An Introduction to the AMQP 1-0 Draft

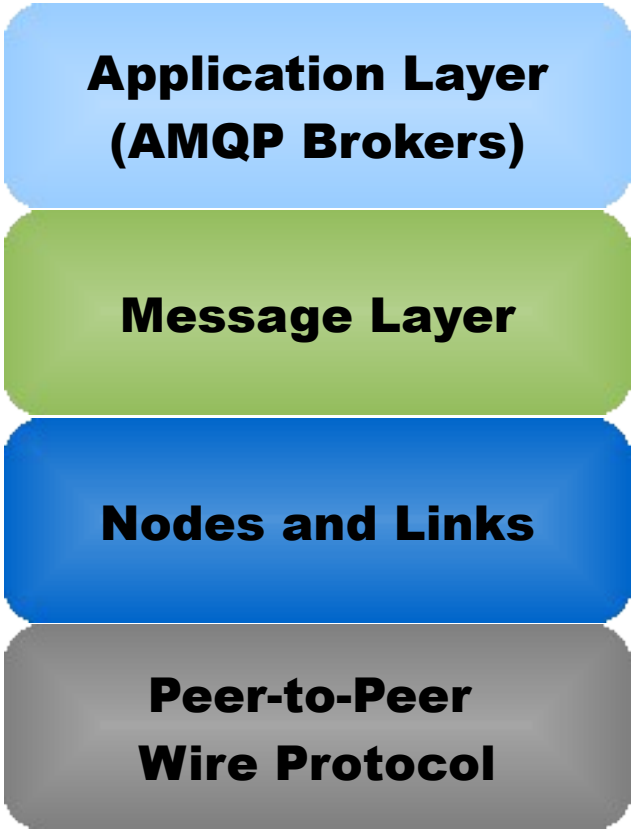
Robert Godfrey, JPMorgan

Path to AMQP 1-0

AMQP 0-10 met reliability requirements, but complex.

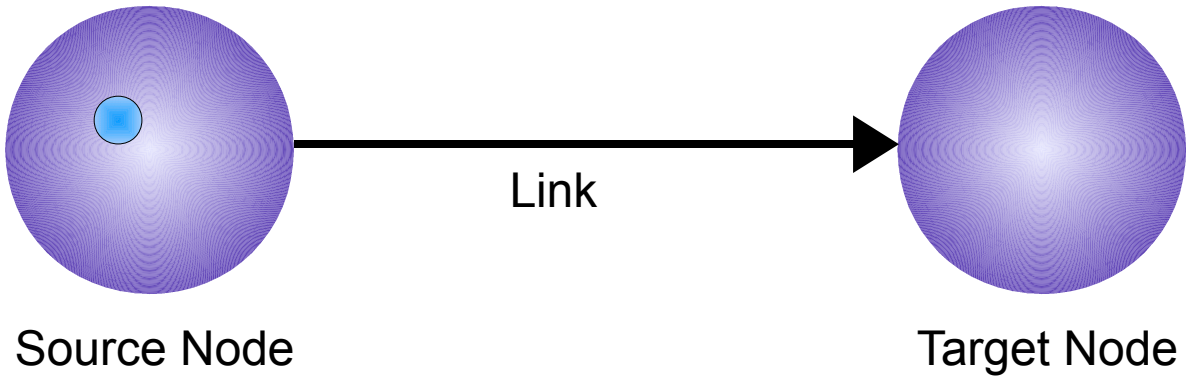
- Simplify wire protocol
- Address the remaining business requirements
  - Message Security
  - Global Addressing
- Disentangle Management activities from base messaging functionality
- Create a model more easy to retro-fit to legacy brokers
- Extensible layered protocol

The AMQP 1-0 Stack



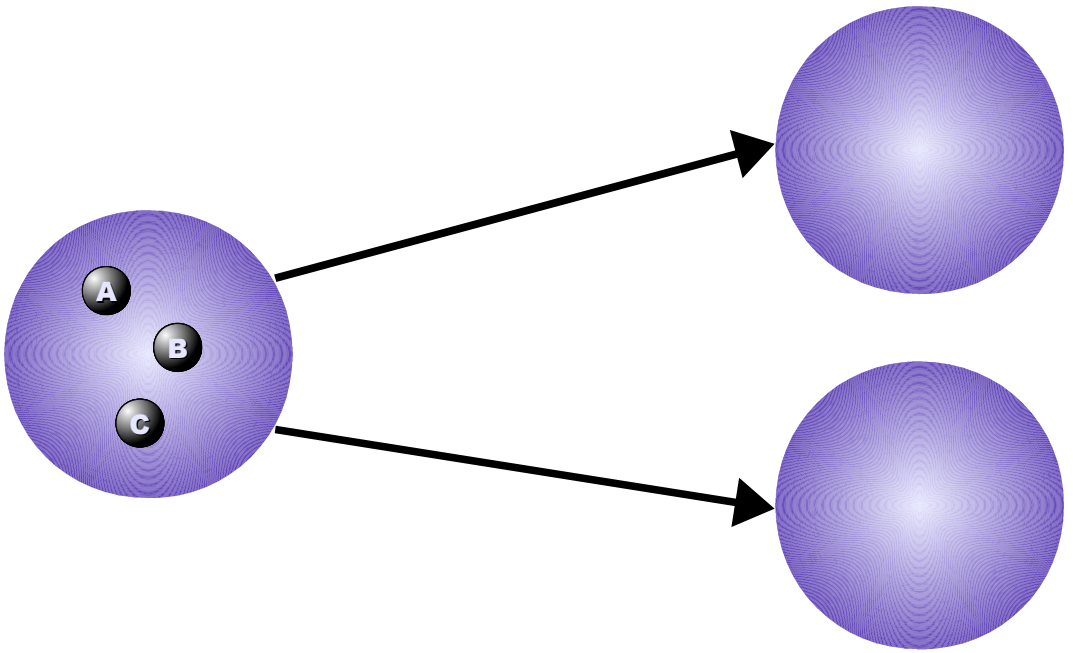
The AMQP Network

- An AMQP Network consists of **Nodes** and **Links**.
- A **Node** is a named source and/or sink of **Messages**.
- **Messages** travel between **Nodes** along named, unidirectional **Links.**



Types of Links

- **Destructive:** the transfer along the link removes the message from the source
- **Non- Destructive:** the message remains at the source node, and is “copied” to the destination.



Messages

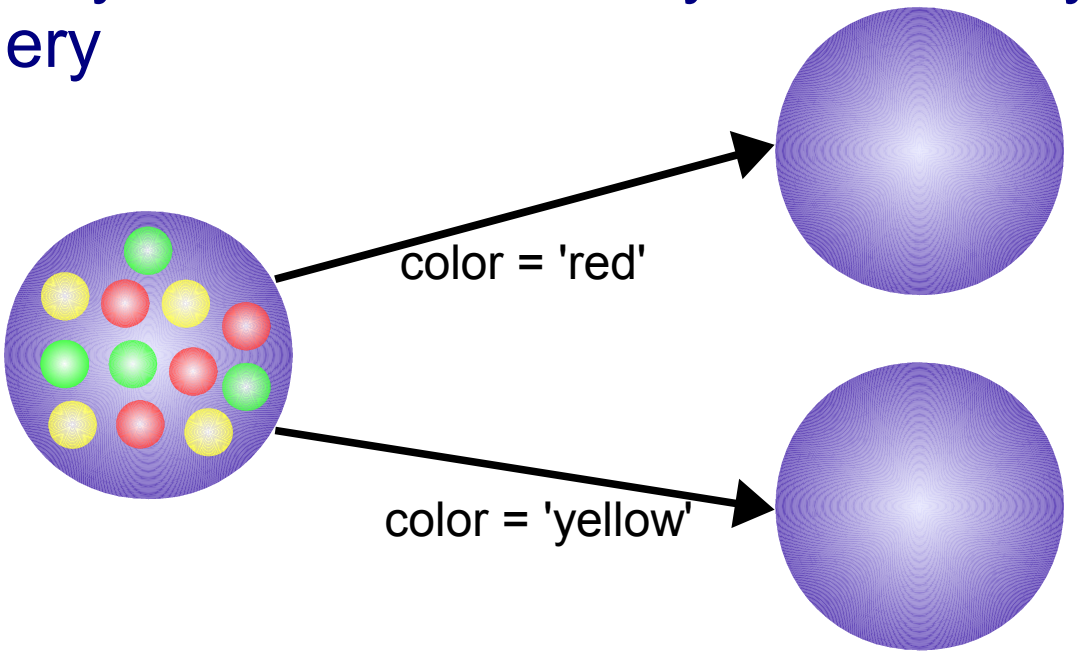
- Messages consist of parseable **Properties** and an opaque **Body**.
- Messages may not be mutated by the AMQP Network.
- The network carries information about the Message in **Headers** and **Footers**.
- Header and Footer values may be modified in the Network.

