Switch or Broker?

A comparison of two models for Reliable Messaging

by Pieter Hintjens, iMatix Corporation

January, 2006

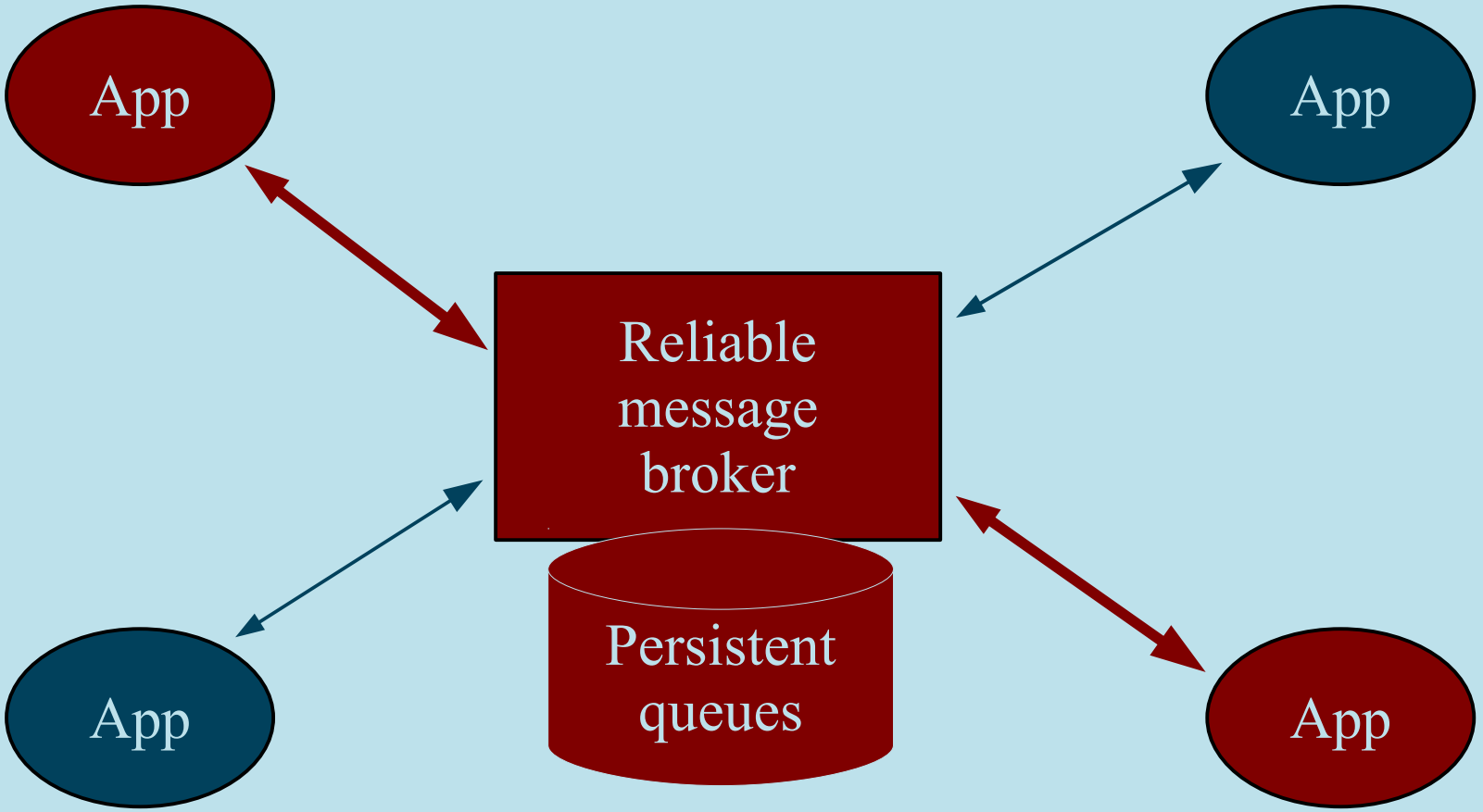
iMatix Corporation

The Classic View

- The "reliable message broker"
- Big, powerful message broker
- Uses high-availability and transactions
- Queues are durable, persistent, heavy
- Consistent with traditional world view
  - Strong centralization
  - One place to look for all data
  - One server to configure, administer, etc.

