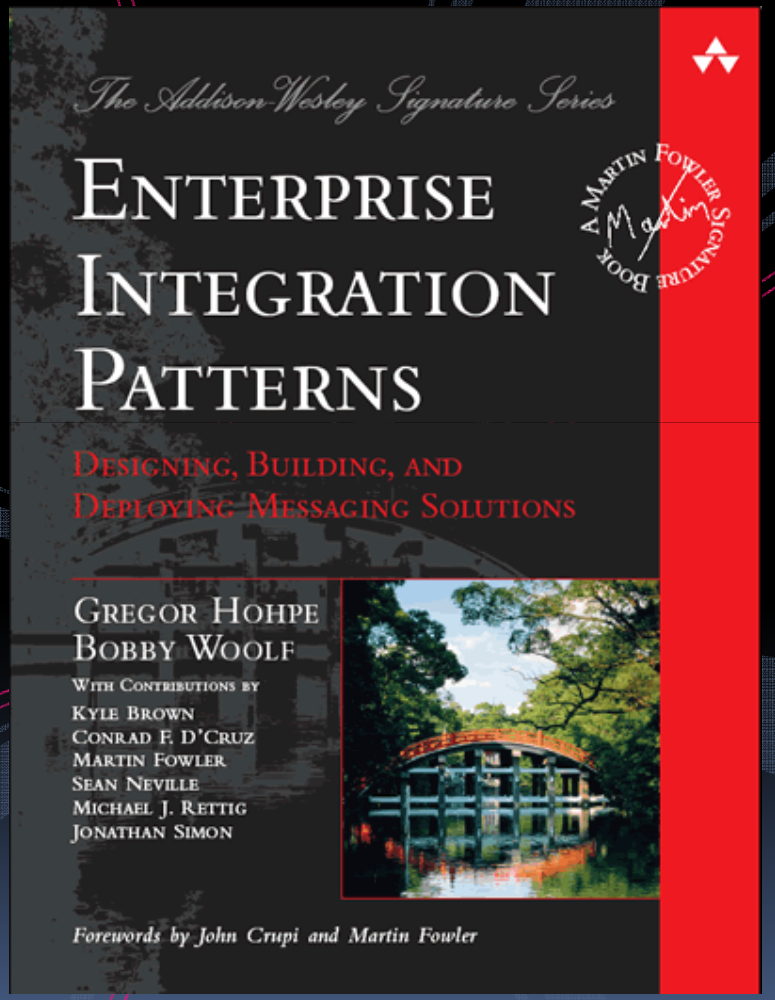
Enterprise Integration Patterns

A quick tour….

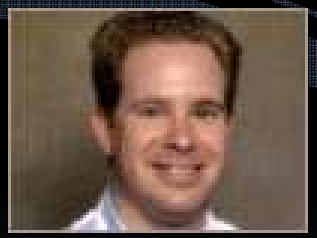


Eva Shon
Software & Systems Engineering Seminar

www.enterpriseintegrationpatterns.com

Enterprise Integration Patterns

The Authors:



Bobby Wolfe

Gregor Hohpe

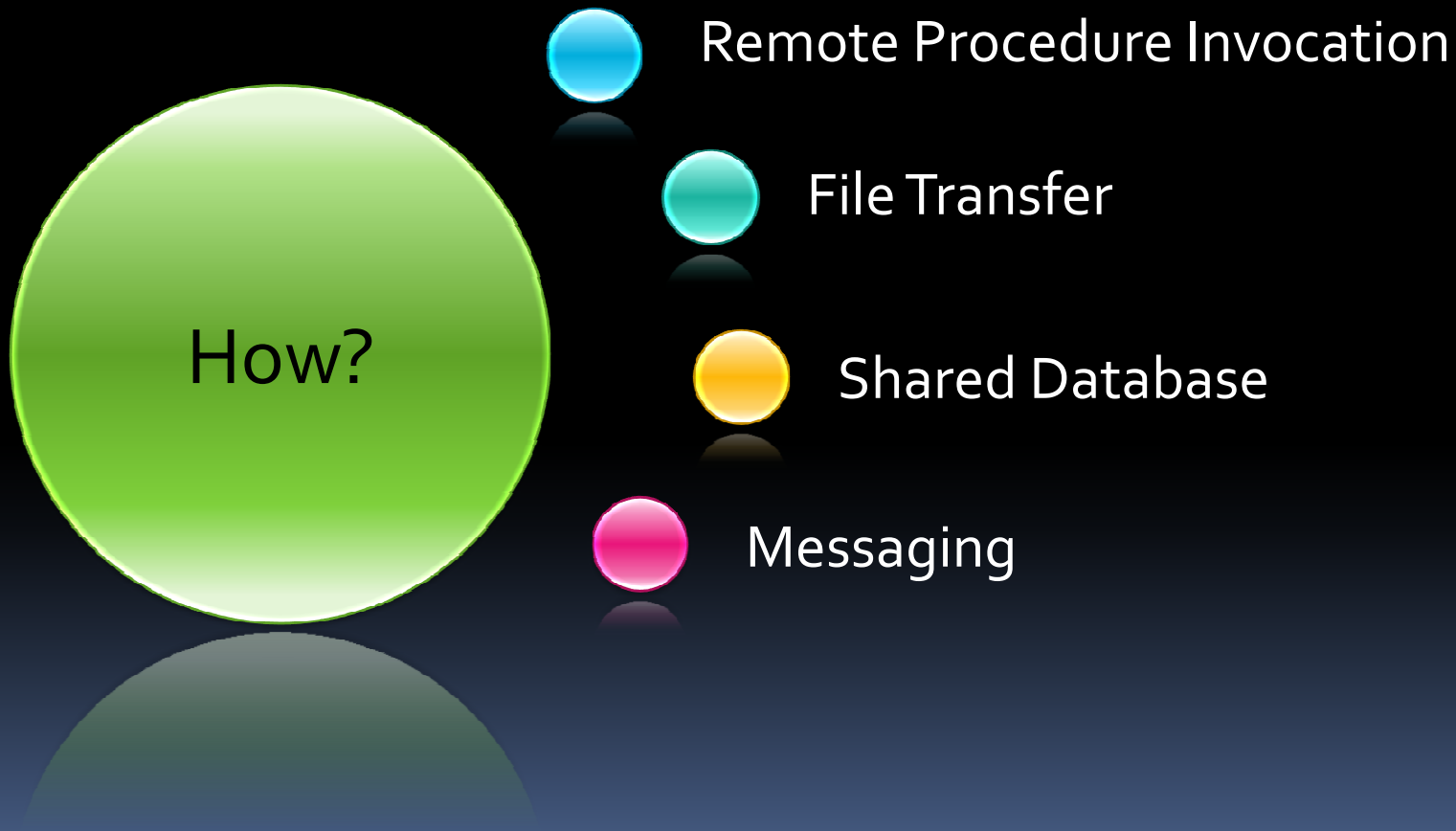
Enterprise Integration Patterns

1. ***Fundamentals: Messaging***
2. Patterns

Fundamental Challenges of Integration

- Networks are slow
- Networks are unreliable
- Applications can be different
- Change is inevitable

4 Main Approaches to Integration



(multiple styles can be combined)

Enterprise Integration

- Not a single distributed application
- Independent applications interacting
- Sharing data *and* processes
- The authors believe asynchronous messaging is the foundation because …

Enterprise Integration

Asynchronous Messaging is ...

more reliable than

Remote Procedure Invocation

more immediate than

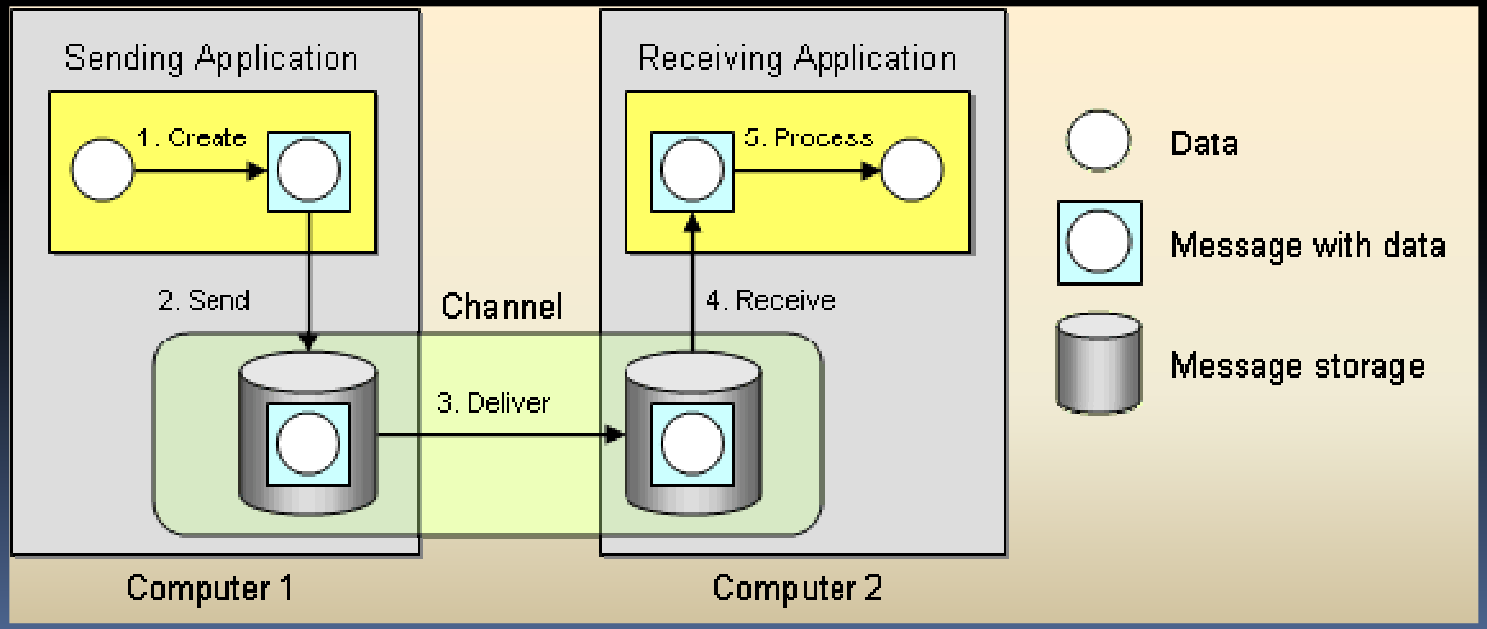
File Transfer

better encapsulated than

Shared Database

Asynchronous Messaging

- loose coupling
- program-to-program communication with reliable delivery
- send & forget, store & forward



Message-Oriented Middleware(MOM)

provides:

- remote communication
- platform / language Integration
- asynchrony
- throttling
- variable timing
- reliability
- mediation (broker)
- federation (façade)
- disconnected operation
- reduction in # of threads

Message-Oriented Middleware

must define:

- interface to applications
- channels with producers, consumers
- when to send
- in what format
- routing (even without full knowledge of receiver)
- error management & monitoring
- connection management
- senders and receivers make no (or few) assumptions about each other's identity

Message-Oriented Middleware

Challenges to using them:

- complex programming model
- sequencing
- synchrony when needed by apps
- performance
- limited platform support
- vendor lock-in
- how to test
- lack of programmers who know how to use

Message-Oriented Middleware

- Toolkits:
  - Application servers
    - J2EE Servers with Java Messaging Service (JMS)
    - IBM WebSphere
  - Operating Systems
    - Microsoft Message Queuing (MSMQ) in XP, Vista, .NET
    - D-Bus in Linux
    - OracleMQ
  - Enterprise Application Integration Suites
    - IBM WebSphere MQ, TIBCO, BizTalk, ActiveMQ, ServiceMix, Camel
  - Web Service toolkits
    - WS-ReliableMessaging, WS-Reliability

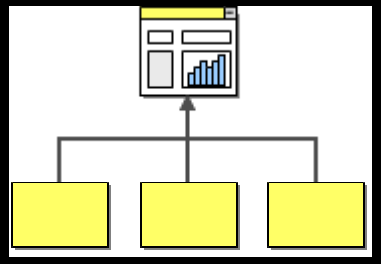
Enterprise Integration

- Okay, so it's based on:
  - Asynchronous Messaging
  - Independent Applications
  - Message-Oriented Middleware
  - But to build what?
    - Tens to hundreds of applications collaborating
    - When clear separation of boundaries is difficult

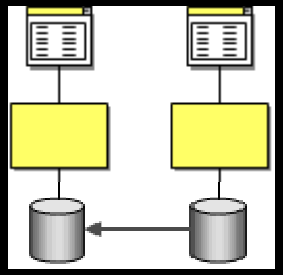
Some Common *Types* of Integration Projects

Note: These are NOT the Patterns!