Message Identity

- A Message is assigned a globally unique identifier.
- Nodes which perform transformations are creating new Messages, with new ids.
- Only one “copy” of a Message can ever exist at a Node.

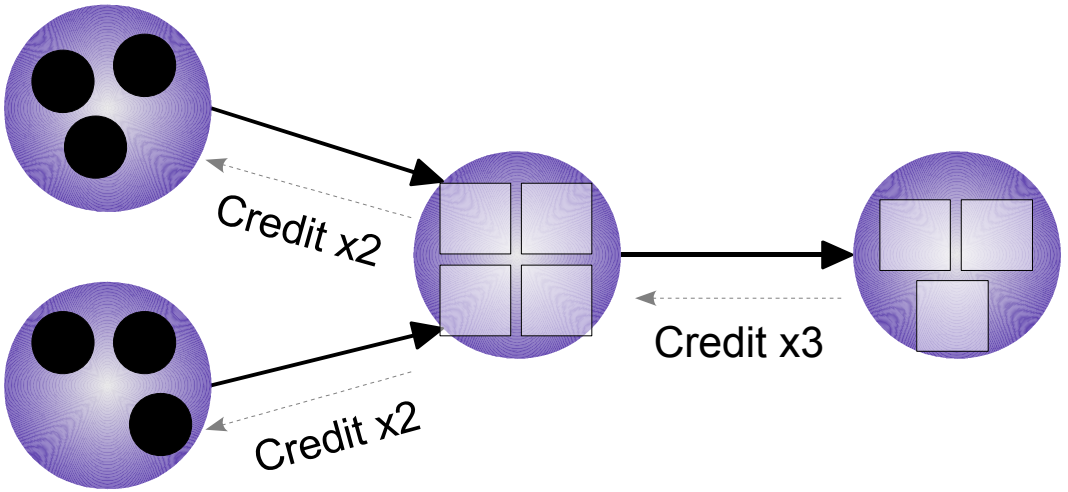
Filters

- Each **Link** may have a **Filter** which evaluates which messages may travel along it.
- **Filters** are evaluated against immutable properties of the Message.
- **Filter** syntax determined by the Filter Type, e.g. SQL, XQuery

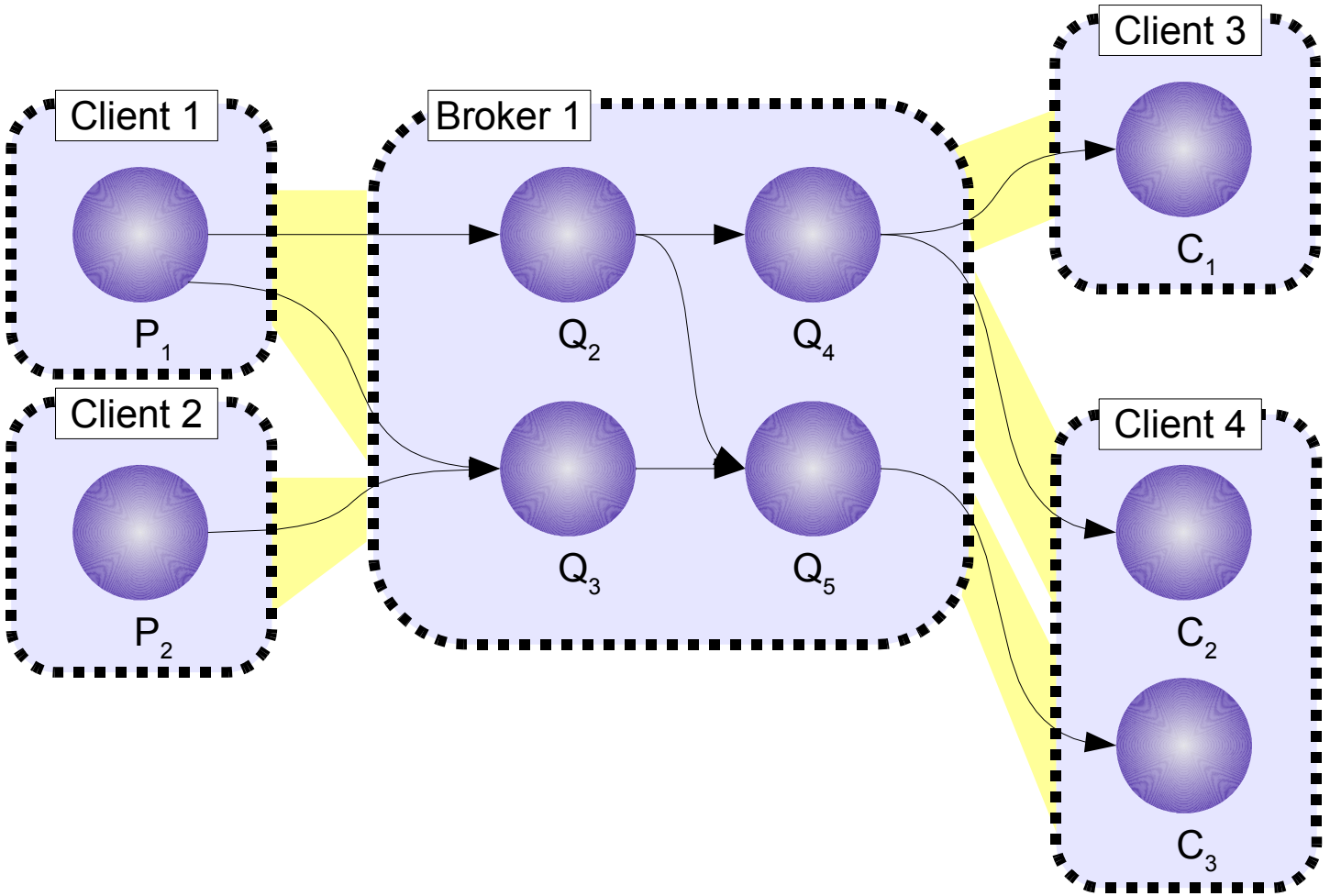


Credit

- A message may only pass along a given link if the destination node has issued **credit** to the link.



Containers


Client 1
Broker 1
Client 3
P_1
Q_2
Q_4
C_1
Client 2
Client 4
Q_3
Q_5
P_2
C_2
C_3

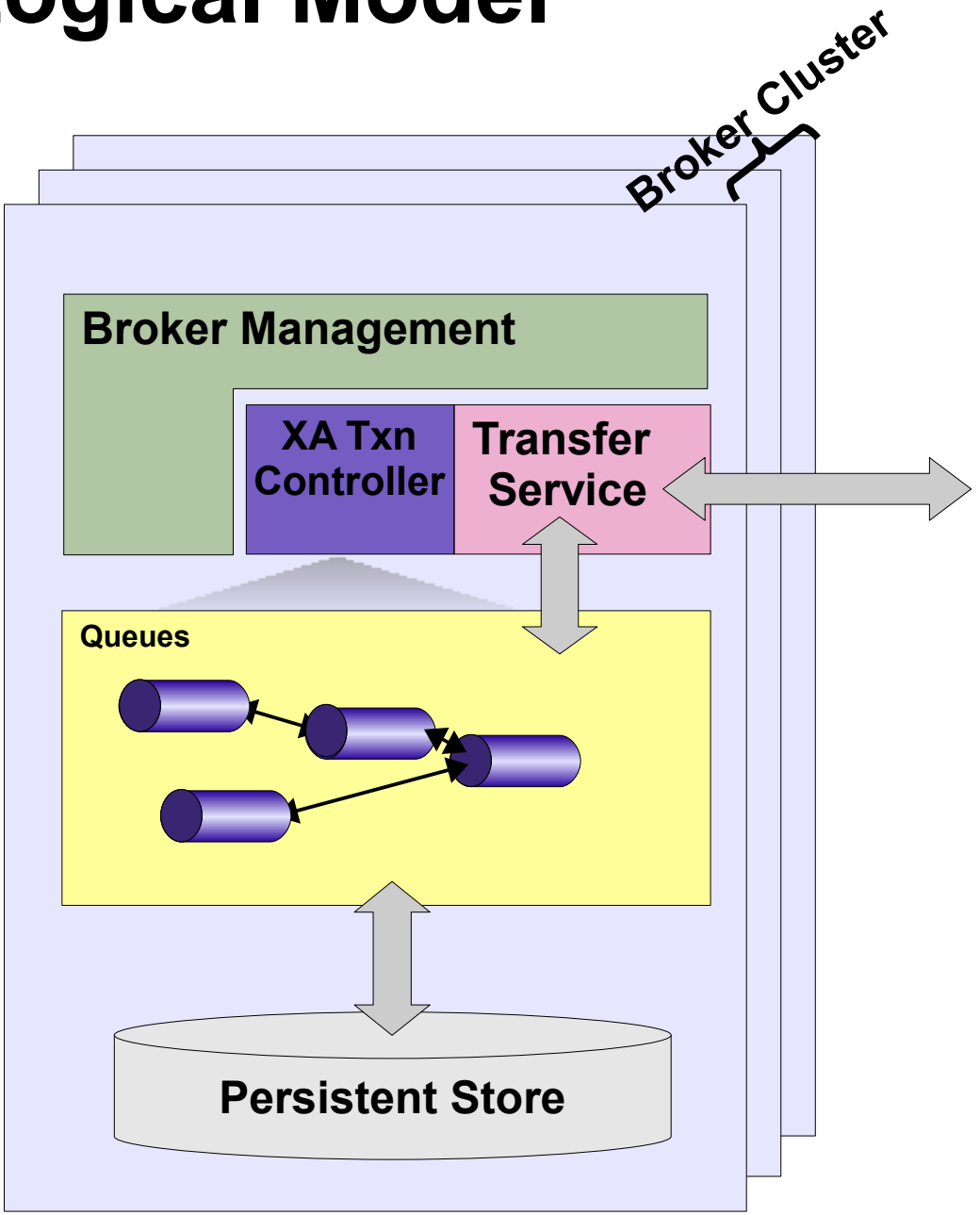
Containers

- **Containers** contain nodes
- Containers have a globally unique name
- Within a container a node name will resolve to at most one node
- Authentication is always with respect to a container
- Observable Container state is consistent