iMatix Corporation

Reliable message broker



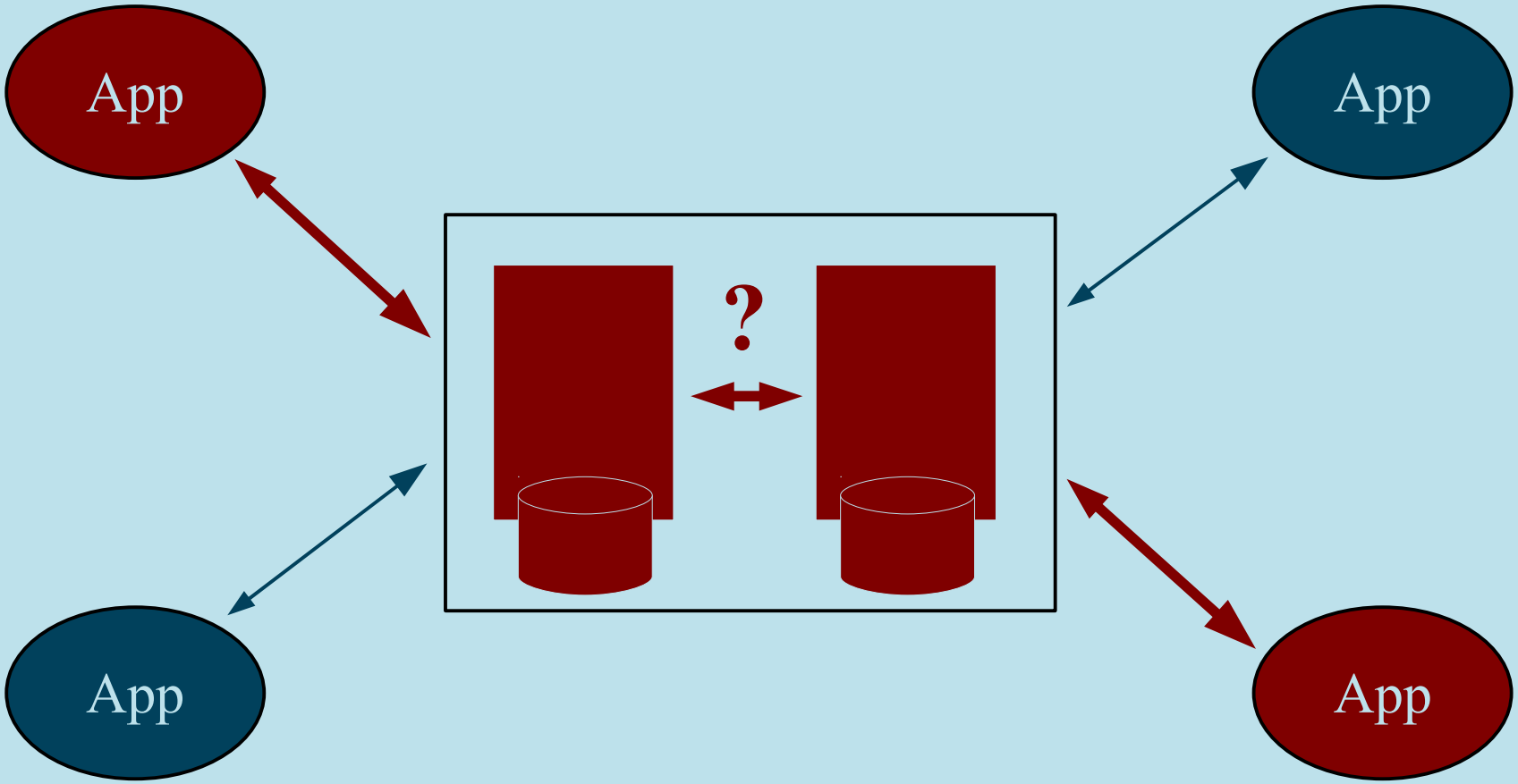
iMatix Corporation

What do we get?