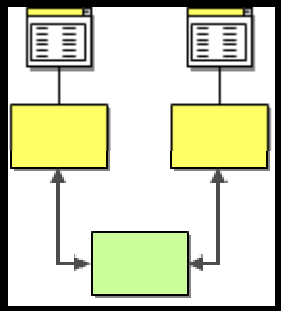
Information Portal



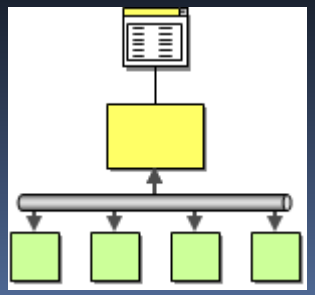
Data Replication



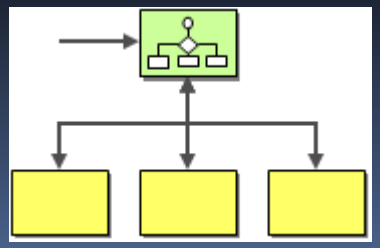
Shared Business Function



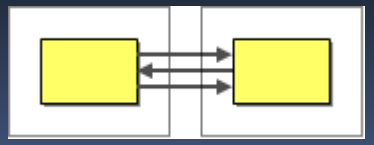
Service-Oriented Architecture



Distributed Business Process



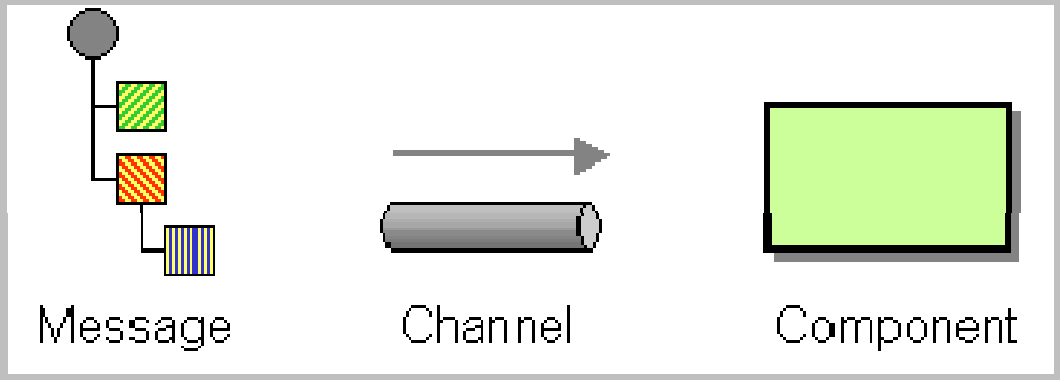
Business-to-Business Integration



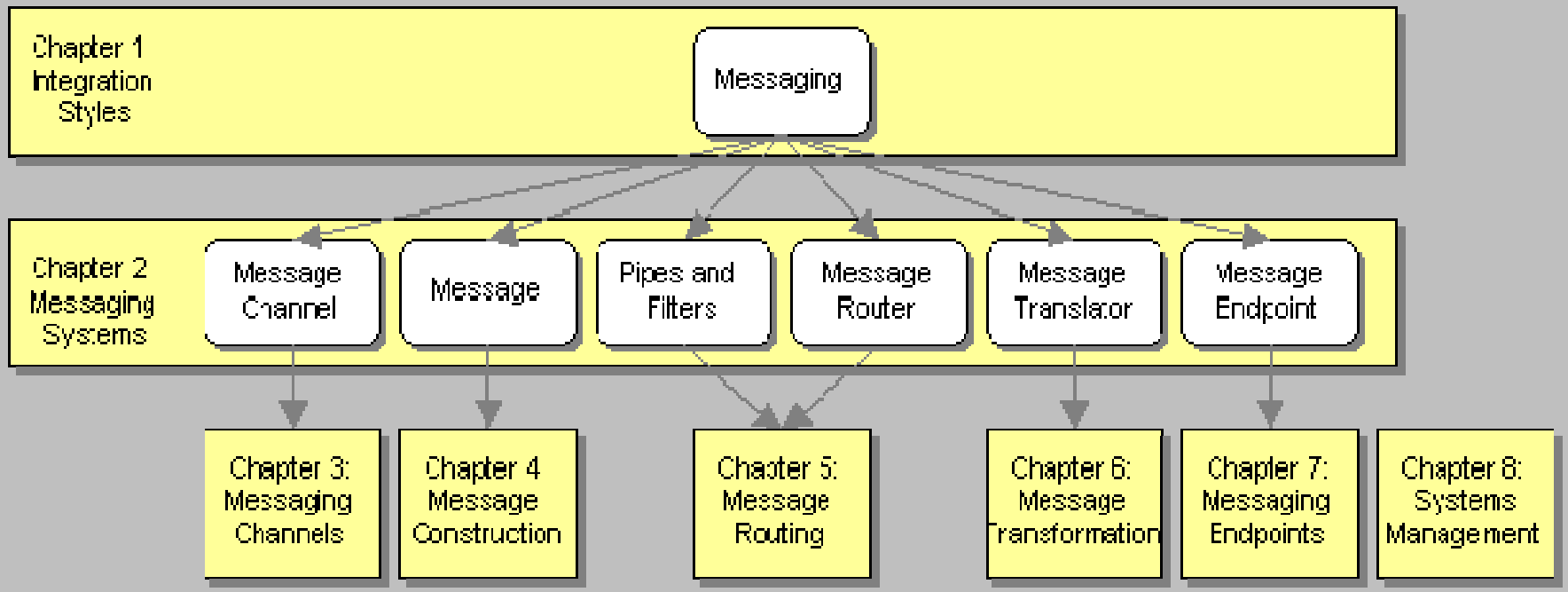
Enterprise Integration Patterns

1. Fundamentals: Messaging
2. ***Patterns***

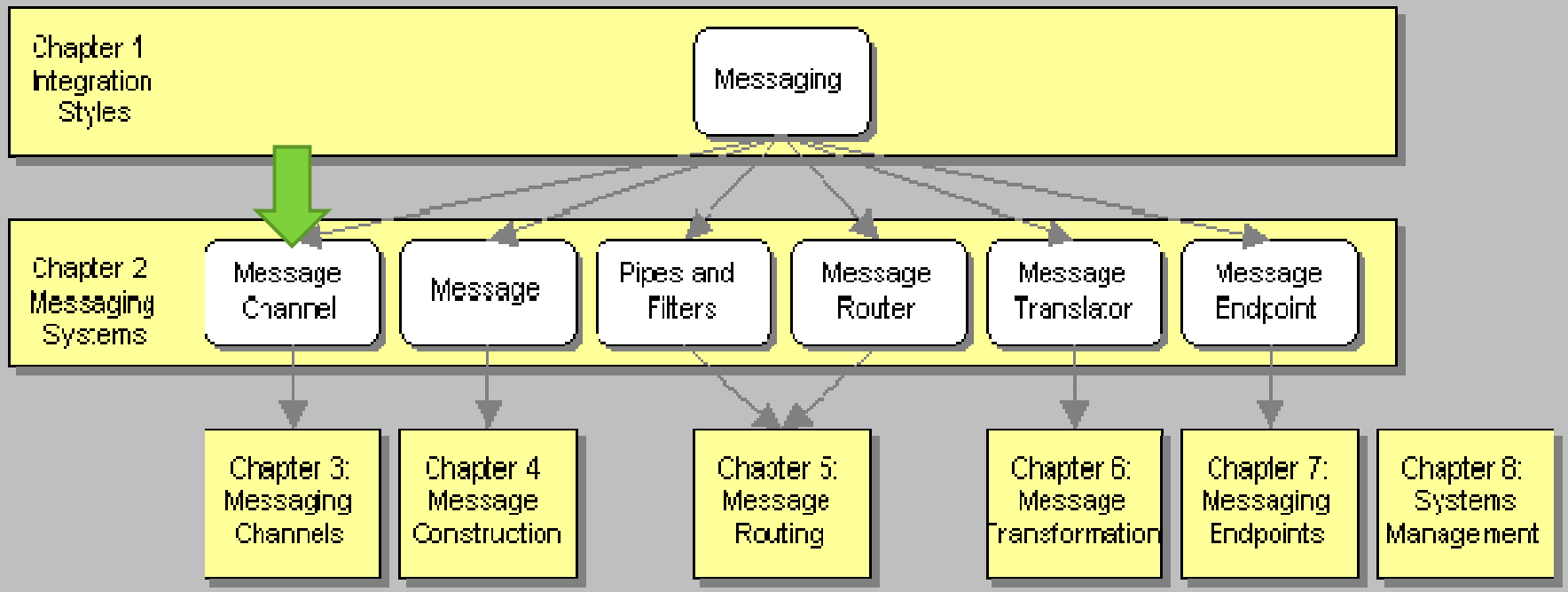
Basic Components



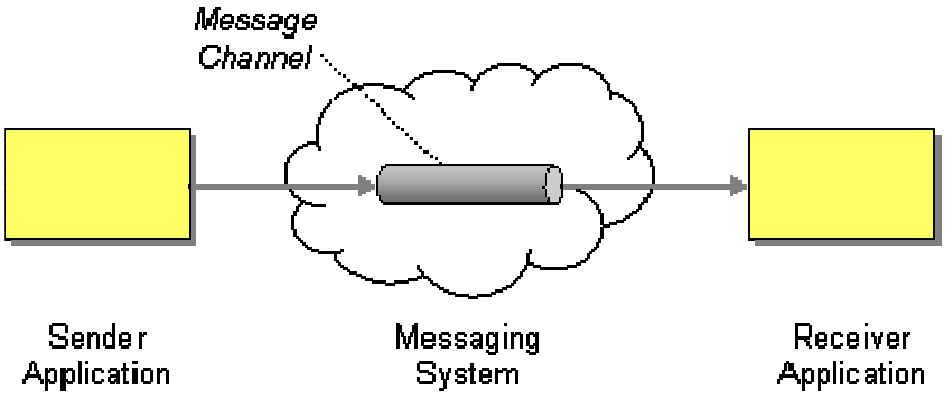
Root Pattern Hierarchy



Root Pattern Hierarchy