AMQP Broker

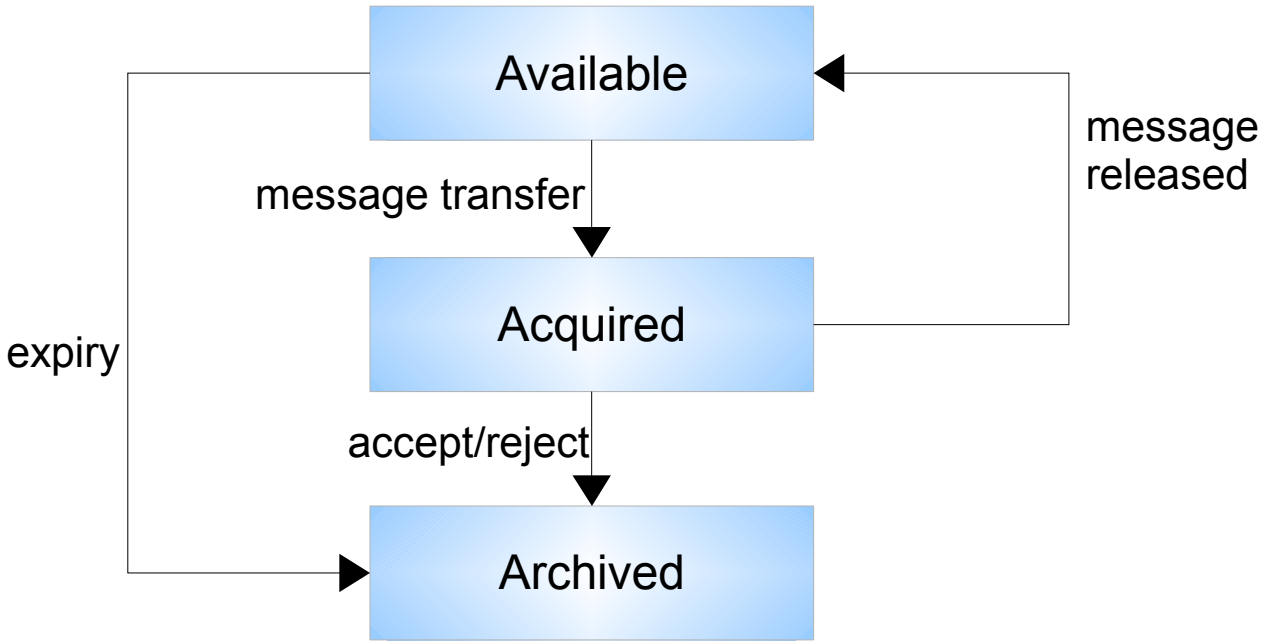
- A **Broker** is a type of **Container** which
  - Allows other containers to establish ingoing and/or outgoing links to **Queues** it contains
  - Allows the construction of “internal” links between **Queues** it contains
  - Authenticates each and every session established with it
  - Provides AMQP services for
    - Managing its Queues and internal Links
    - Transferring Messages to remote addresses
  - Can act as a Transactional Resource

Broker Logical Model

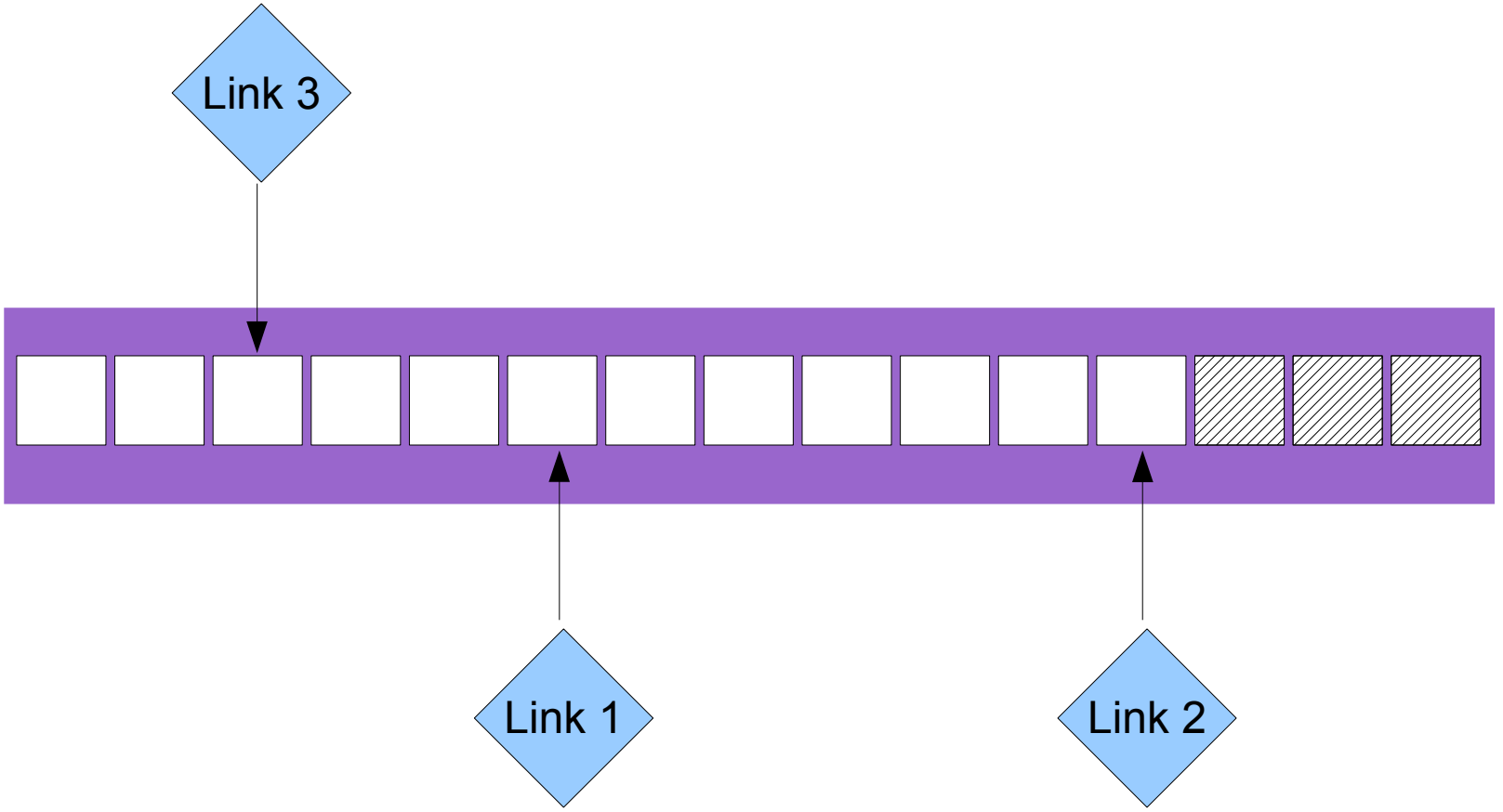


Queues

- **Queues** are a type of **Node** providing
  - Limitted ordering guarantees
  - Durability
  - Well-Defined Message state transition

