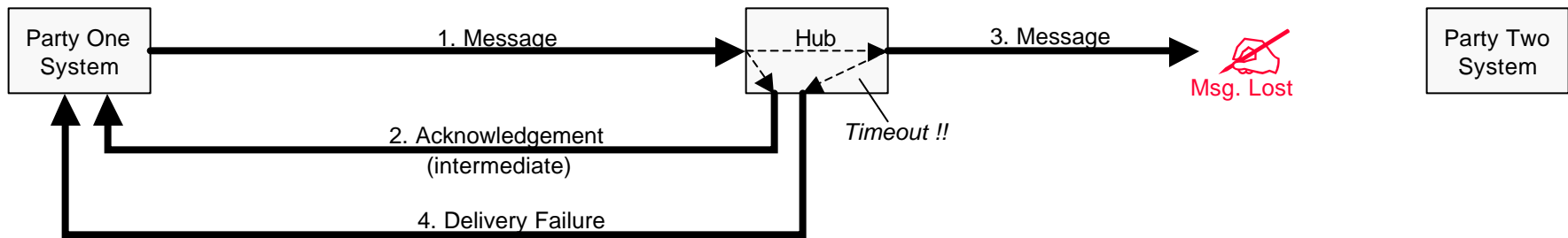


- A sender of a message may need to know that it has reached its final destination
 - requires final destination to acknowledge receipt