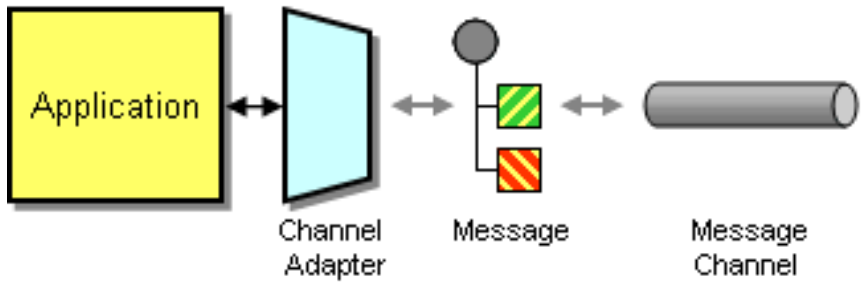
Message Channels: Building



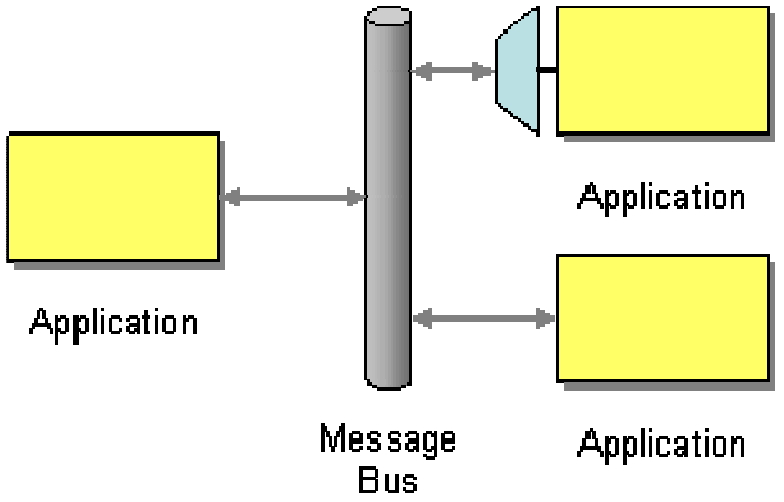
- Known at deployment time, static?
- Who decides what channels are available?
- How many channels will be needed?
- Unidirectional

Message Channels: Patterns

- Channel Adapters

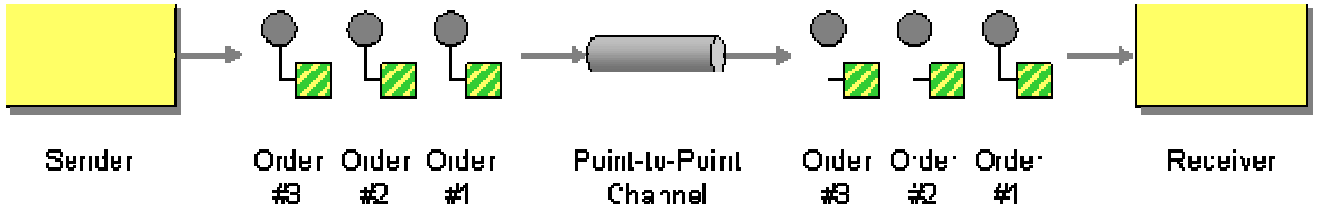


- All-access "Bus"

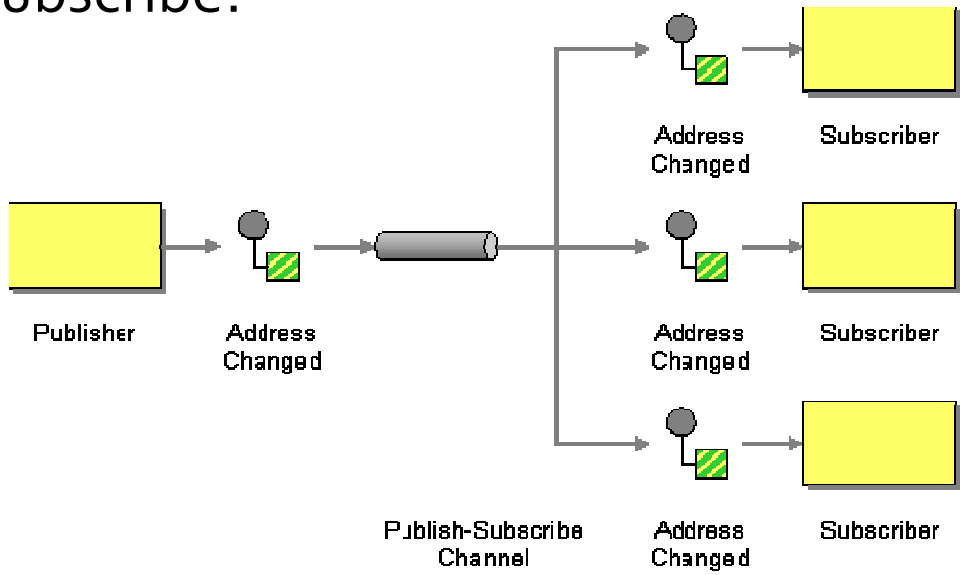


Message Channels: Patterns

Point-to-point

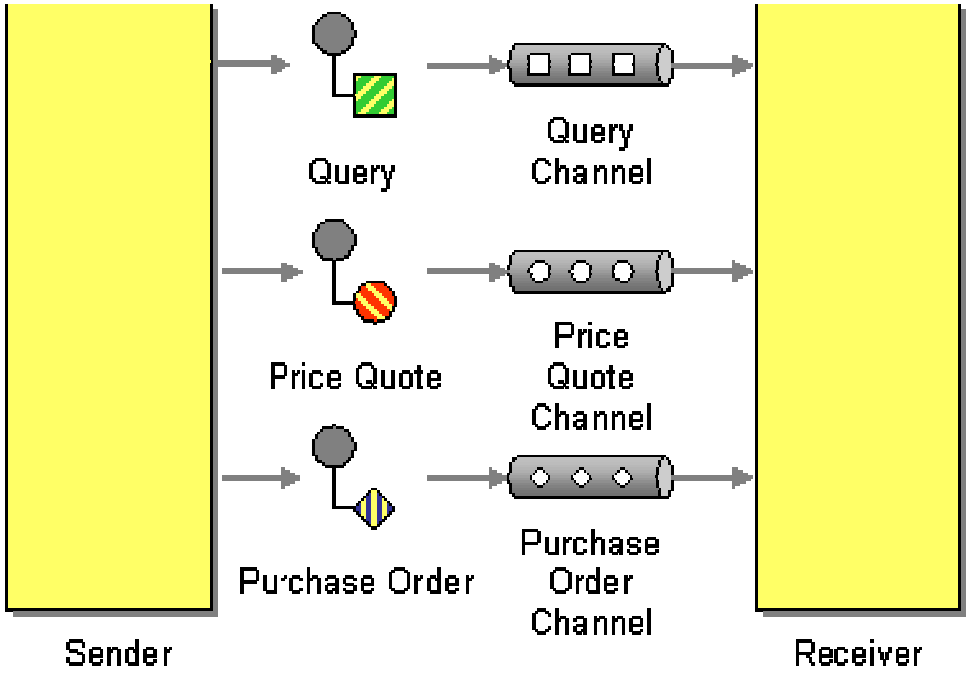


or Publish-subscribe?



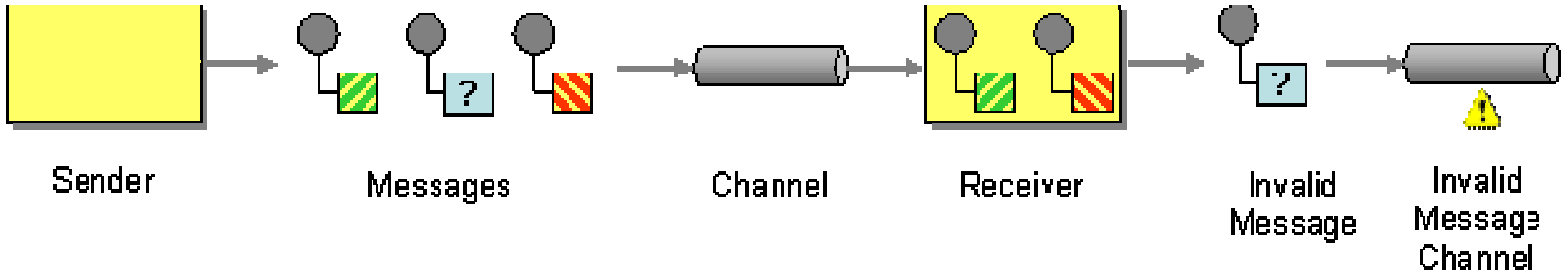
Message Channels: Solutions

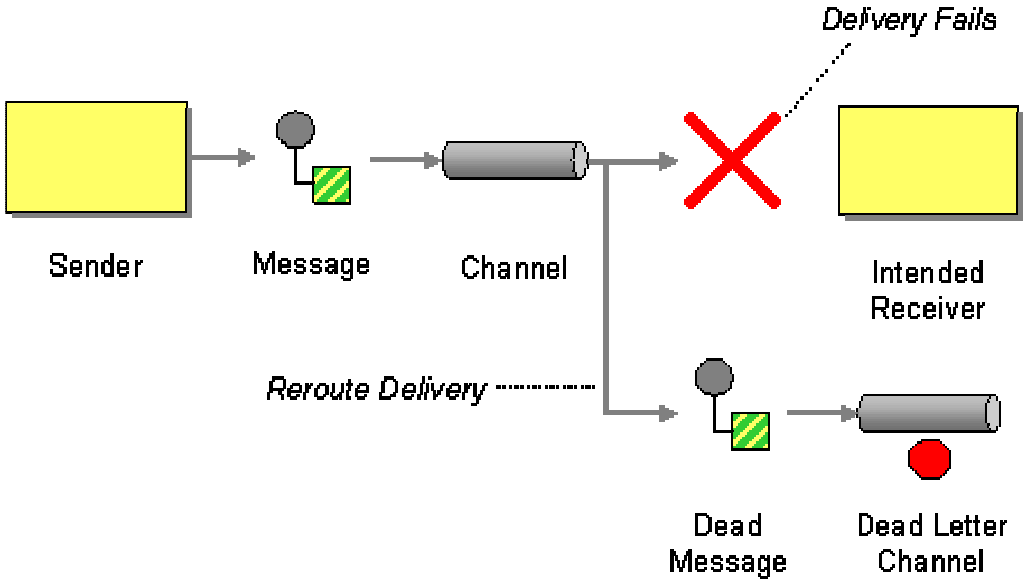
Different Data Types?