- Point-to-point reliability
  - When we have successfully passed a message to the broker, we can be sure it will be delivered to the recipient.
  - *If there is a recipient...*
- Pretty complex
  - Pedantic message delivery
  - Distributed transaction processing
  - Performance vs. persistence trade-offs

iMatix Corporation

High-availability cluster



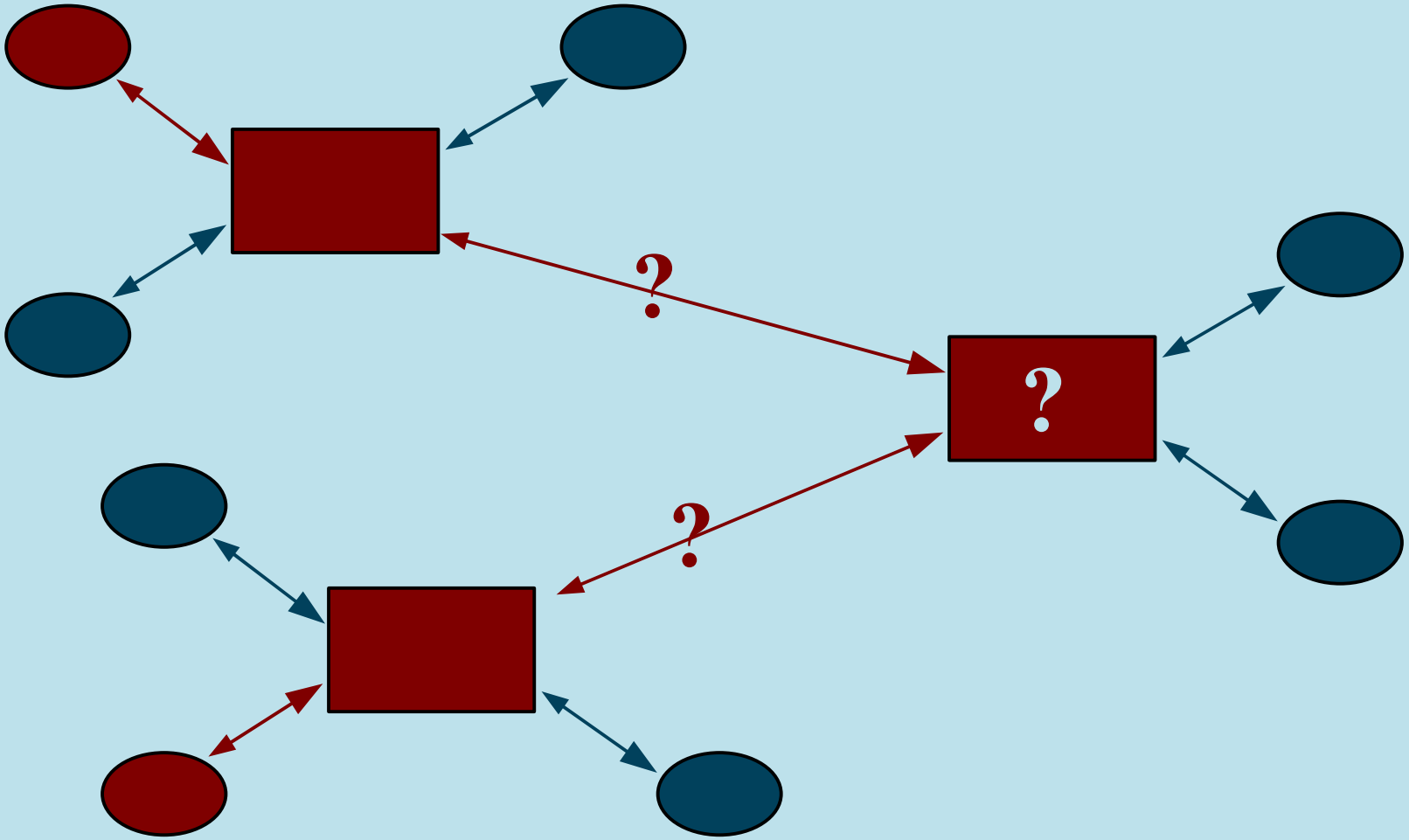
iMatix Corporation

HA reliability challenges

- Guarantee reliability during failover
- Coordination between HA peers
- Transactions between HA peers
- A problem area for most products
- Presumably a solvable problem...
  - But even MQ Series occasionally drops messages
- Broker is now *very complex*
  - ...and HA is visible in the protocol

iMatix Corporation

What about wide-area networks?