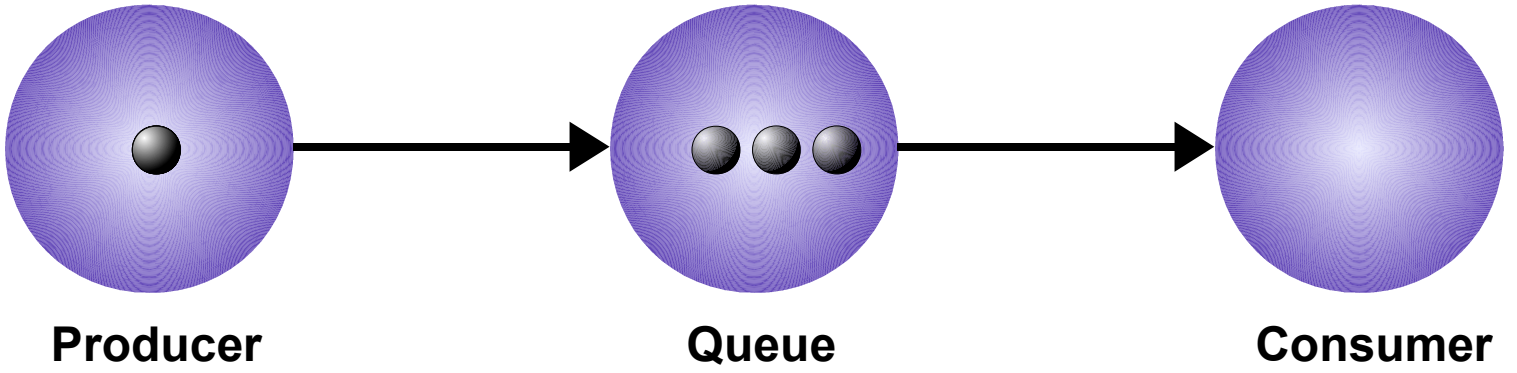


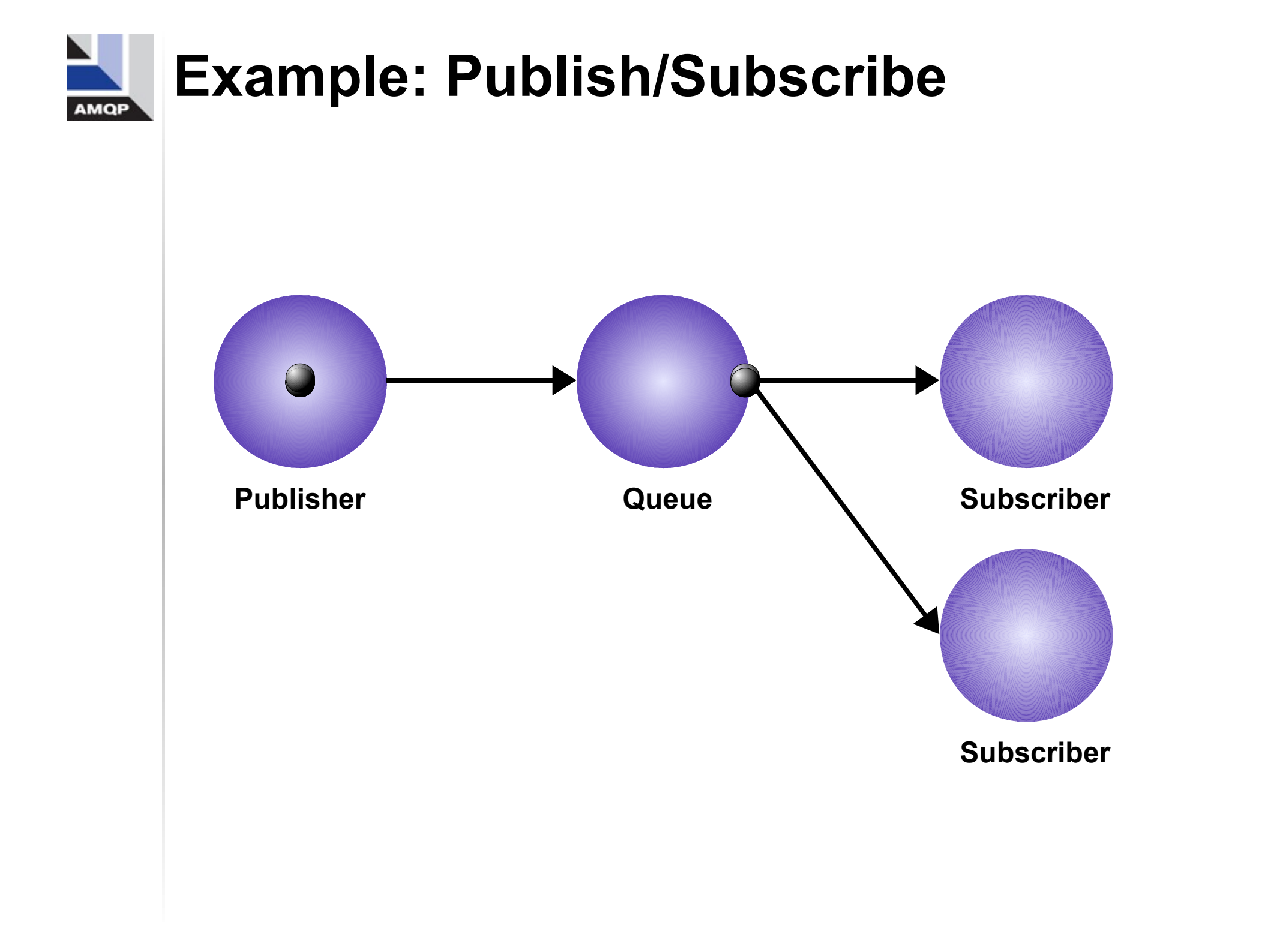
Queues



- Messages join the Queue at the tail
- Each Link maintains its own “head” pointer
- Links with filters will skip over some entries