Use Case 1 - Acknowledgement by Destination

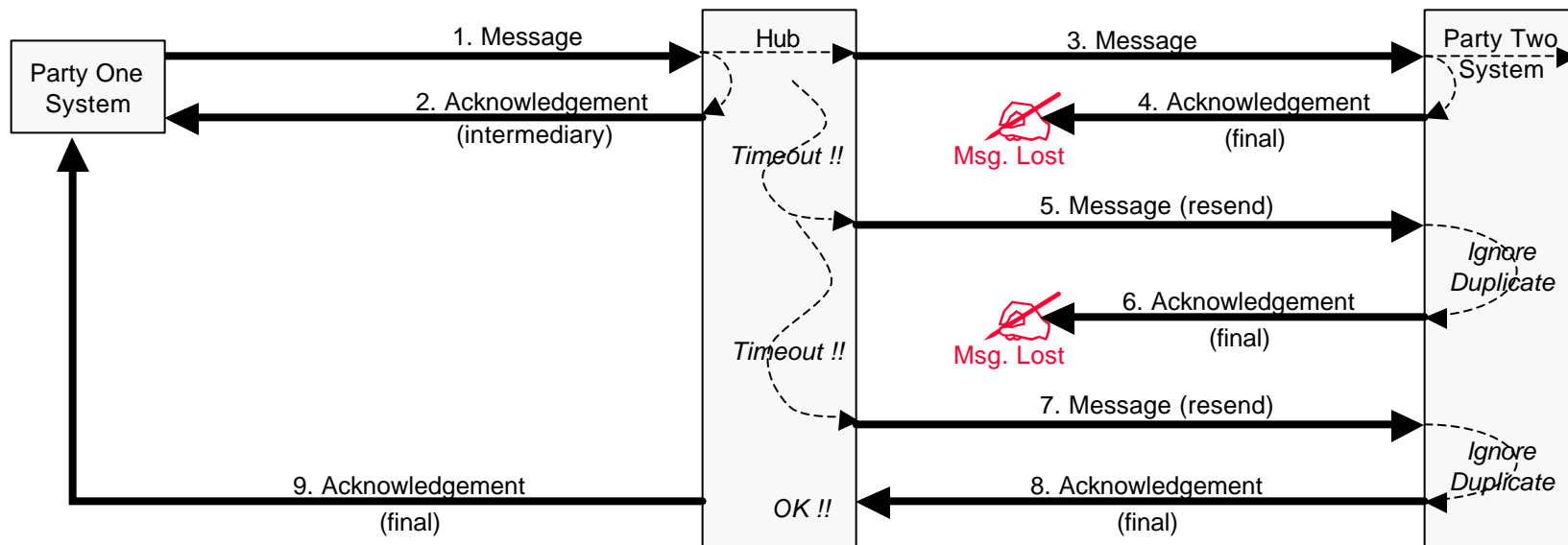


- A sender of a message may want to know that the message has reached an intermediate point - for example if one leg of trip uses a slow protocol such as SMTP
- Requires intermediate destination (hub) to:
 - acknowledge initial receipt, and
 - forward receipt from final destination

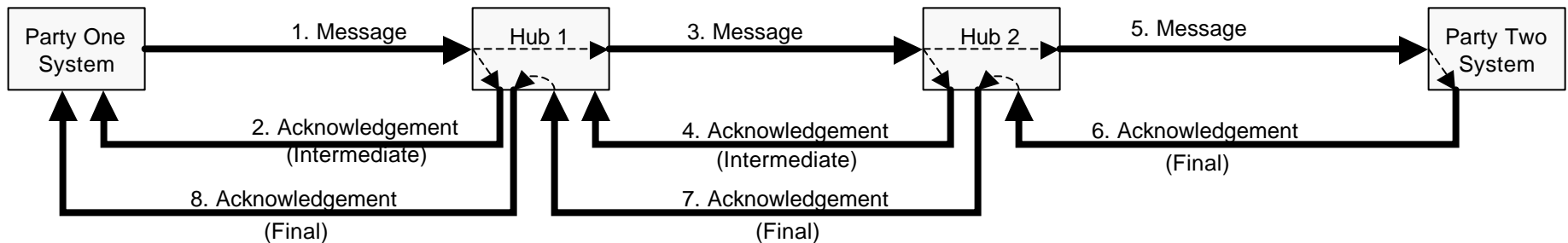


- A sender of a message may need to know if a message could not be delivered to its final destination
- Require intermediate destination (e.g. a hub) to:
 - notify the original sender of a message of its failure to forward it to its final destination

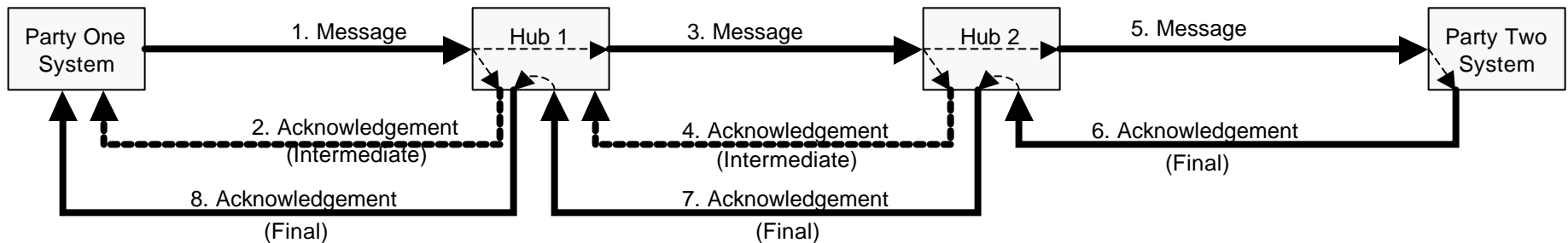
Use Case 3 - Reporting failed message delivery