iMatix Corporation

WAN reliability challenges

- Only as reliable as weakest link
- Excludes use of untrusted middlemen
- Excludes use of thin "edge" brokers
- Contrary to modern network design
  - Requires few, expensive boxes
  - Does not scale by adding hardware
  - Reliability through complexity?

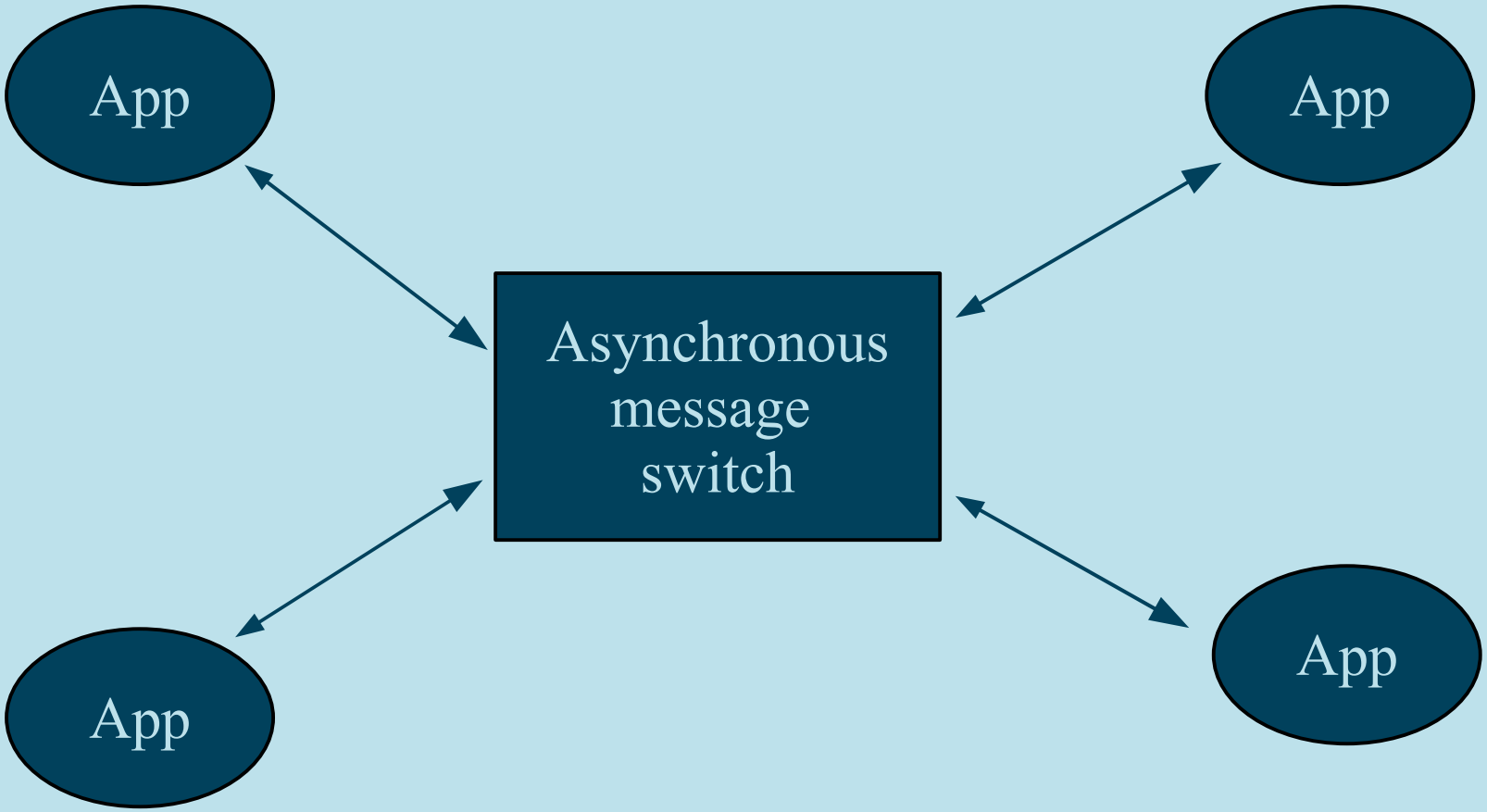
iMatix Corporation

The iMatix View

- The "asynchronous message switch"
- Cheap, disposable, stateless devices
- Organized into high-availability pairs
- Queues are transient and memory-based
- Consistent with modern world view
  - Cheap and simple
  - No data to look for
  - Minimal configuration and administration

iMatix Corporation

Asynchronous message switch



iMatix Corporation

What do we get?

- No reliability in protocol or server
  - Messages can be lost en-route
- Very simple
  - Trivial message delivery
  - No transactions
  - Fastest possible performance
  - Message loss is *very* rare
- But...
  - Of course, we need full reliability