Example: Point to Point Messaging

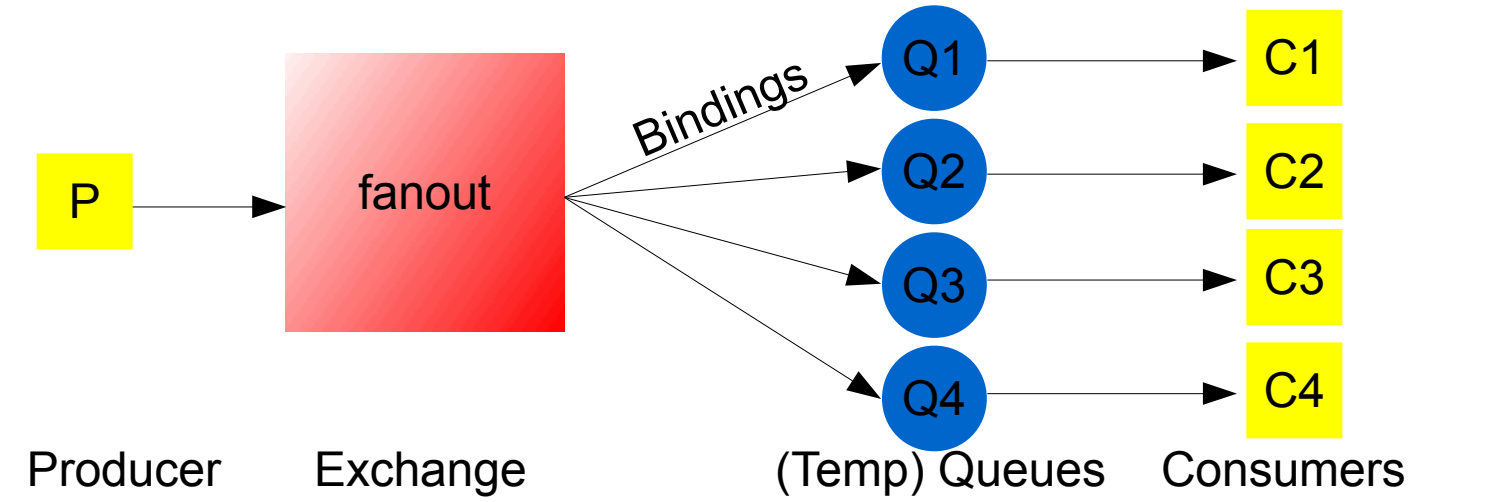


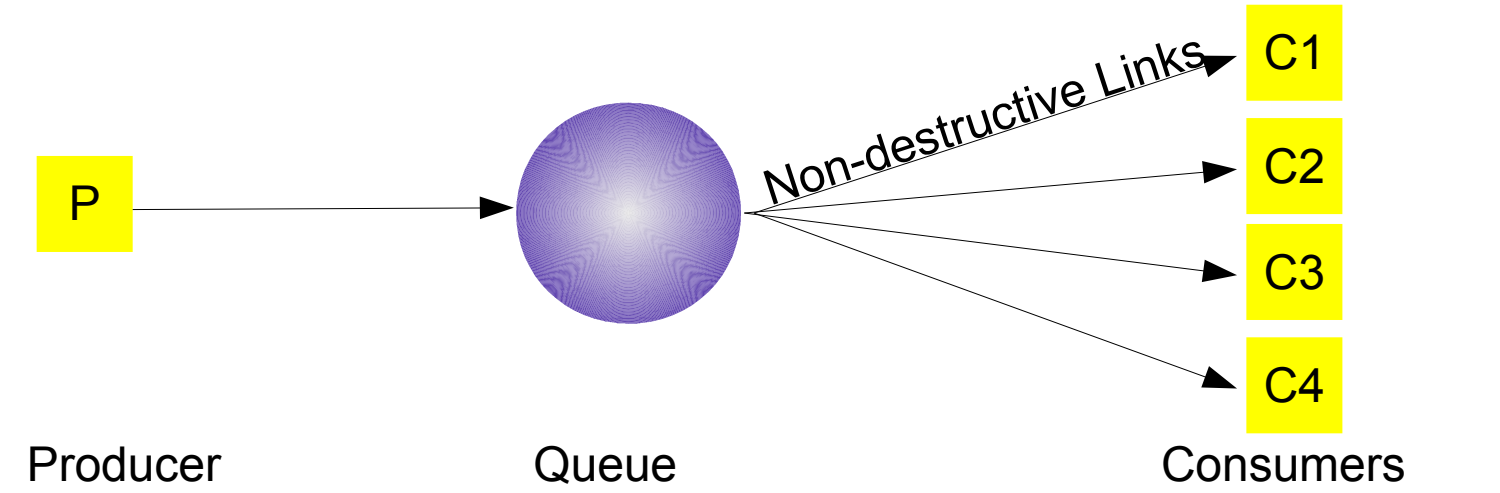

AMQP
Example: Publish/Subscribe
Publisher
Queue
Subscriber
Subscriber

Emulating Exchanges

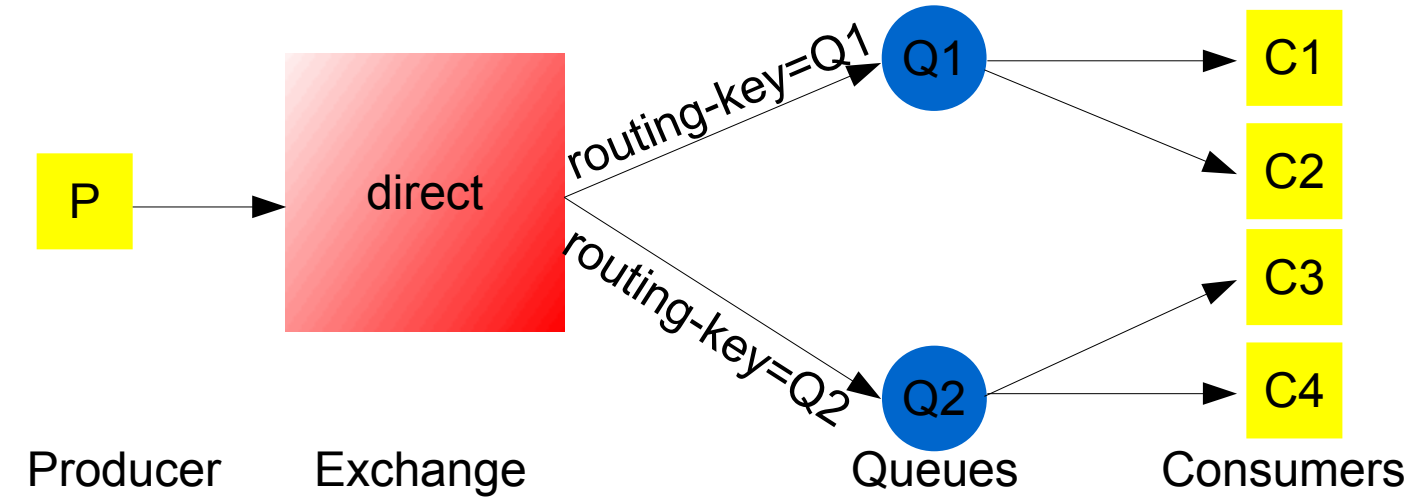
- Previous versions of AMQP defined **Exchanges**
- An **Exchange** defined a routing algortithm
- **Exchange** functionality can be emulated with **Nodes** and **Links**