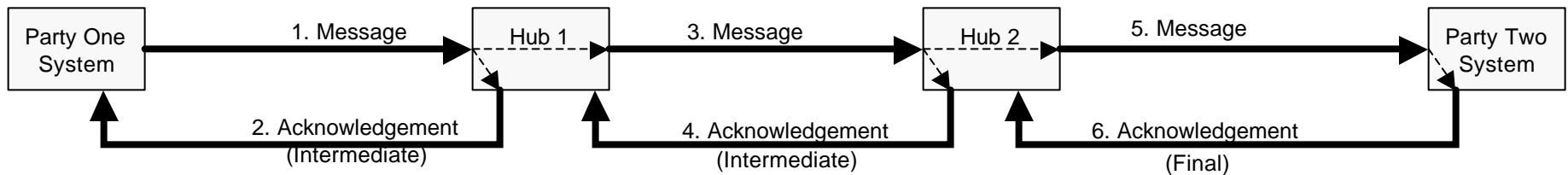
- A sender of a message should try several times before reporting failure
 - a second or later try might work
- Must work for inbound and outbound messages
- Requires recipient to ignore duplicates



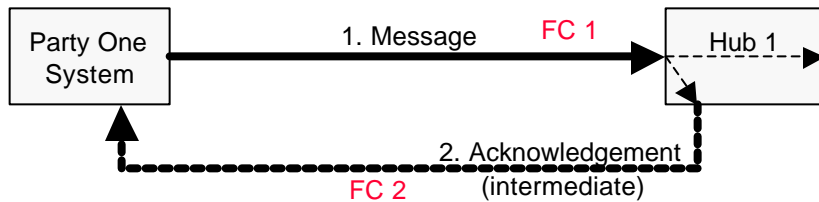
- The Final acknowledgement from Party Two should be sent back to Party One
- Earlier requirements (2, 3 and 4) must work equally well:
 - Over multiple hops
 - If different protocols/standards are used along the way



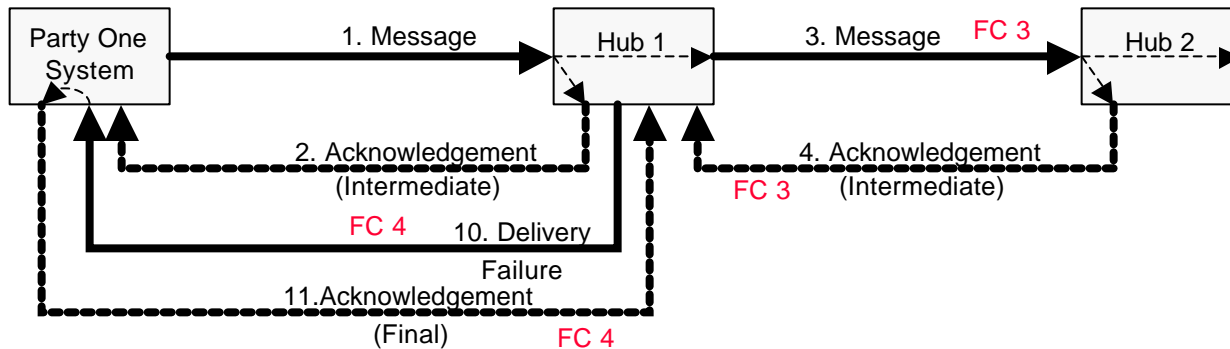
- Intermediate acknowledgements (messages 2 & 4 in diagram) are not needed if final acknowledgement can be returned within the time required



- There may be no need to send back the Final Acknowledgement to party one if:
 - Party one has received an intermediate ack, and
 - Party one knows that he will be informed of delivery failures