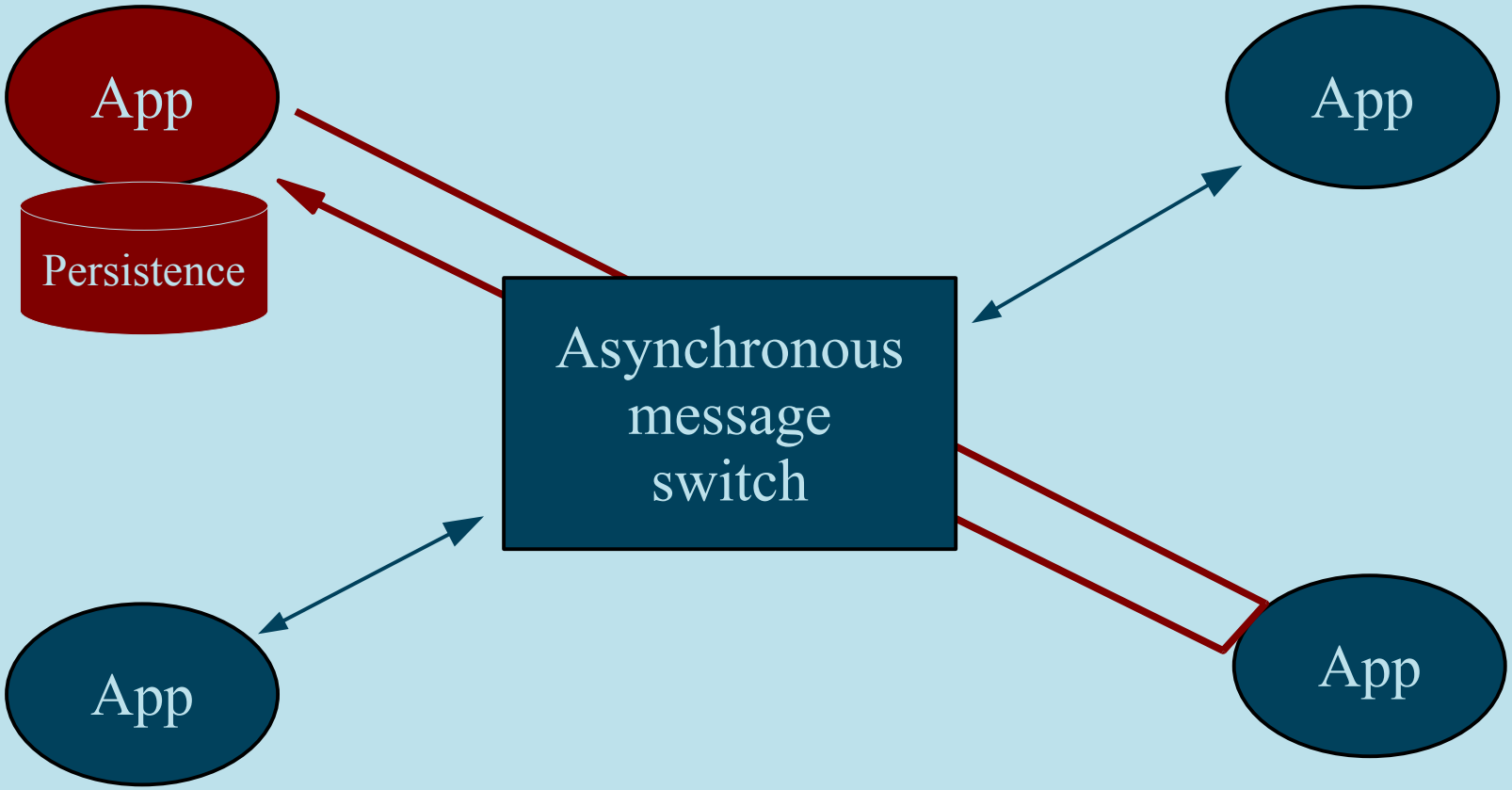
iMatix Corporation

Reliability over AMS (RAMS)

- Two main messaging scenarios
  - Data distribution (publish-subscribe)
  - Service requests (request-response)
- Publish-subscribe can be unreliable
  - Data can be resent periodically
- Request-response should be reliable
  - This is the classic MQ scenario