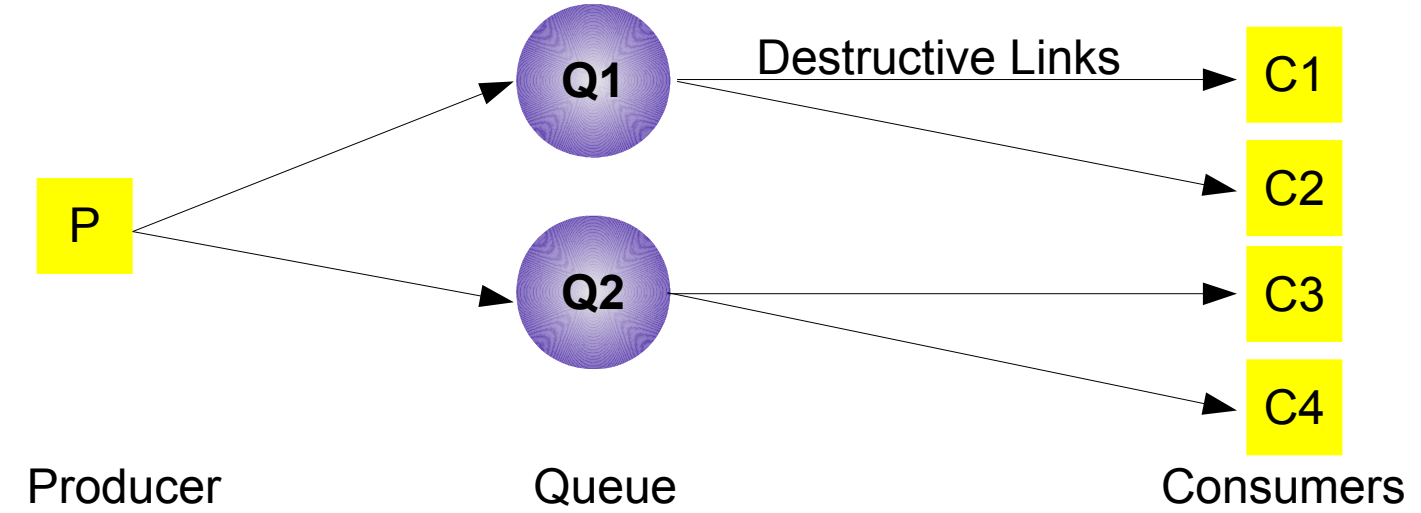
Fanout Exchange



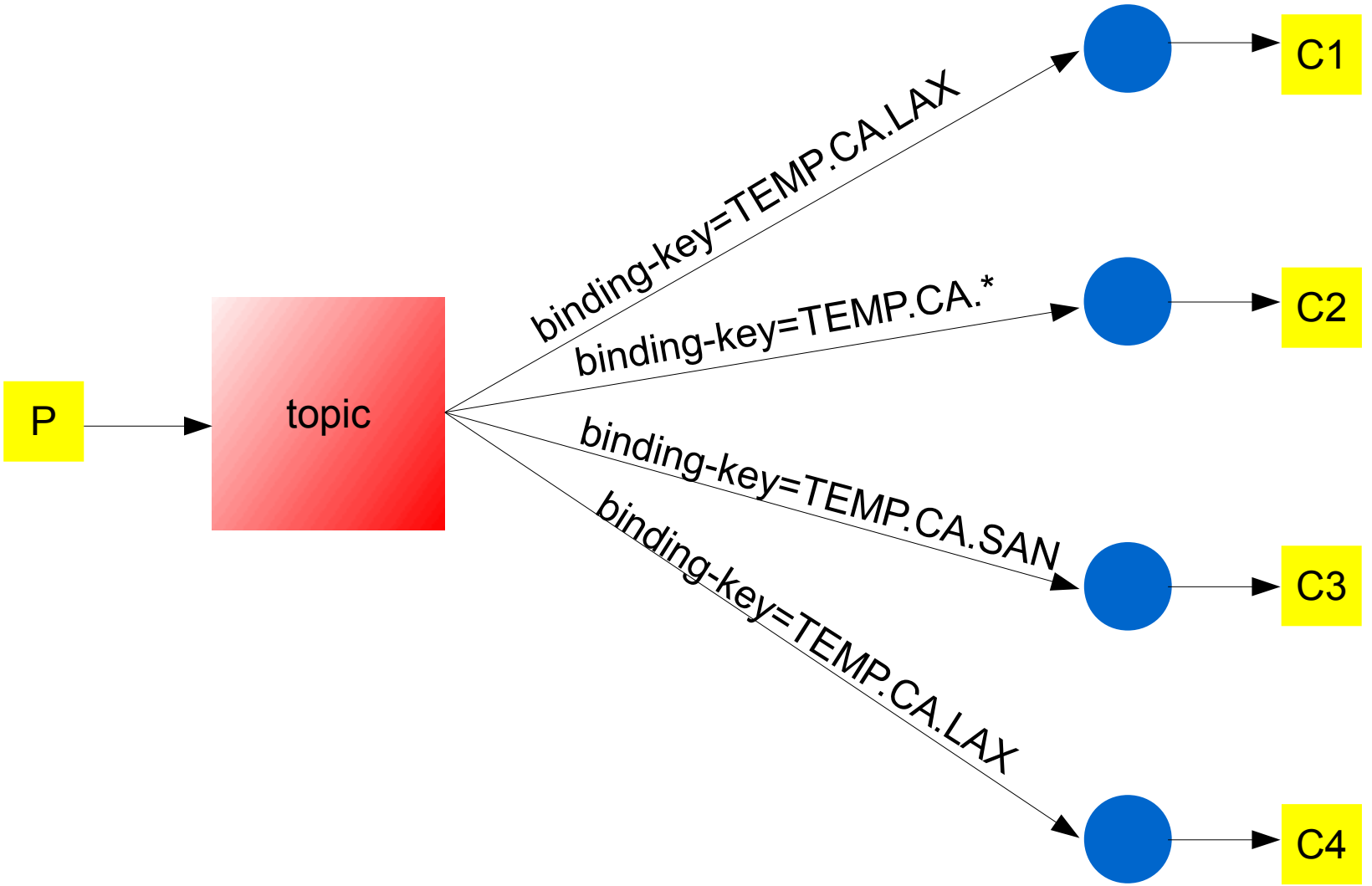


Direct Exchange

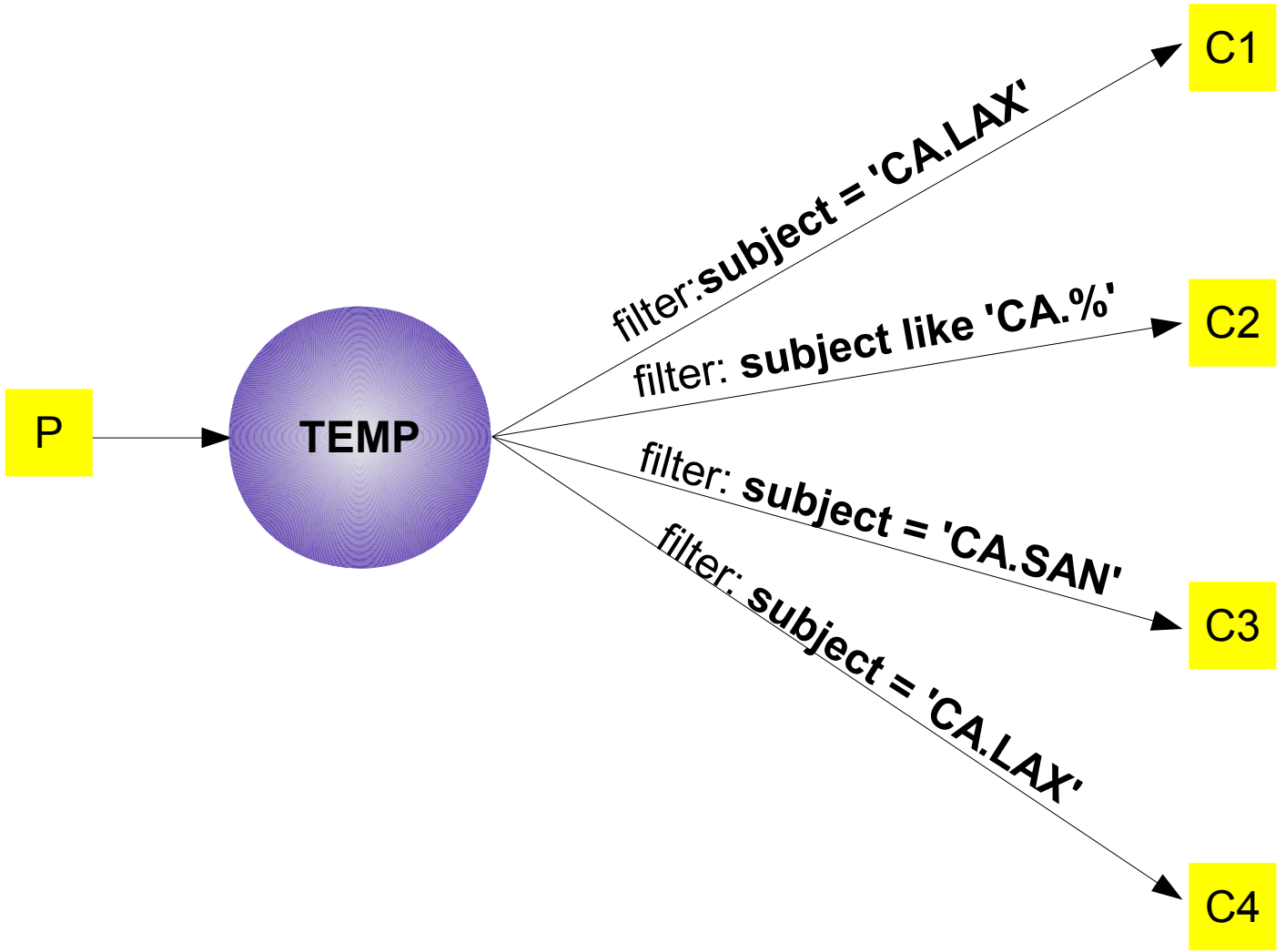




Topic Exchange



Topic Exchange



Services

- A **Service** can be thought of as an application running inside the **Broker**
- Interact with Services by sending Messages
- Each Service exposes at least one Node Name
- Service Nodes *are* ***not*** Queues
- Brokers provide services for
  - Inter-broker transfer
  - Broker Management
  - (Option) Distributed Transaction Coordination

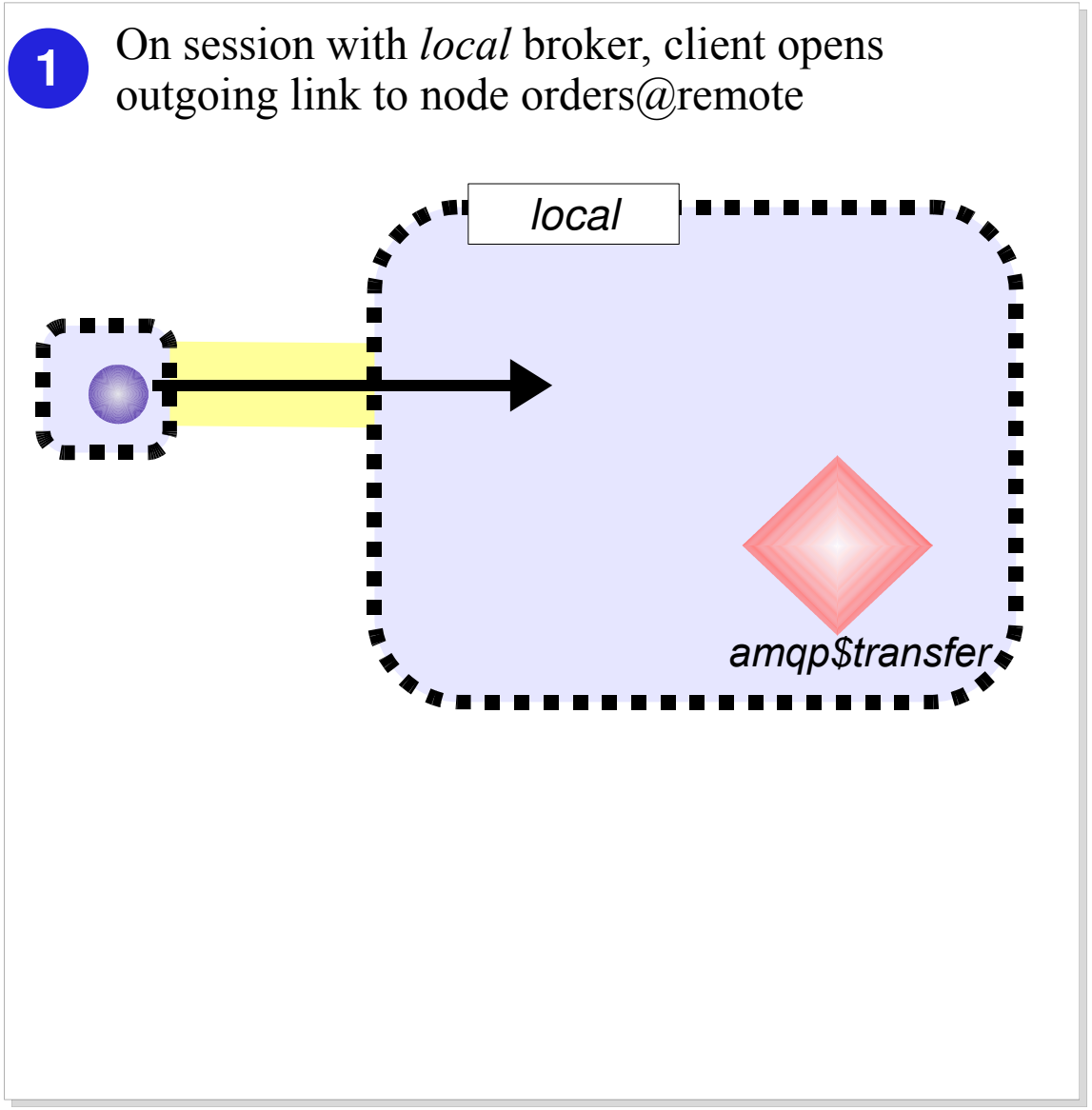
Global Addressing

- Standard scheme for sending messages across departmental and organisational boundaries
- e-mail: local-part@domain
  AMQP: node-name@container
- Container can be expressed as network name/address
- Address is set as a **Message Property** "`to`"
- Can use "`reply-to`" to direct replies to a local response queue