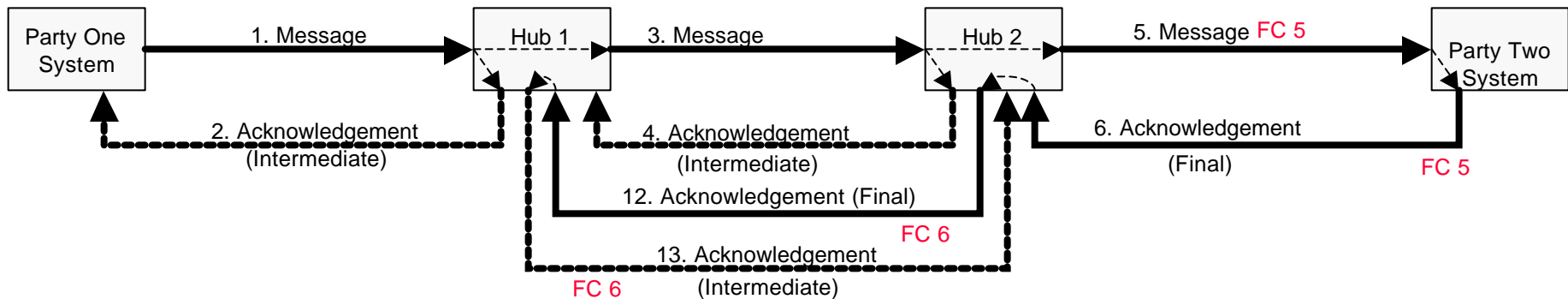


- Fail Case 1: If message 1 is not delivered:
 - re-send message 1 until message 2 is received, or
 - eventually give up
- Fail Case 2: If message 2 is not received:
 - same as Fail Case 1



- Fail Case 3: If message 3 is not delivered or message 4 not received:
 - re-send message 3 until message 4 is received, or
 - eventually give up and send message 10 Delivery Failure
- Fail Case 4: If message 10 not delivered, or message 11 not received
 - re-send message 10 until message 11 received, or
 - eventually give up and notify TP1 by other approaches if required
 - Note Delivery Failure needs to be Acknowledged

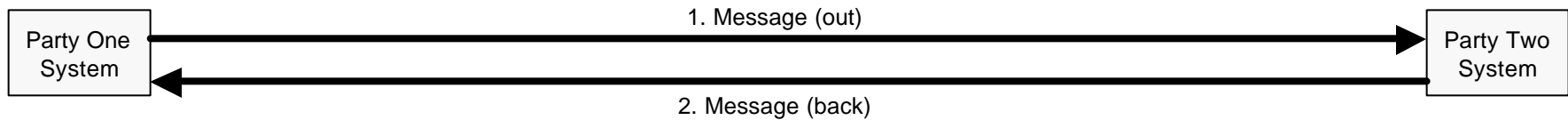
Failure Analysis - 2



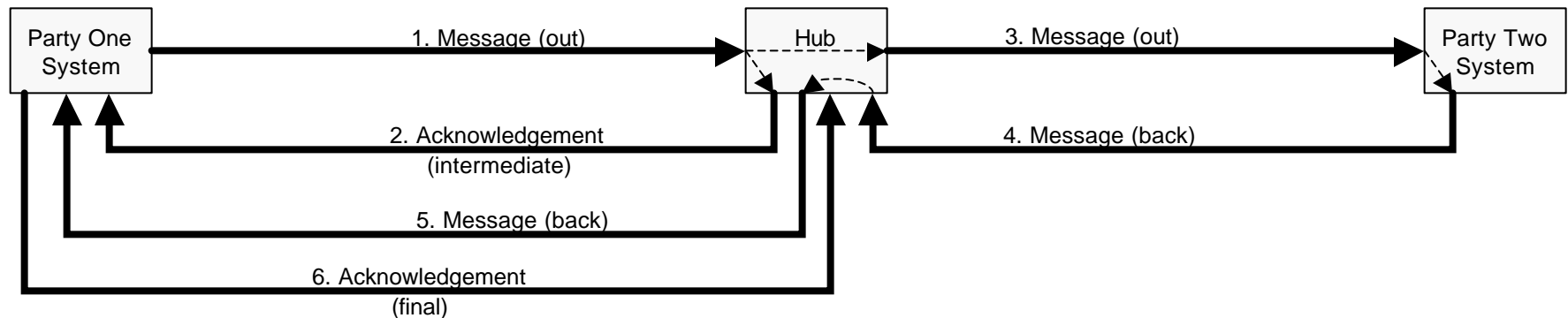
- Fail Case 5: Message 5 not delivered or message 6 not received:
 - re-send message 5 until message 6 received, or
 - eventually give up and send message 12
- Fail Case 6: Message 12 not delivered or message 13 not received:
 - re-send message 12 until message 13 received, or
 - eventually give up and rectify by other means
 - Note requires that Final Acknowledgement is itself acknowledged