iMatix Corporation

RAMS Schema



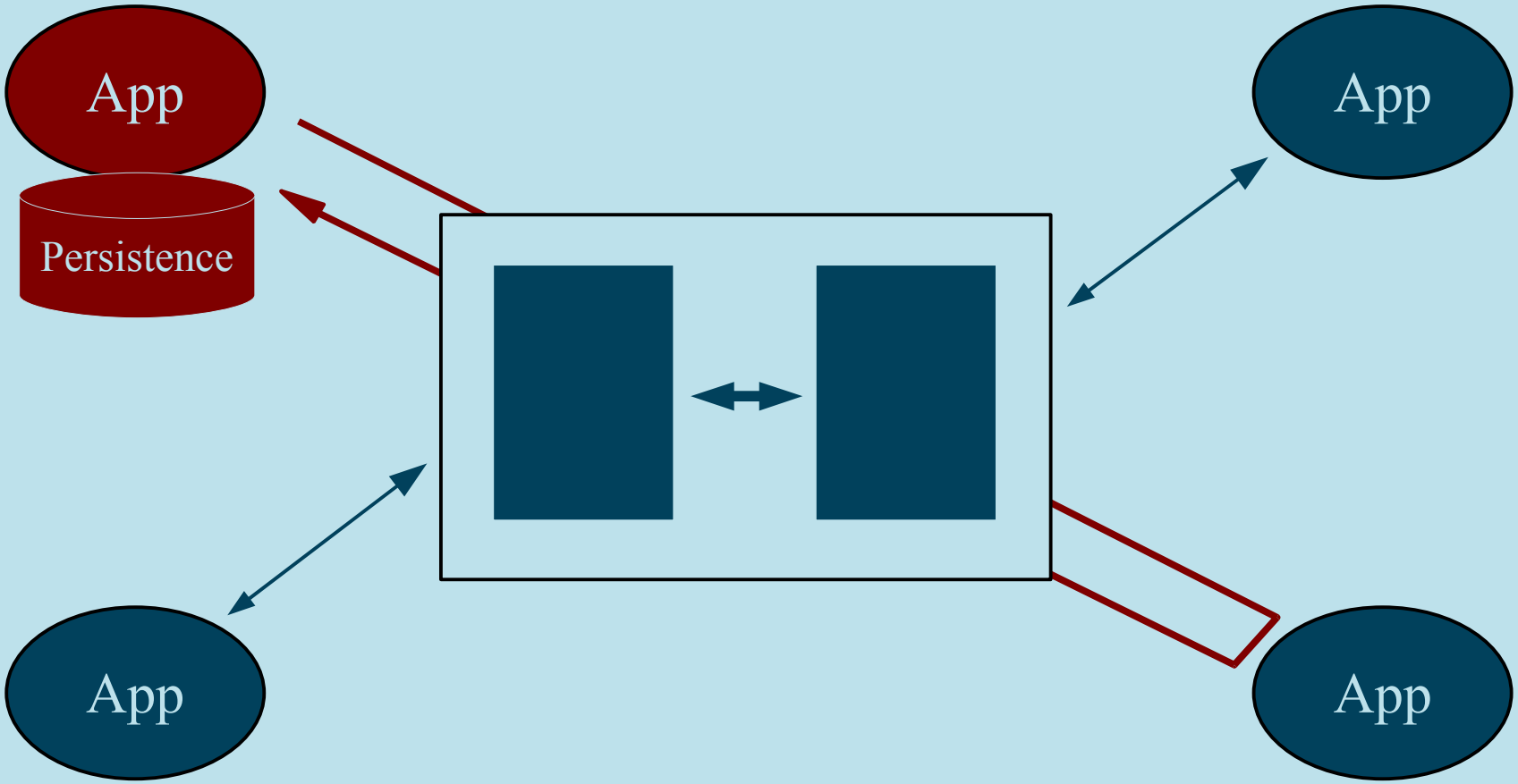
iMatix Corporation

RAMS Technique

- Client API provides two messaging APIs
  - Data distribution
  - Request-response (R-R)
- Reliability is implemented for R-R only
  - Record request in store
  - Send request to service
  - Wait for a matching response
  - If needed, send the request again
  - When response arrives, update store
  - After timeout, alert application to failure