Message Channels: Solutions

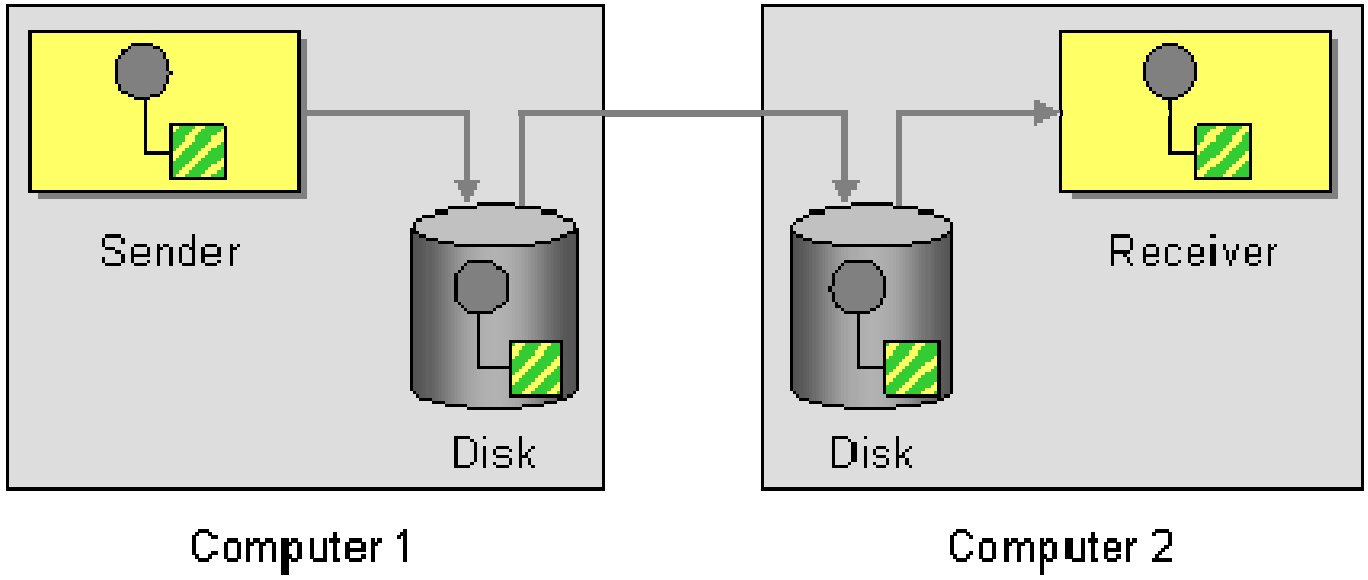
Invalid messages?





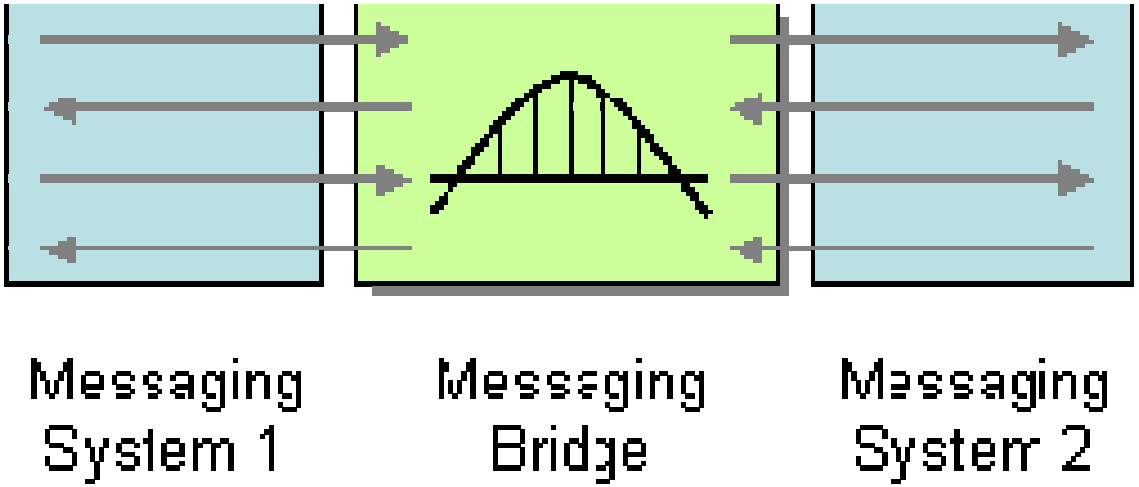
Message Channels: Solutions

Crash proof? Guaranteed Delivery Pattern



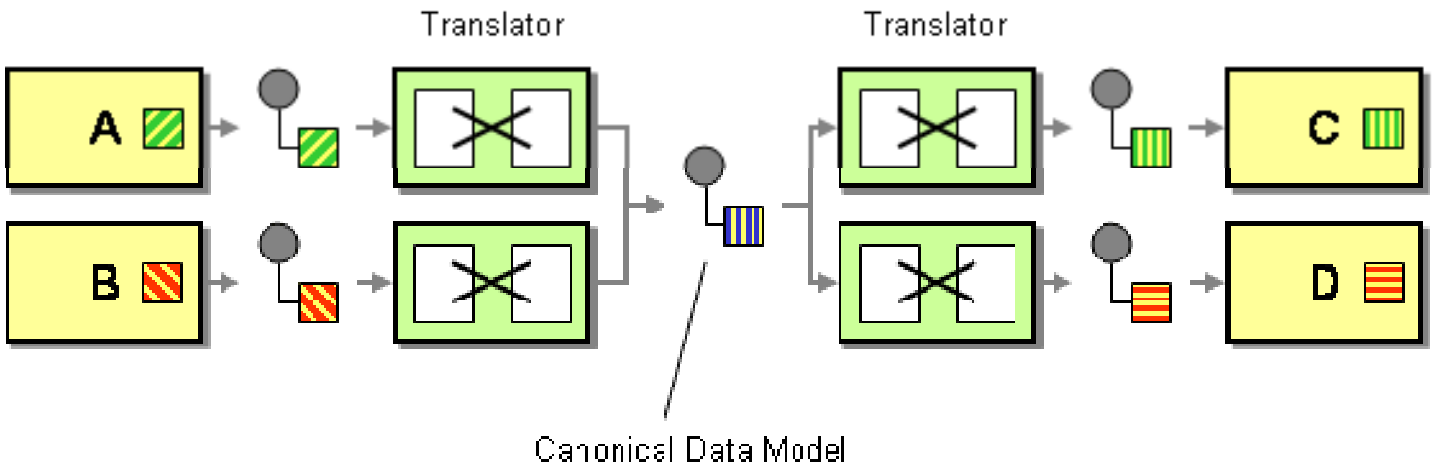
Message Channels: Solutions

Multiple Messaging Systems?



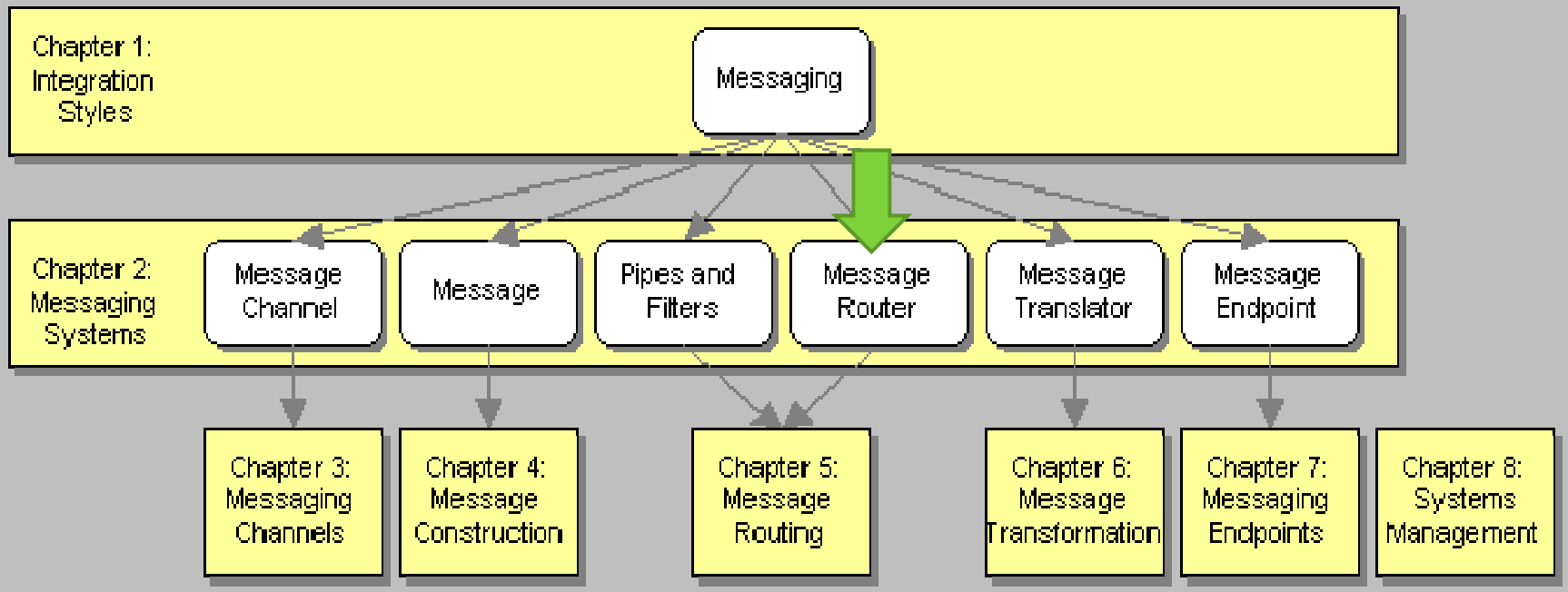
Message Channels:

Applications with different data formats



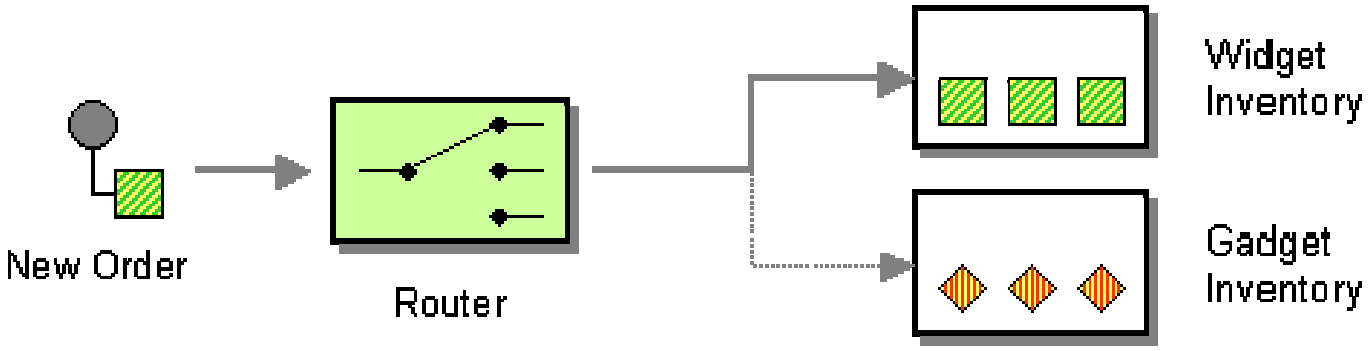
Canonical Data Model + Translators

Root Pattern Hierarchy

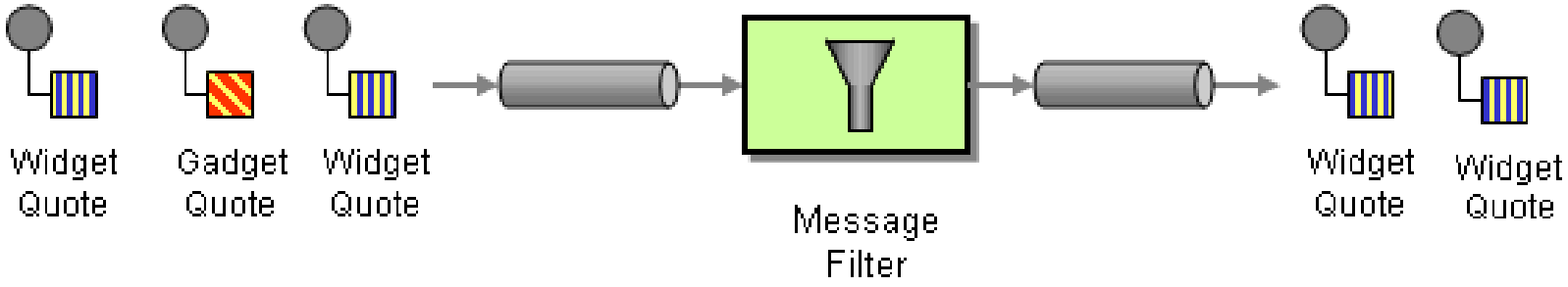


Message Routers

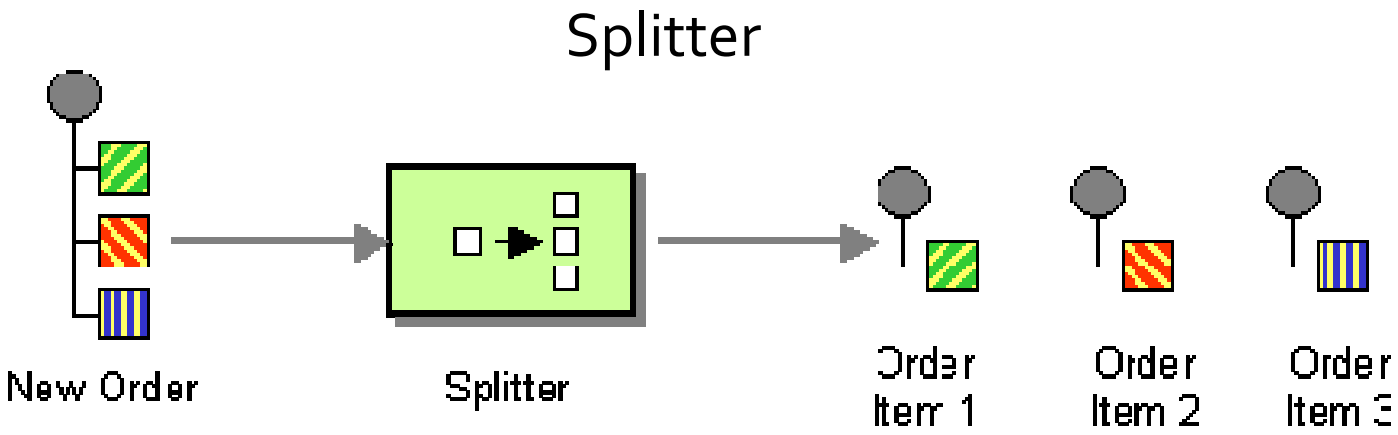
Content-based Router

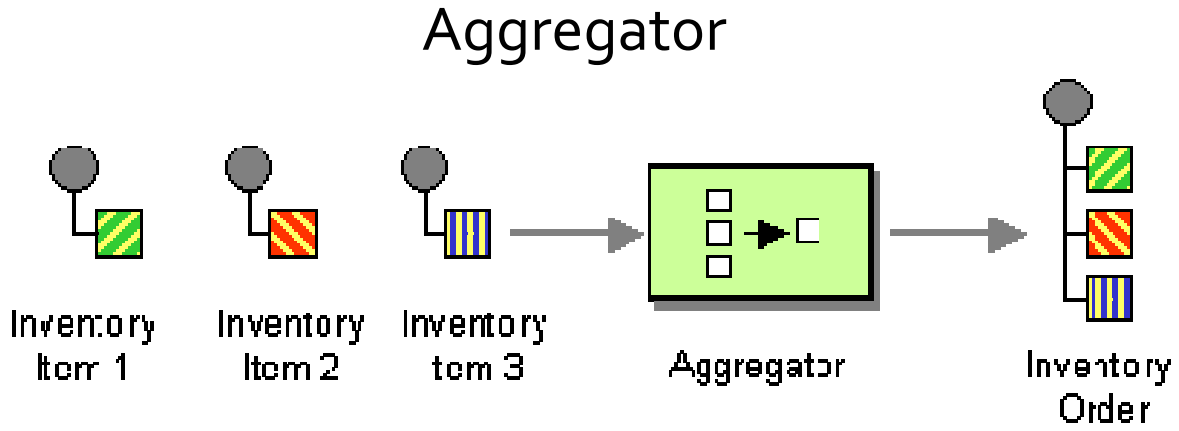


Message Filter



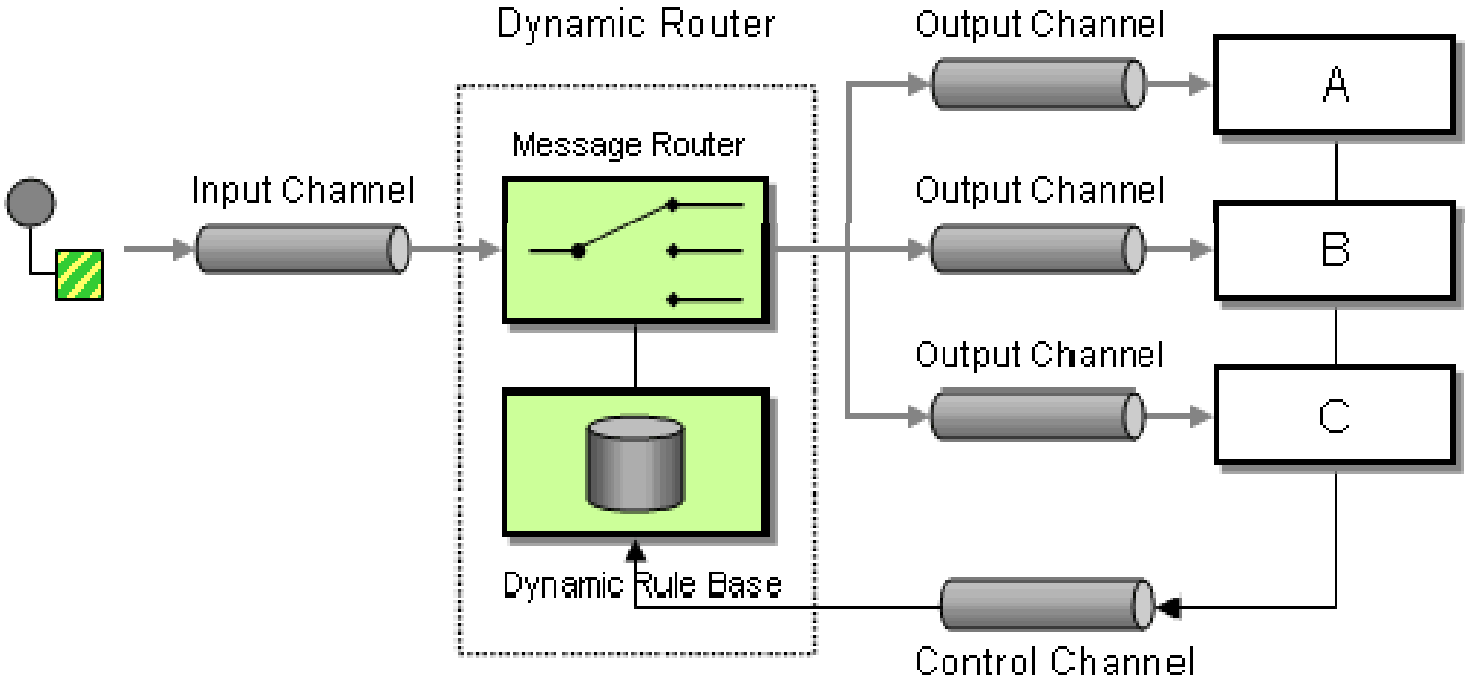
Message Routers



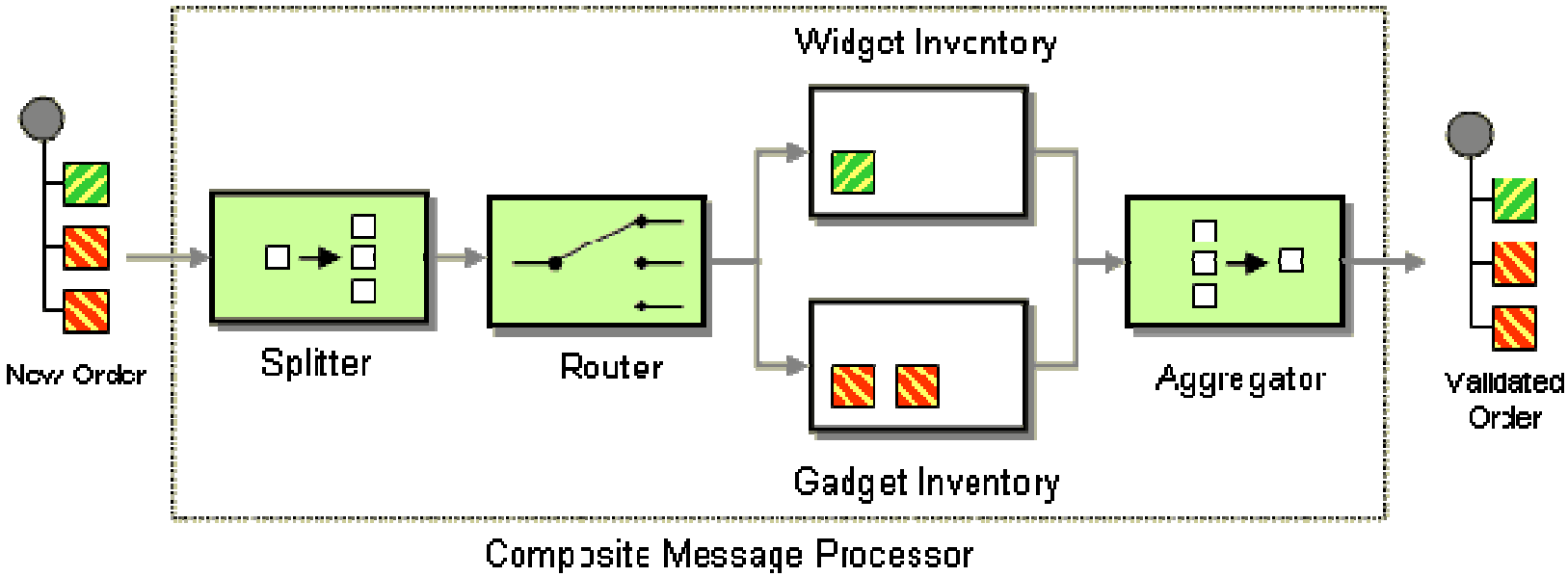


Message Routers

Dynamic Routing (recipients announce their preferences)