Failure Analysis - 3

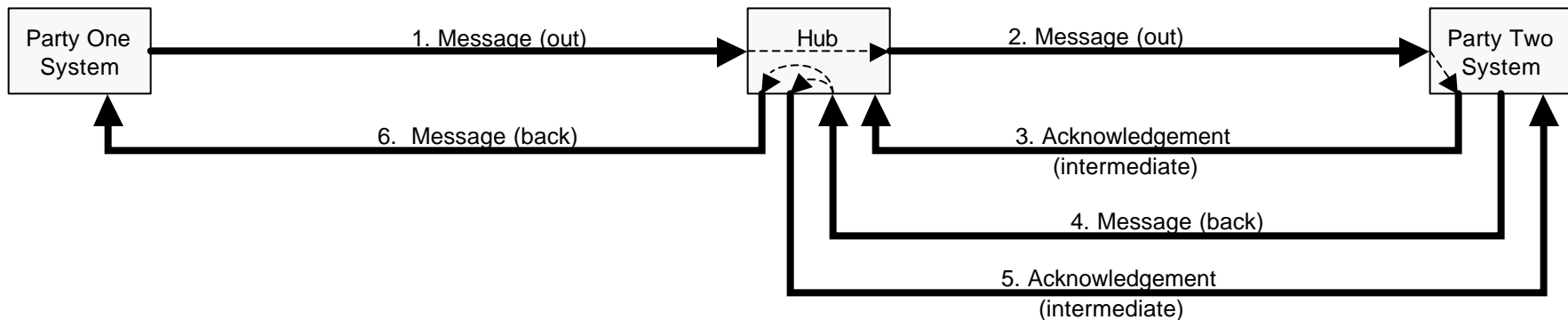
Use Cases – Implicit Acknowledgements



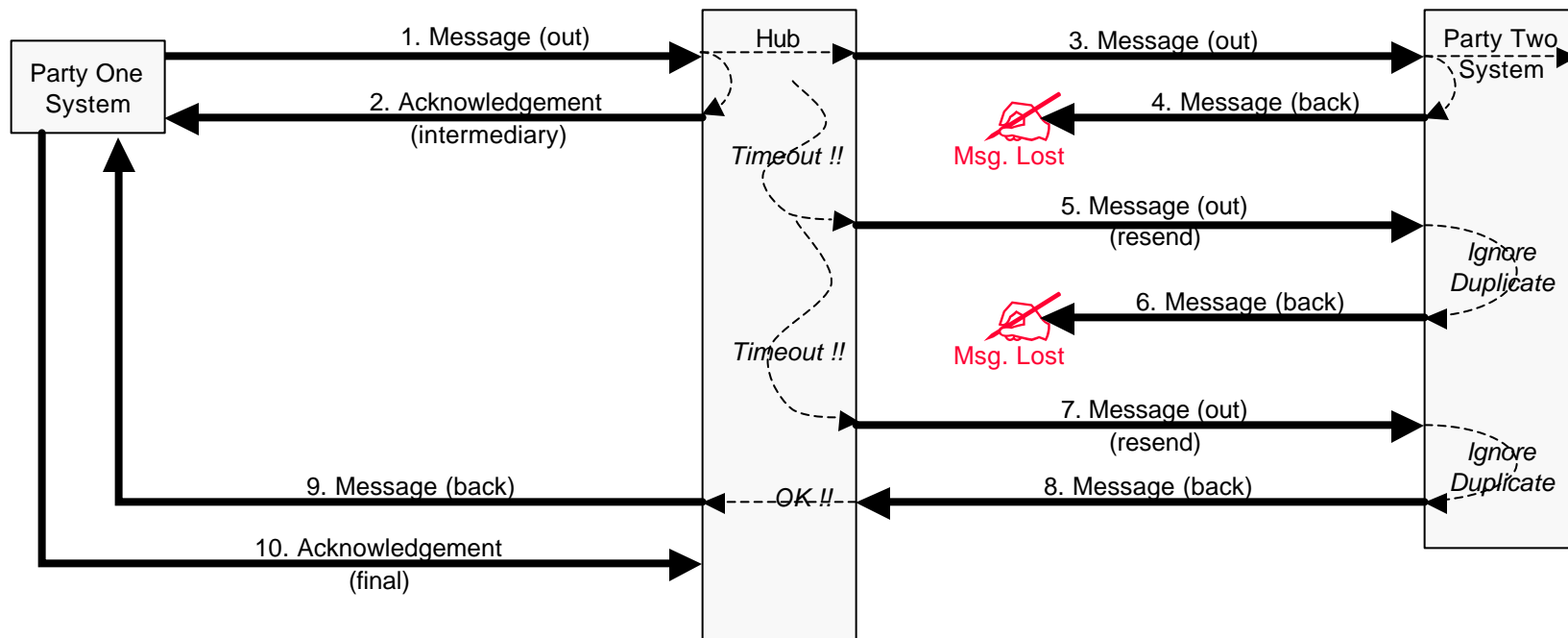
- The Message (back) acts as an implied acknowledgement for the Message (out).
- Party One knows, once the Message (back) has been received that Party Two received the Message (out)