iMatix Corporation

RAMS with a HA cluster



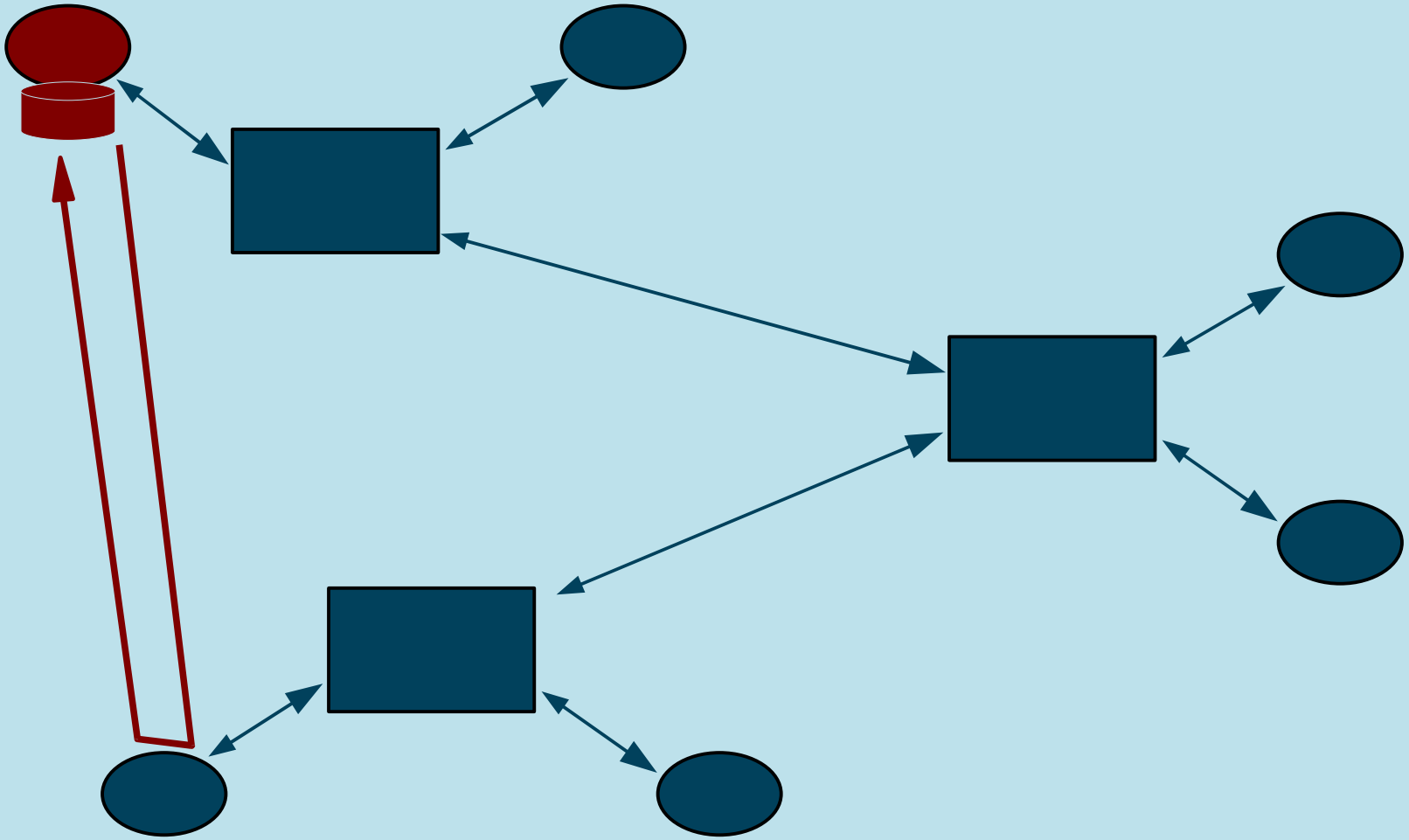
iMatix Corporation

RAMS + HA is simple

- No assumption of reliability in network
- HA is limited to peer-coordination
- No message flow between peers
- Proven solution in production use
- Broker remains relatively simple
- HA is invisible in the protocol
  - Persistence is also invisible

iMatix Corporation

What about wide-area networks?



iMatix Corporation

RAMS + WAN is simple

- No assumption of reliability in network
  - It does not matter how large network grows
- Reliability is orthogonal to network
- Allows use of untrusted middlemen
- Allows use of thin "edge" servers
- Moves towards "messaging in every wall-plug" vision

iMatix Corporation

Why is RAMS nice?

- Protocol remains simple
  - No reliability semantics
- Brokers remain simple & light
  - No persistence, no transactions, no acks
- Reliability is (almost) unilateral
  - No interoperability burden
  - Recipient must detect & discard duplicates
- Reliability is easy to understand
  - Very intuitive, obvious for programmers

iMatix Corporation

RAMS is fast

- Data distribution can be scaled
  - Fanout for high volumes & subscribers
  - Zero overhead for data distribution
- No centralized storage bottleneck
  - Reliable message broker can only run as fast as its data store
  - AMS runs at full speed