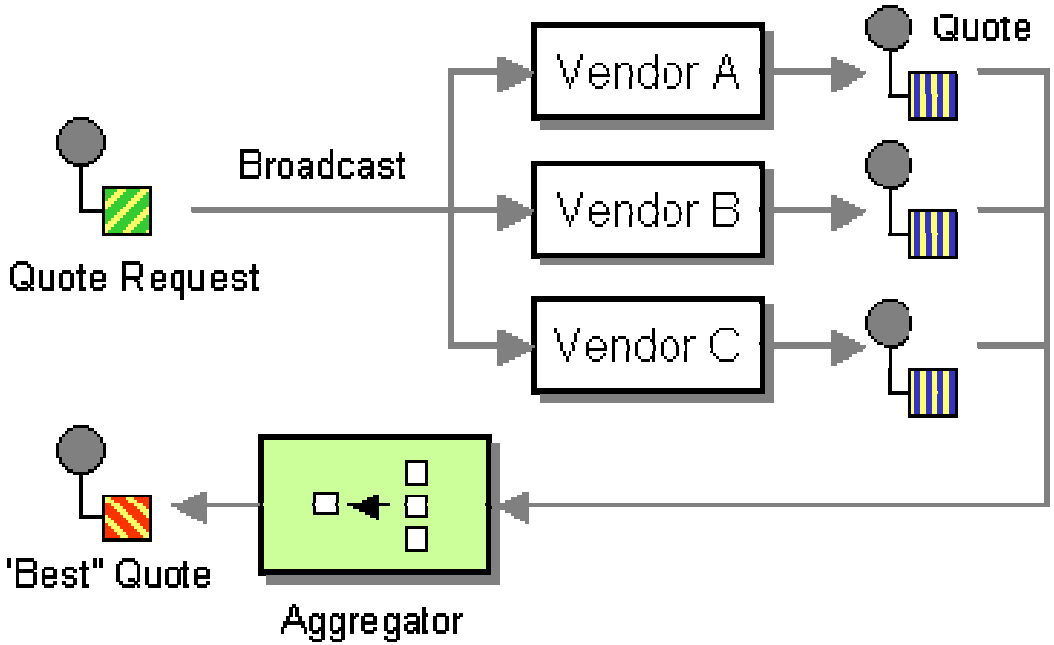
Message Router Composition



- Composing is supposed to be easy if the original messaging architecture is based on Pipes & Filters

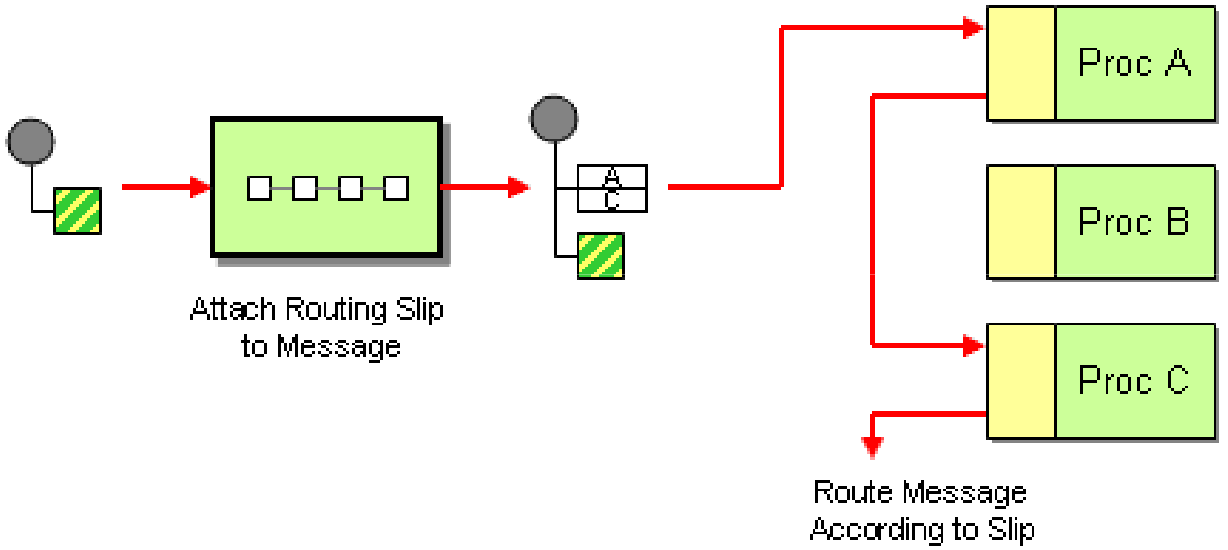
Message Router Composition

"**Scatter-Gather**": Basic Bidding



Message Router Composition

Don’t know the processing sequence at design time?

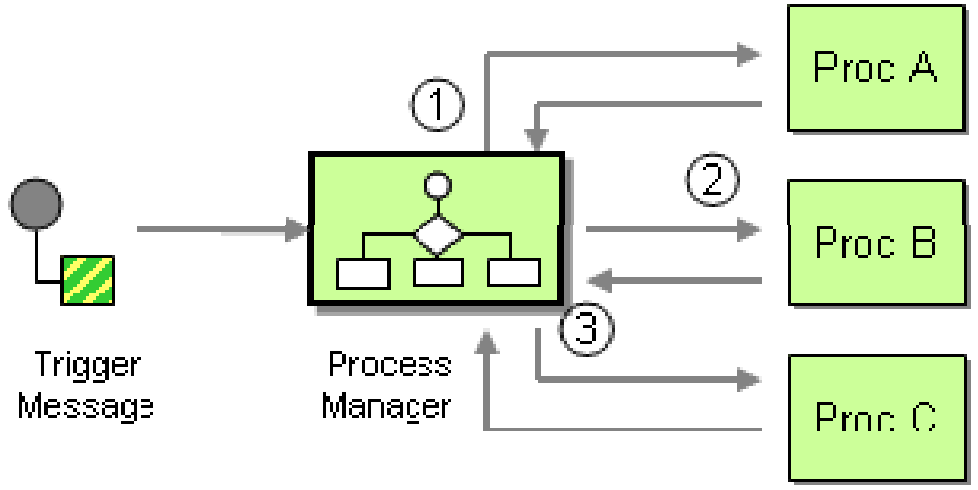


Routing Slip Pattern

Message Router Composition

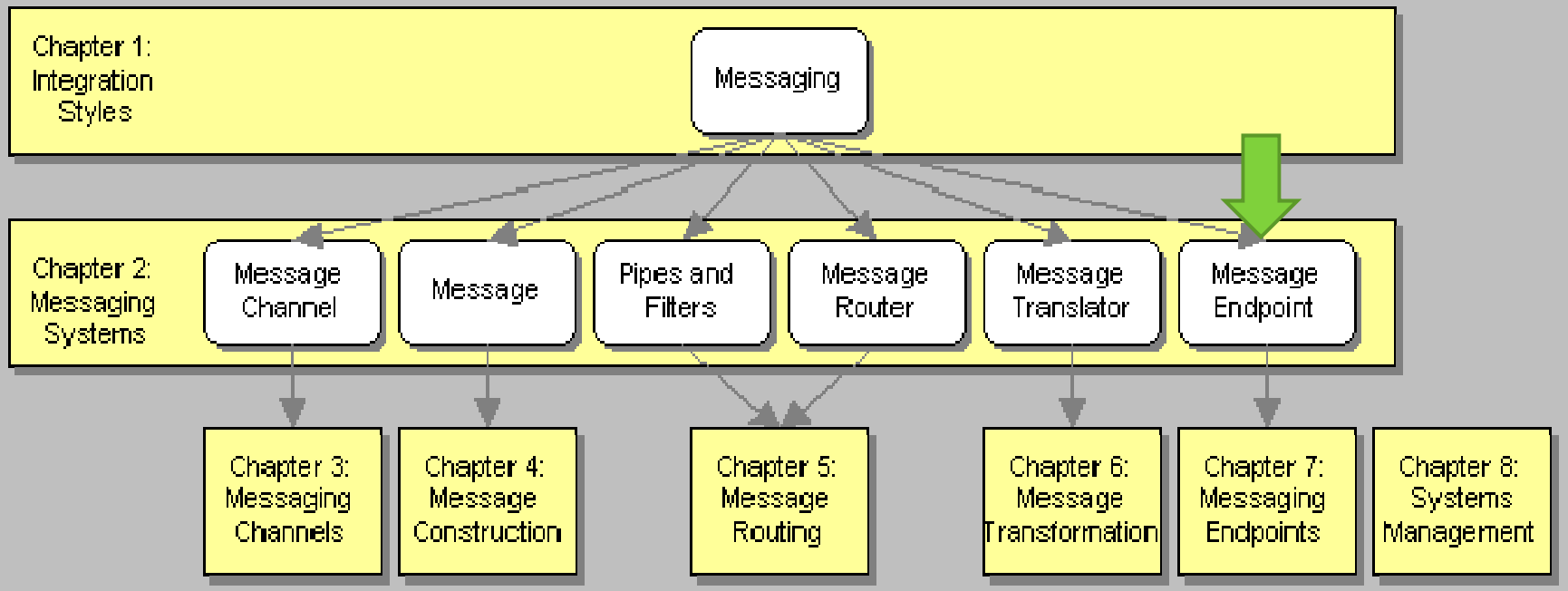
Unknown processing sequence that may not be sequential?

Process Manager Pattern



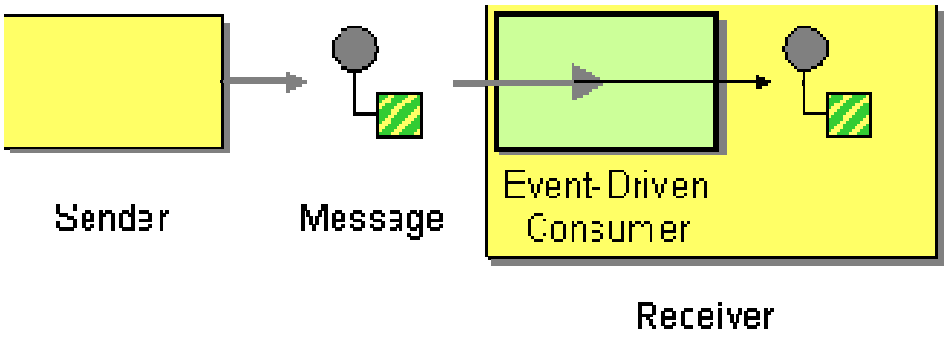
- Process template (orchestration)
- Process instance (stores data in a database)
- Can turn this into an SOA by adding:
- 1) registry 2) service interface contracts 3) bus