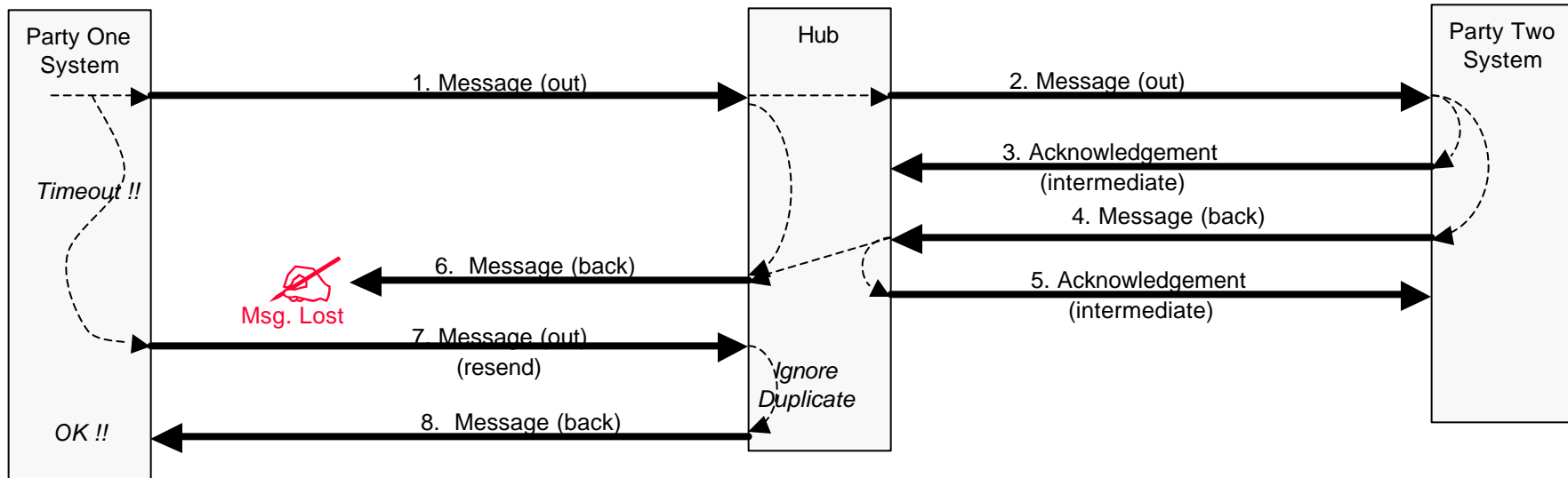
- An intermediate Ack may still be required if, for example, one leg of trip uses a slow protocol such as SMTP
- Message 5 also needs an ack (message 6) if it is to be delivered reliably