AMQP Transfer Service

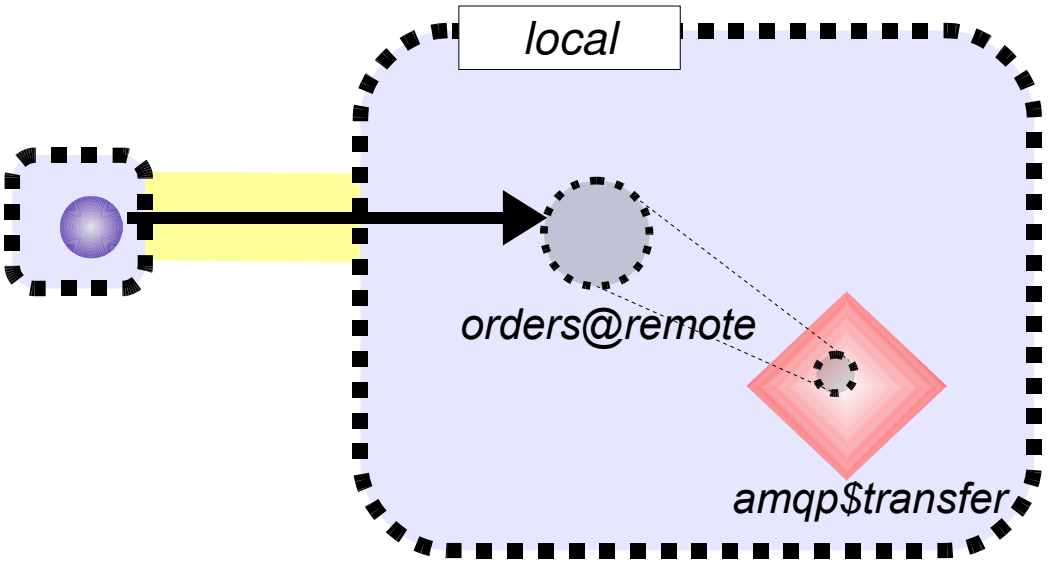
- Messages for global addresses sent to a transfer service in the local broker – node name *amqp$transfer*.
- Service forwards arriving message either to local node, or via authenticated links to another broker.
- For convenience all node names of form *x@y* aliased to *amqp$transfer*.

Global Addressing Example



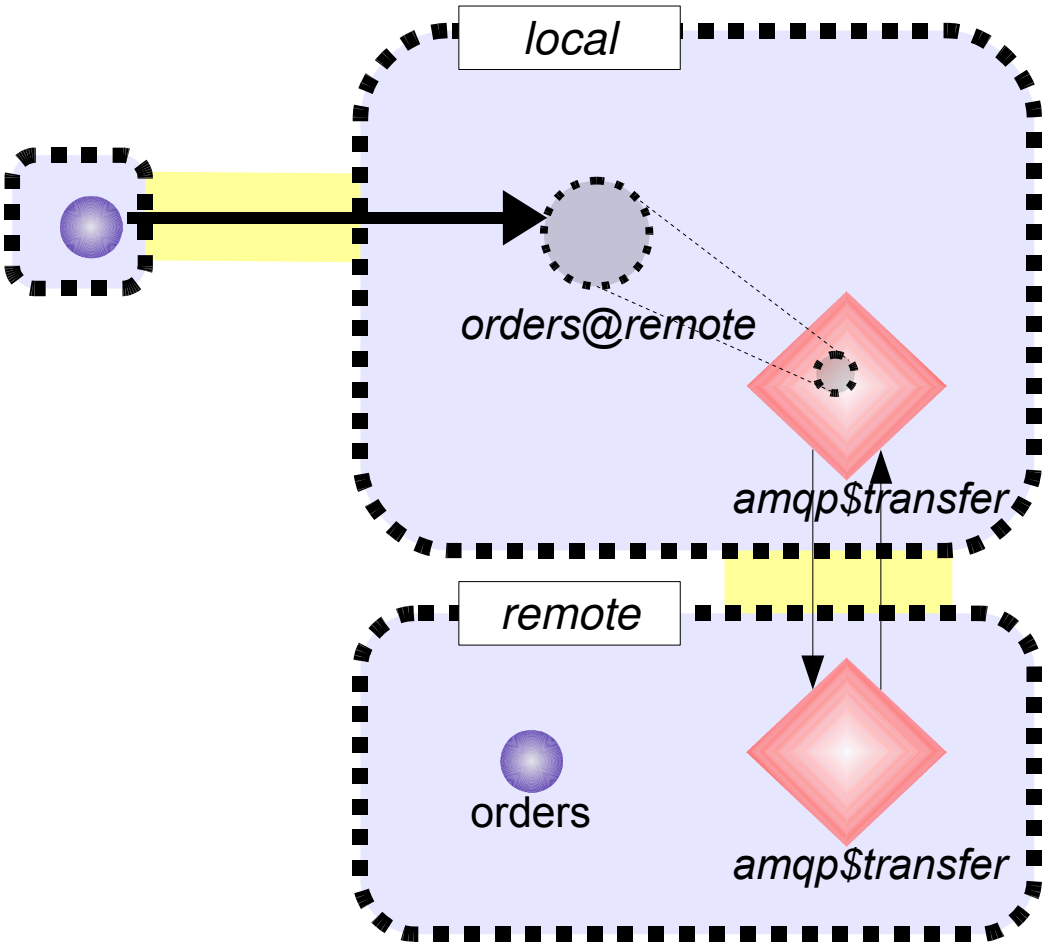
Global Addressing Example

2 The *local* broker interprets orders@remote as an alias for *amqp$transfer* with the restriction that incoming messages have “to” property of “orders@remote”



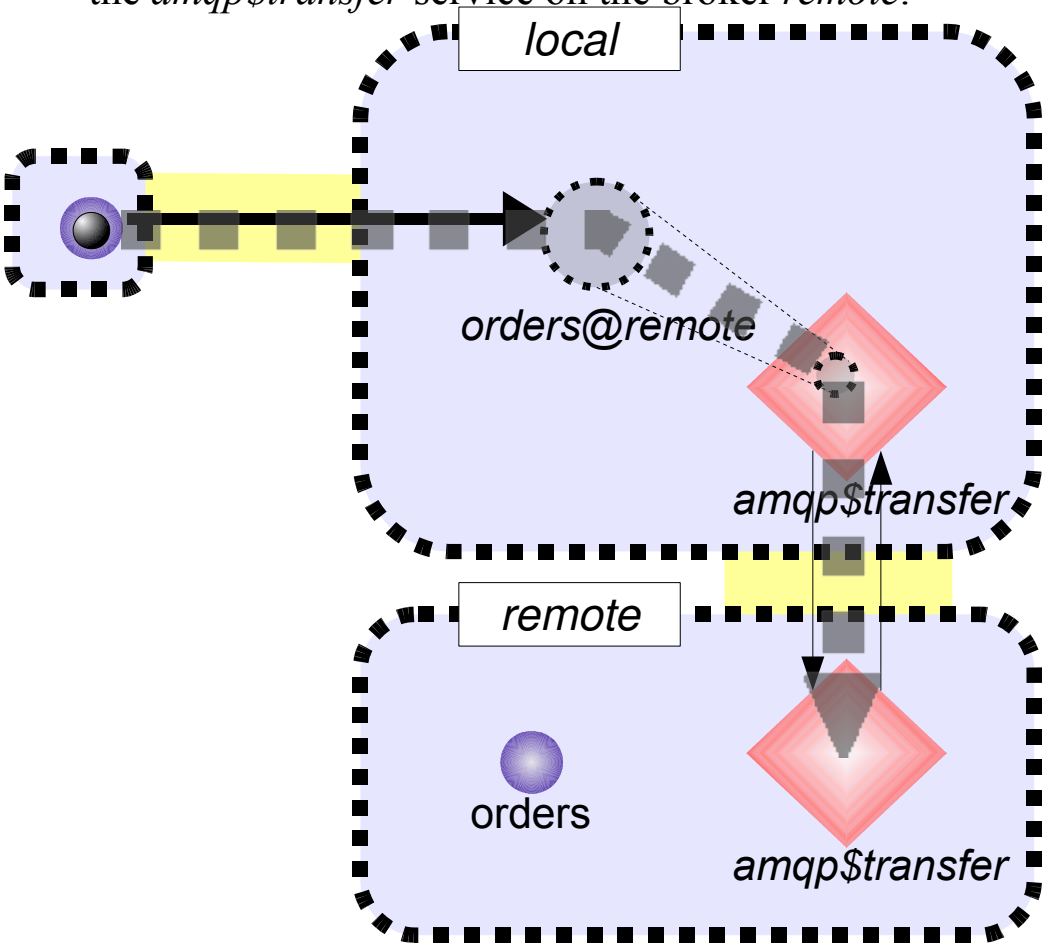
Global Addressing Example

3 The broker *local* looks up the connection information for broker *remote*, and ensures it has a transfer session with link to the remote *amqp$transfer*.