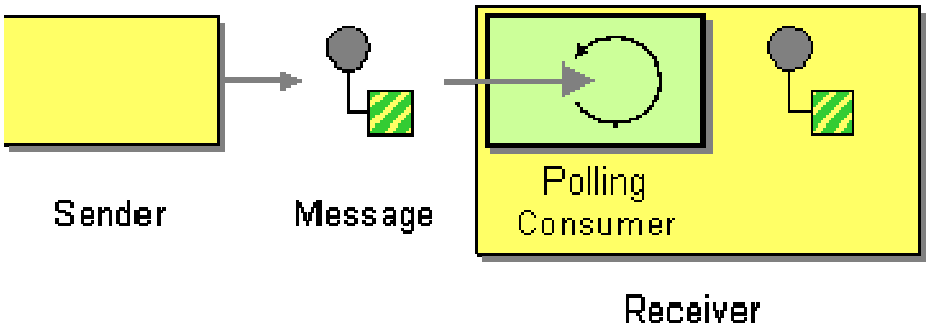
Root Pattern Hierarchy



Message Endpoint Styles

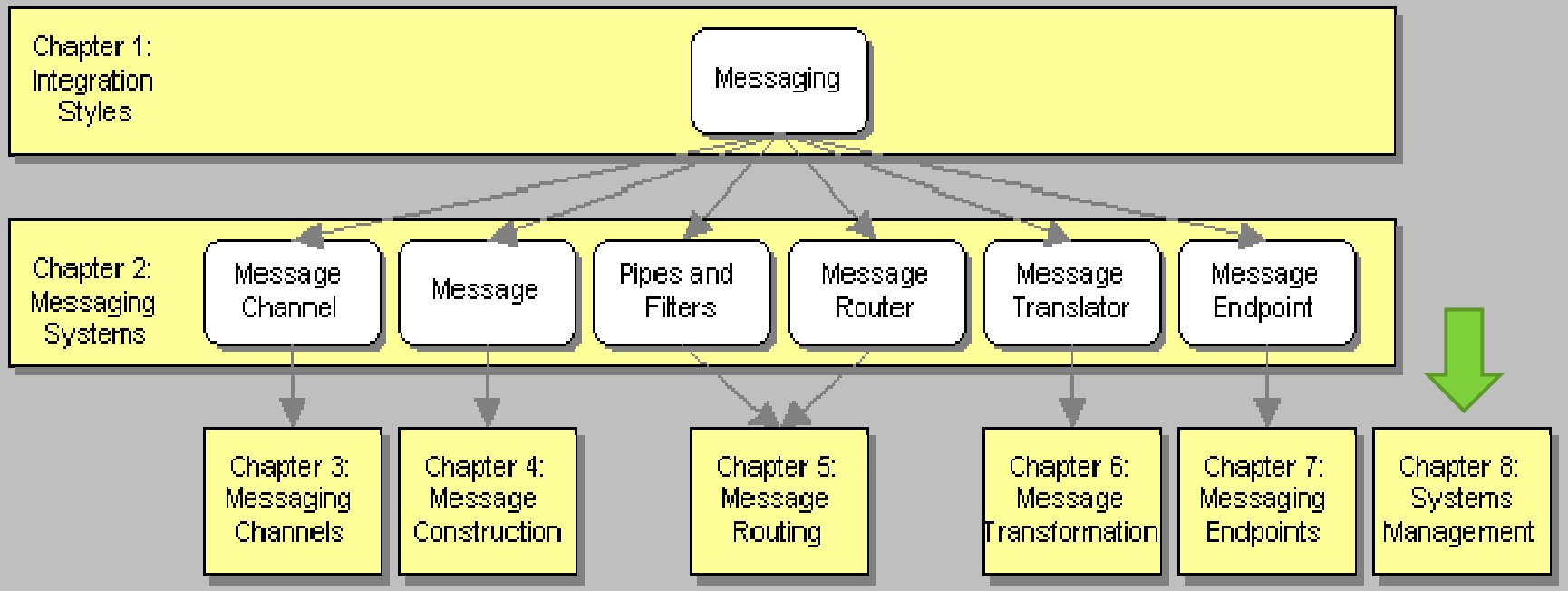
Event-Driven Consumer





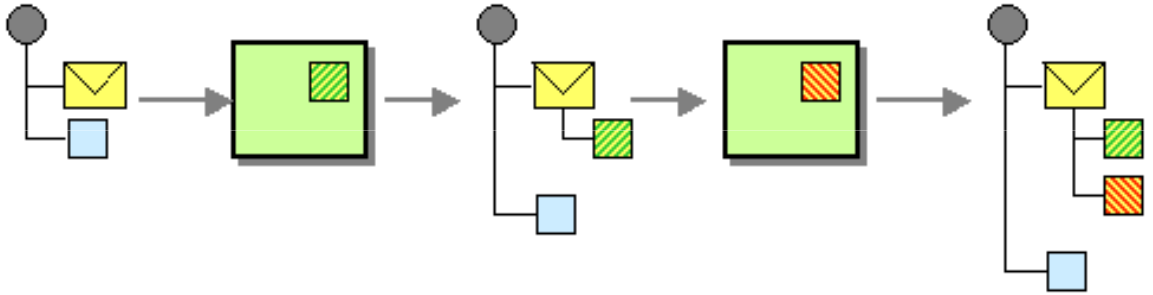
Polling Consumer Pattern

Root Pattern Hierarchy



System Management Patterns

Message History



Keep the history of components the message passes thru with the message itself

Good for testing

Enterprise Integration Patterns

- Like an Architecture Description Language
- Integration Solutions get complicated very quickly, so these are useful abstractions
- Free Visio stencil
- In conclusion, useful for an overview of Enterprise Integration designs

THE END

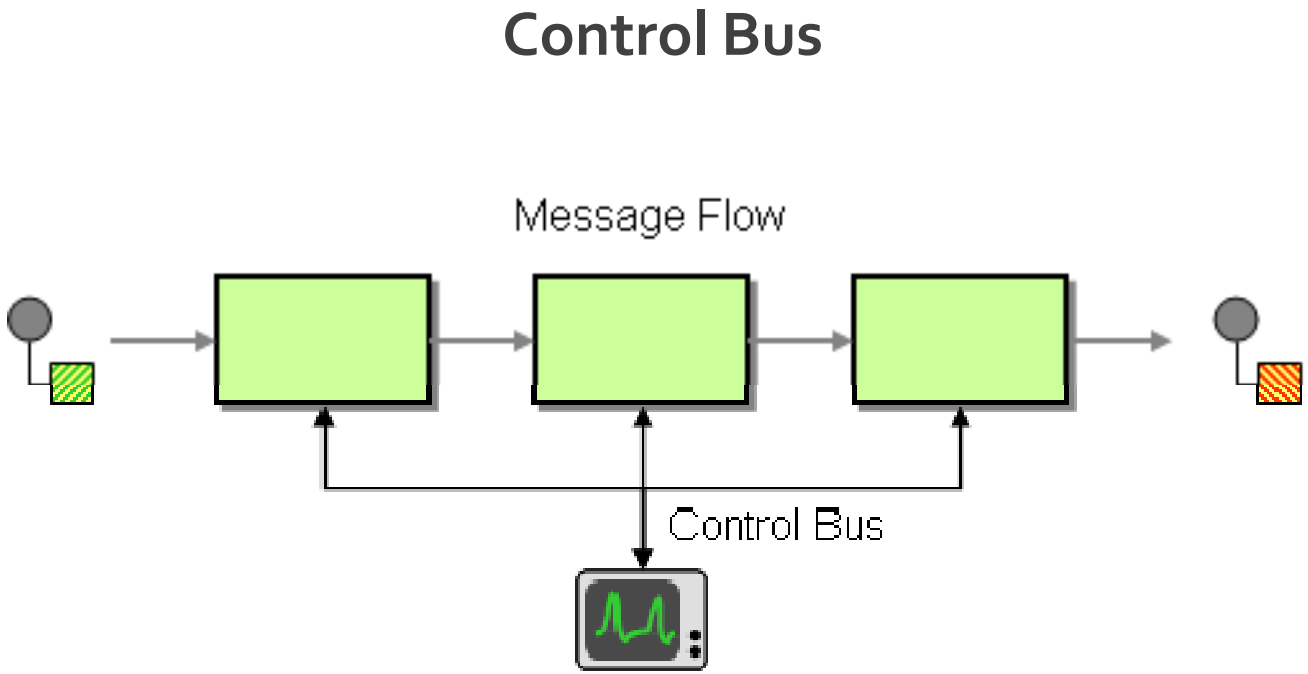
References

- Gregor Hohpe & Bobby Woolf . *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions*. Addison-Wesley, 2004.

- Apache Camel Documentation http://activemq.apache.org/camel/enterprise-integration-patterns.html