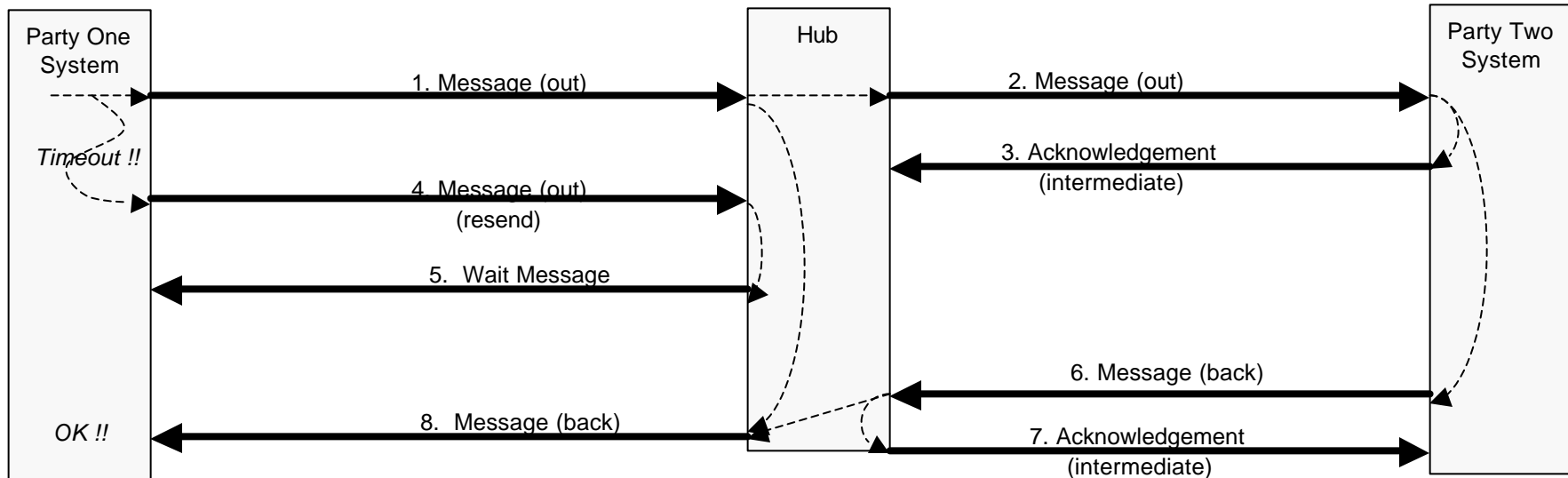
- Implicit and explicit acks may be combined on a separate hops in a multi-hop routing
- Implicit and explicit acks must not be mixed on a single conversation over one hop