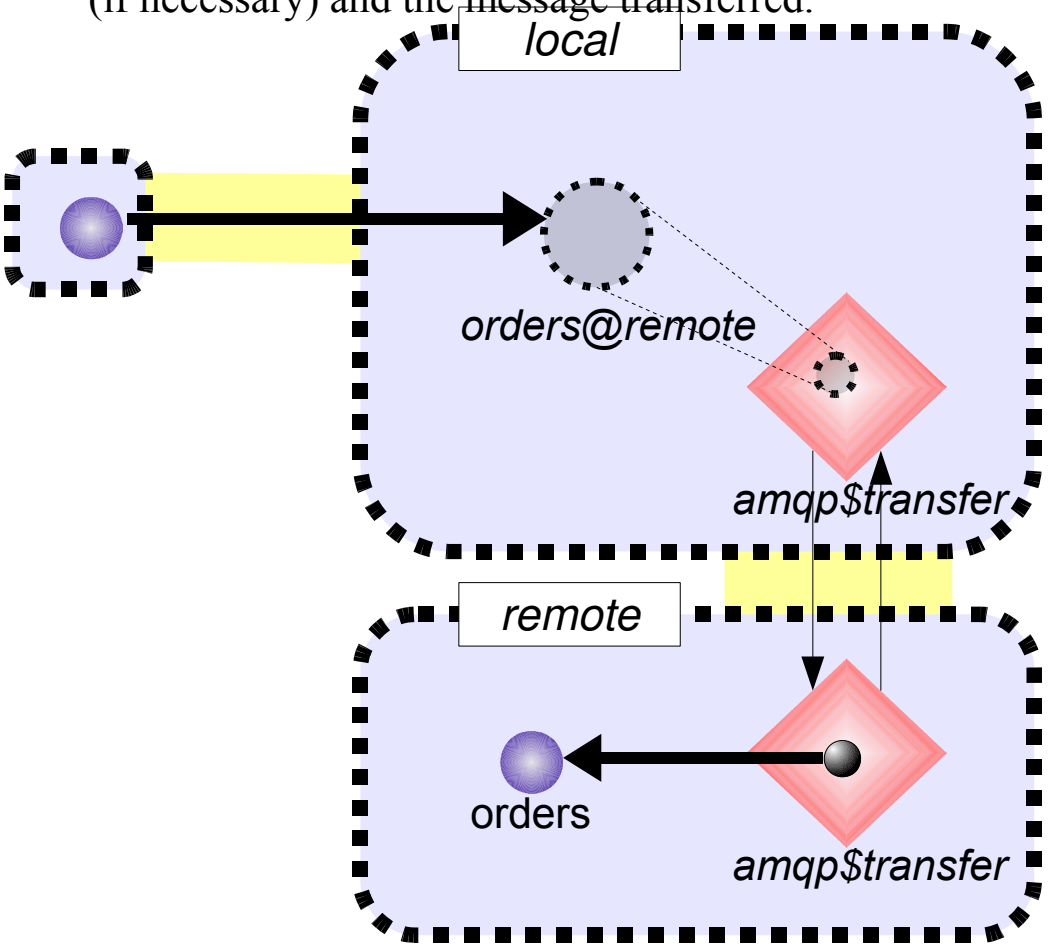
Global Addressing Example

4 The Client sends a message with the "to" property set to "orders@remote" along the link to the broker *local*. The *local* broker *amqp$transfer* service forwards it to the *amqp$transfer* service on the broker *remote*.

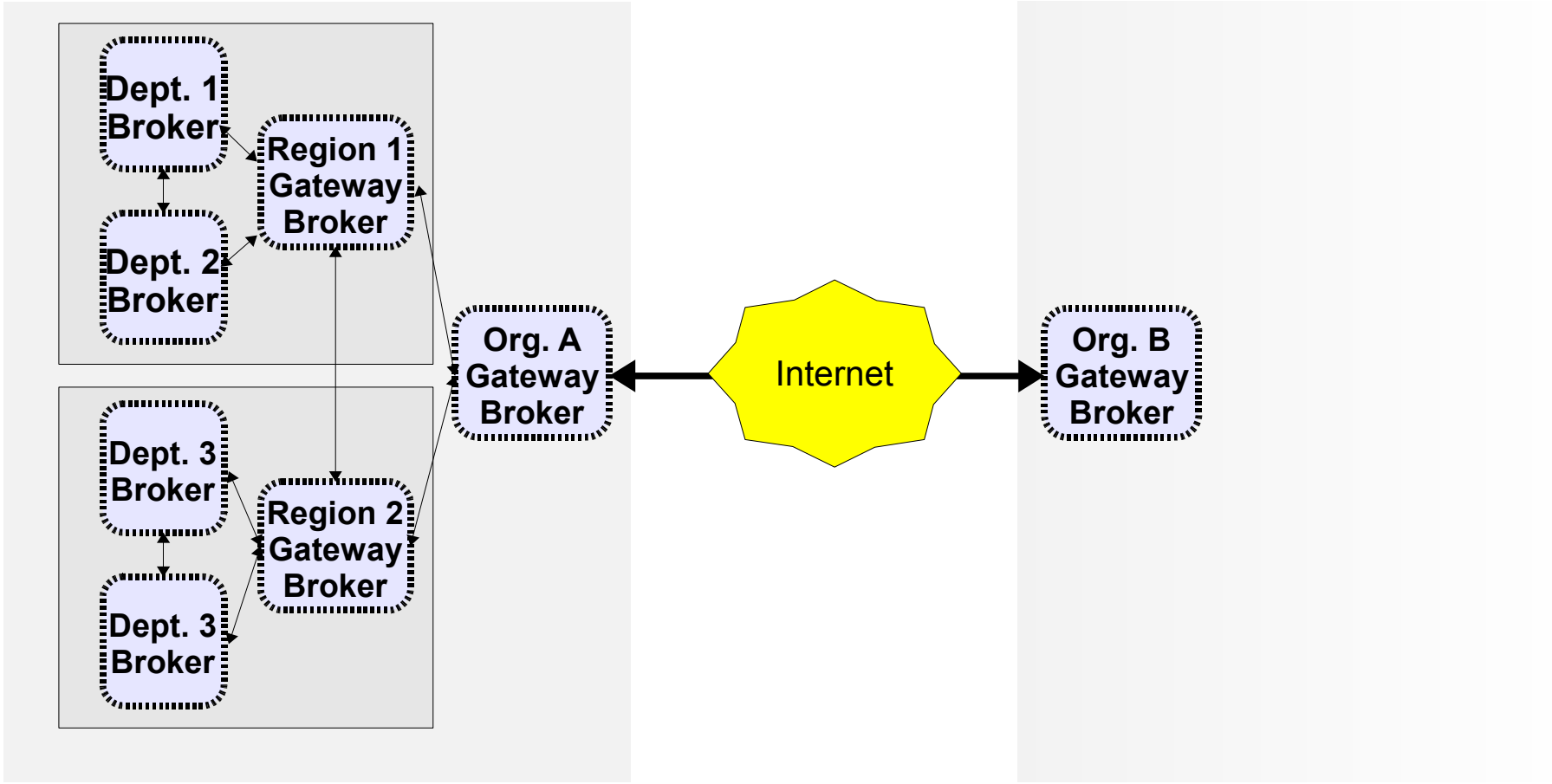


Global Addressing Example

5 The *amqp$transfer* service on the broker *remote* inspects the "to" property of the message and discovers that it should be routed to the "orders" node. A link is established (if necessary) and the message transferred.



Global Addressing



Management

- Broker Management includes
  - Creating, Updating, and Deleting Queues
  - Creating, Updating, and Deleting Internal Links
  - Setting Threshold Alerts
  - Listening for broadcast events
  - Dealing with erroneous or poisined messages

Management Protocol

- Broker Management commands sent *over* AMQP – not part of the wire protocol.
- Broker has service node *amqp$management*
- Messages containing management commands
- Batched (atomic) but not transactional
- Management can be federated

Questions

?