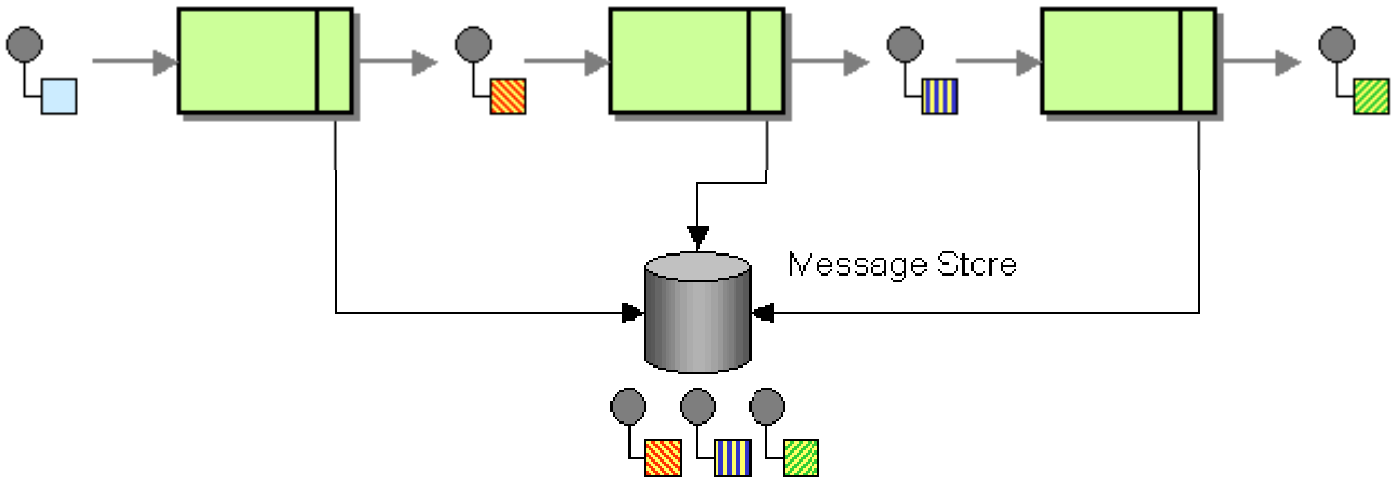
Extra Pattern Slides

System Management Patterns



System Management Patterns

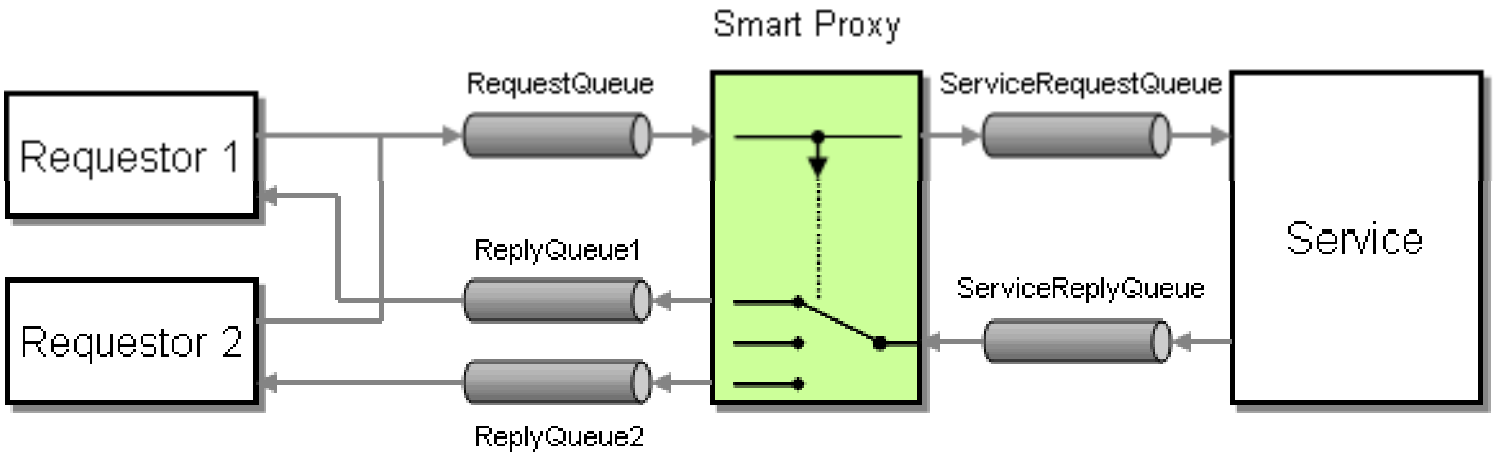
Message Store



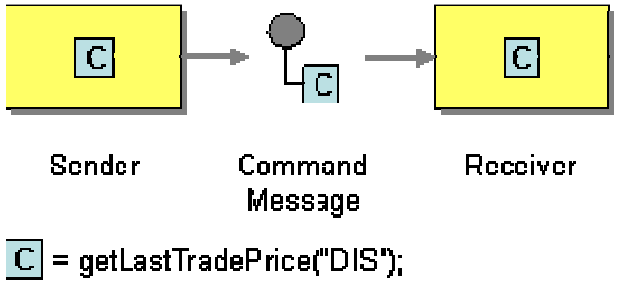
Good for testing

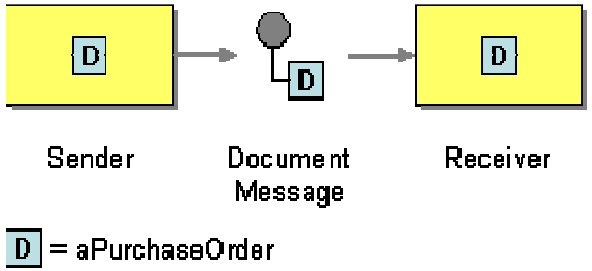
System Management Patterns

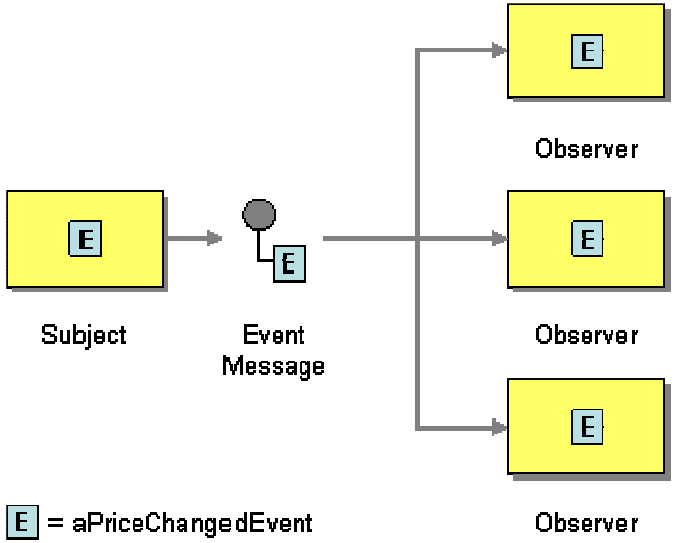
Smart Proxy



Messages: Message Intent

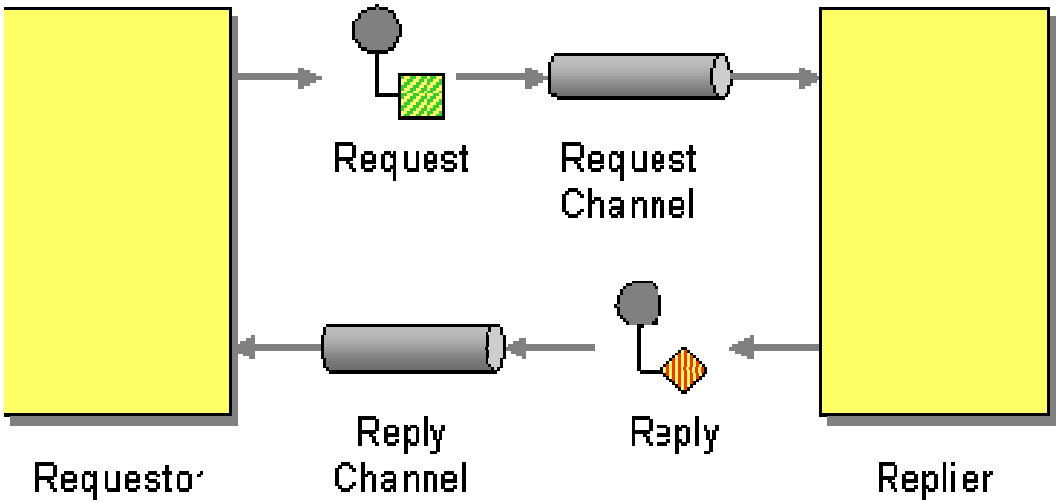






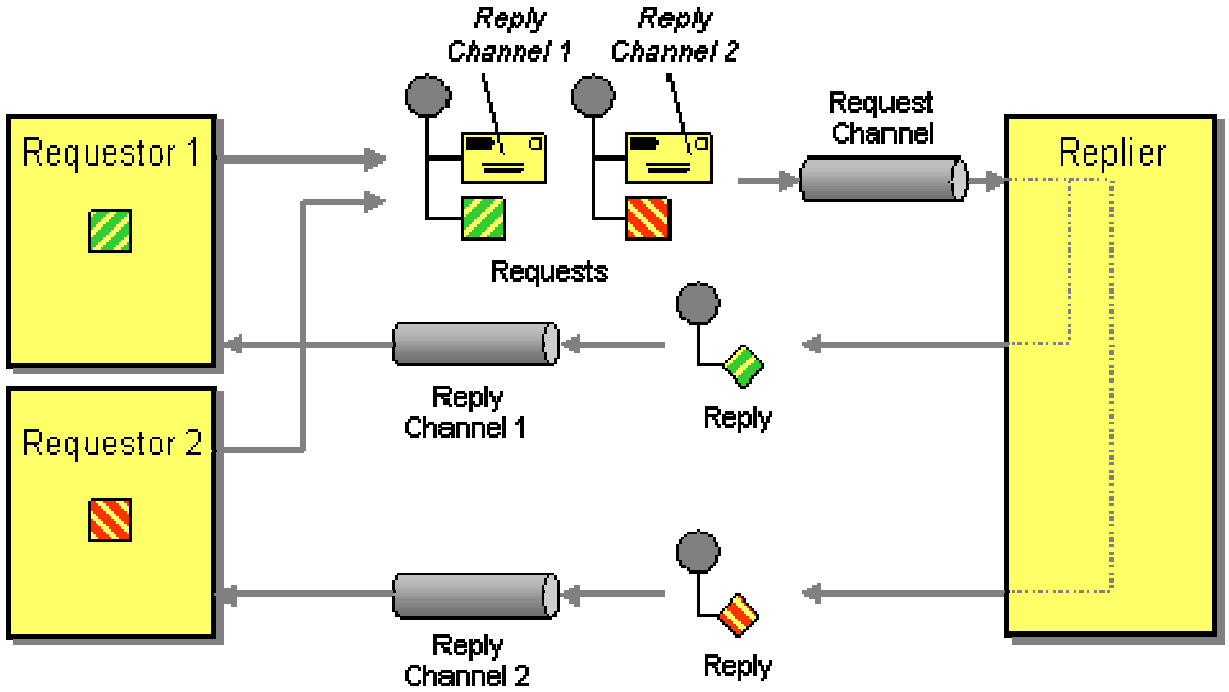
Messages: Request-Reply

Basic Request-Reply Pattern



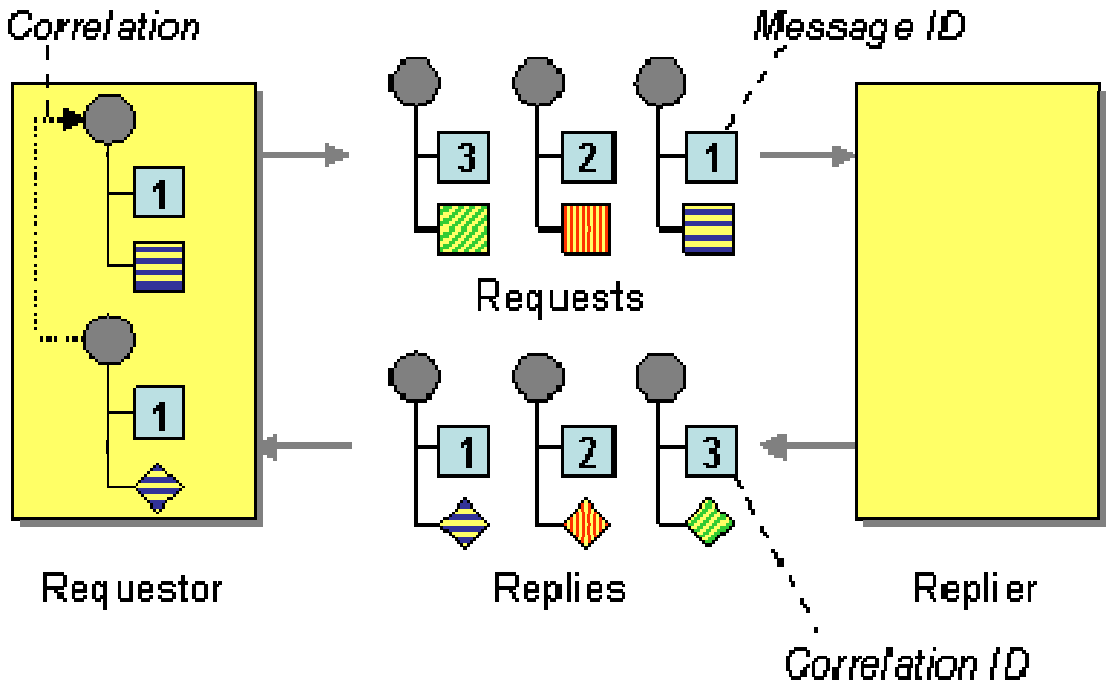
Messages: Request-Reply

Different Requestors? Return Address Pattern



Messages: Request-Reply

Multiple types of requests? Correlation Identifier Pattern



Messages: Diverse Data Formats?

Canonical Data Model + Translators