  - Persistence cost is spread out across network

iMatix Corporation

Can RAMS do everything?

- No, it cannot
- RAMS is a "90%" solution
  - Specifically for request-response
  - Also solves data distribution optimally
  - Does not do file distribution
  - Does not do other messaging models
  - But... these can be layered on top
- Cannot solve “MQ Series” challenge

iMatix Corporation

RAMS as a transport layer

- How do we implement file distribution?
- Consider RAMS as a transport layer
- Use only request-response messaging
- Construct services around RAMS
- File distribution over RAMS:
  - Through use of "file broker" application
  - RAMS to and from file broker

iMatix Corporation

Switch vs Broker

Switch

- Simple protocol
- Simple server
- Edge storage
- Intelligent API
- WAN friendly
- Silicon friendly
- Unconventional

Broker

- Complex protocol
- Complex server
- Central storage
- Simple client API
- WAN hostile
- Silicon hostile
- Conventional

iMatix Corporation

Conclusions

- RAMS has significant advantages
  - Simply, cheap, flexible
  - Compatible with move to hardware
  - Elegant solution for HA & WAN
- Reliable message broker is outdated
  - Solves problems of the past
- RAMS is future-proof design
  - Cheap, scalable, simple
  - Forces standard messaging patterns

iMatix Corporation

Thank you

- For more information please contact the author at ph@imatix.com.

iMatix Corporation