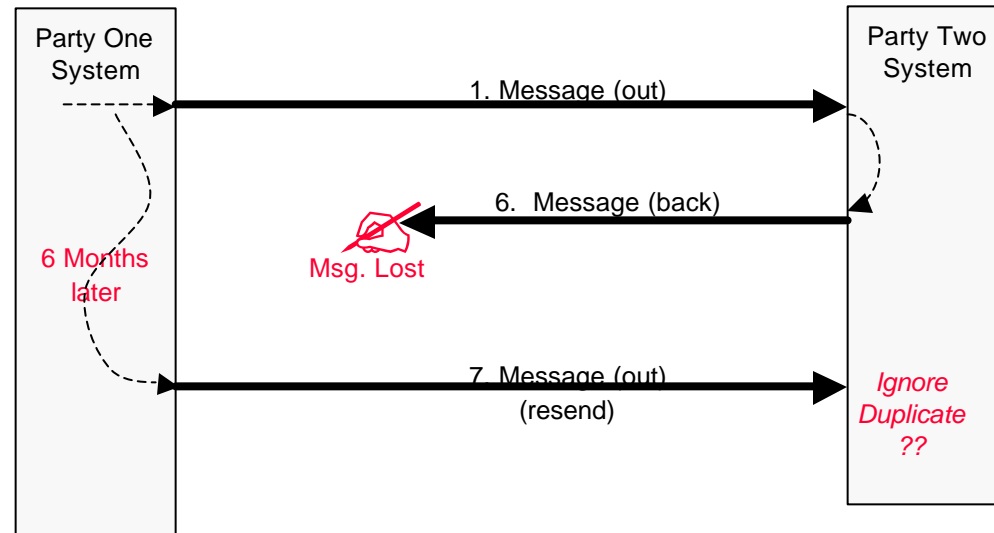
- Message 8 is an implicit Ack of Message 7
- Message 9 also needs an Ack if it must be delivered reliably (message 10)
- *(no new requirements)*



- Intermediate MSHs needs to know that Message 4 from Party Two is implicit ack to Message 1 from Party One. Requires Message 4 to contain “Ref to” to Message 2
- If an intermediate MSH receives a duplicate it should send back the last message it sent



- Wait message tells party one to stop resending and wait for message to arrive
- Same approach applies if Party Two had received a duplicate message before a response could be sent
- *(this is the reason for the “transient error” in the original Error Handling Spec – David Burdett)*



- How long should Party Two retain a message so that it can check for duplicates – 2 hours?, 2 days?, 2 weeks?, 2 months? 2 years????
- Need a concept of a lifetime for a message for duplicate filtering purposes
- Is this set in the message?, in the CPA?, in the CPP?, or any/all of these

Summary Requirements

- Reliable Messaging must work equally well over multiple and single hops (UC 5)
- Each hop may use a different standard/protocol (UC 5)
- The final MSH MAY acknowledge receipt of a message (UC 1)
- An intermediate MSH MAY, if required, acknowledge receipt of a message that it will forward to the MSH that sent it the message (UC 2)
- An intermediate MSH MAY omit an intermediate ack if the final acknowledgement can be returned in time (UC 6)
- An intermediate MSK MAY omit the return of a Final Ack if Delivery Failures are reported
- An intermediate MSH MAY, if required, report the failure to successfully forward a message to the MSH that sent it the message (UC 3)
- An intermediate destination MAY, if required, need to re-send a message it is forwarding before giving up (FA 1)

Summary Requirements - 1

- The “final” acknowledgement message received from the final MSH MAY, if required need to be sent – via any intermediate MSHs back to the MSH that sent the original message (UCs 1 & 5)
- A MSH that receives a delivery failure message MAY, if required, need to be sent, via any intermediate MSHs back to the MSH that sent the original message (FA 2)
- Final Acknowledgement messages and Delivery Failure messages themselves MAY, if required, need to be sent reliably using acks (FAs 2 and 3)
- An Implicit ack MAY replace an Explicit Ack (UC 8)
- Implicit acks and Explicit acks MAY be used on each leg of a multi-hop route (UCs 9 & 10)
- Implicit and explicit acks MUST NOT be mixed on a single conversation over one hop (UCs 9 & 10)

- An Intermediate MSH MUST keep track of messages that were sent as an acknowledgement to an earlier message and send back the latest message sent (UC 11)
- An MSH SHOULD send a Wait Message if a duplicate message is received before the response (either an implicit or explicit ack) is ready to be sent (UC 12)
- Messages MUST have a maximum lifetime for duplicate filtering purposes (UC 13)