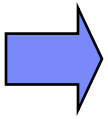
The ESB Architectural Pattern

Topics



- Introduce the concepts of a Service Oriented Architecture (SOA)
- Introduce the Enterprise Service Bus architectural pattern
- Evaluate the appropriate application of several ESB implementations
- Review a methodology for designing an SOA using an ESB
- Introduce the concepts of SOA governance and the role of a service registry and repository

Topic Agenda

- **What is an Enterprise Service Bus?**
- **The Enterprise Service Bus Pattern**
- **Service Virtualization**
- **What's inside an Enterprise Service Bus?**

What is an Enterprise Service Bus (ESB)

ESB is an “Architectural Pattern”

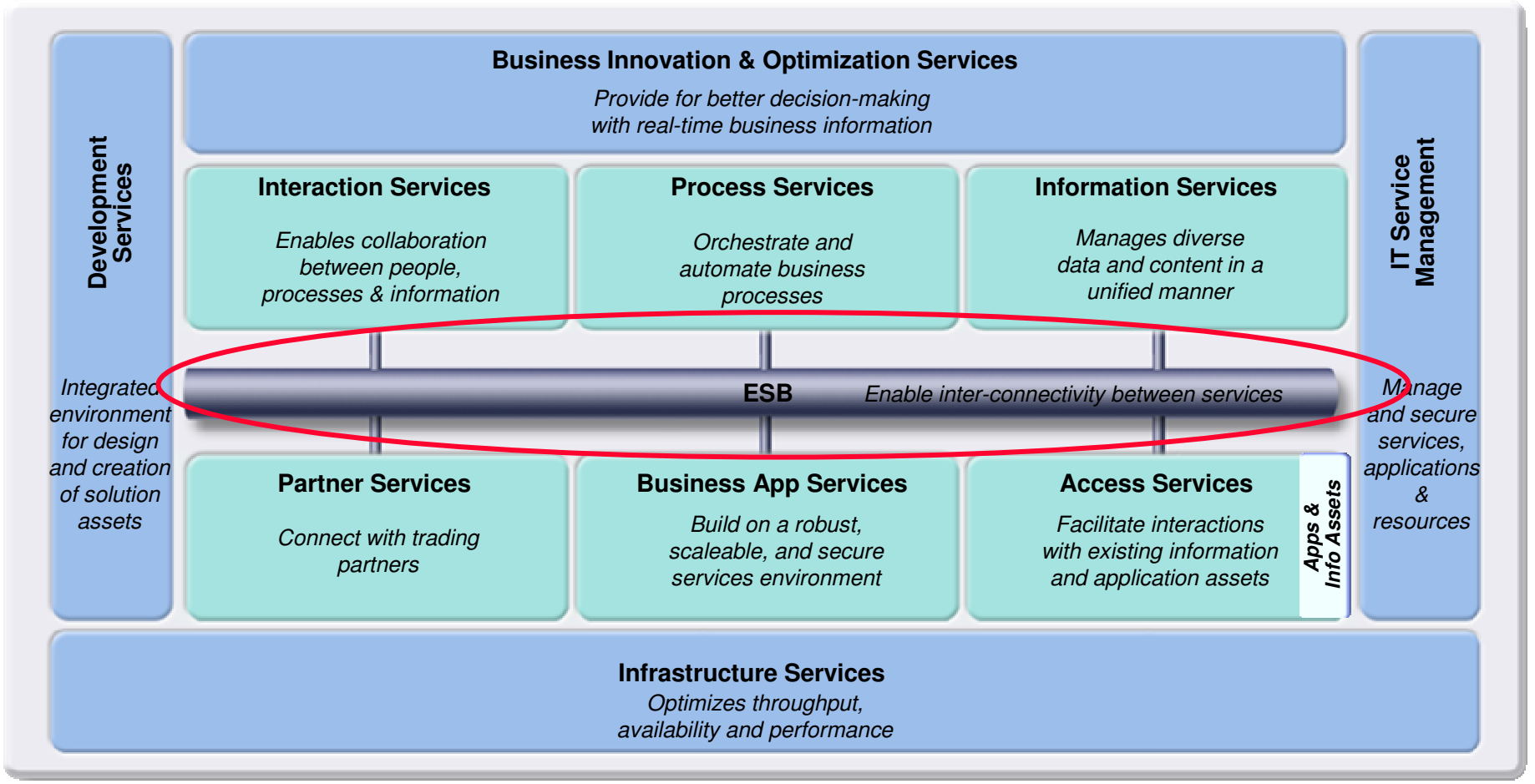
*“ We describe the enterprise service bus first and foremost as **an architectural pattern**. In fact, it is possible to construct service buses from a variety of different underlying integration technologies.*

***The architecture pattern** remains valid and is a guiding principle to enable the integration and federation of multiple service bus instantiations.”*

– Rob High, SOA Foundation Chief Architect in the SOA Foundation Architecture Whitepaper

SOA Reference Architecture

Model of the Logical Architecture



What is an Enterprise Service Bus?

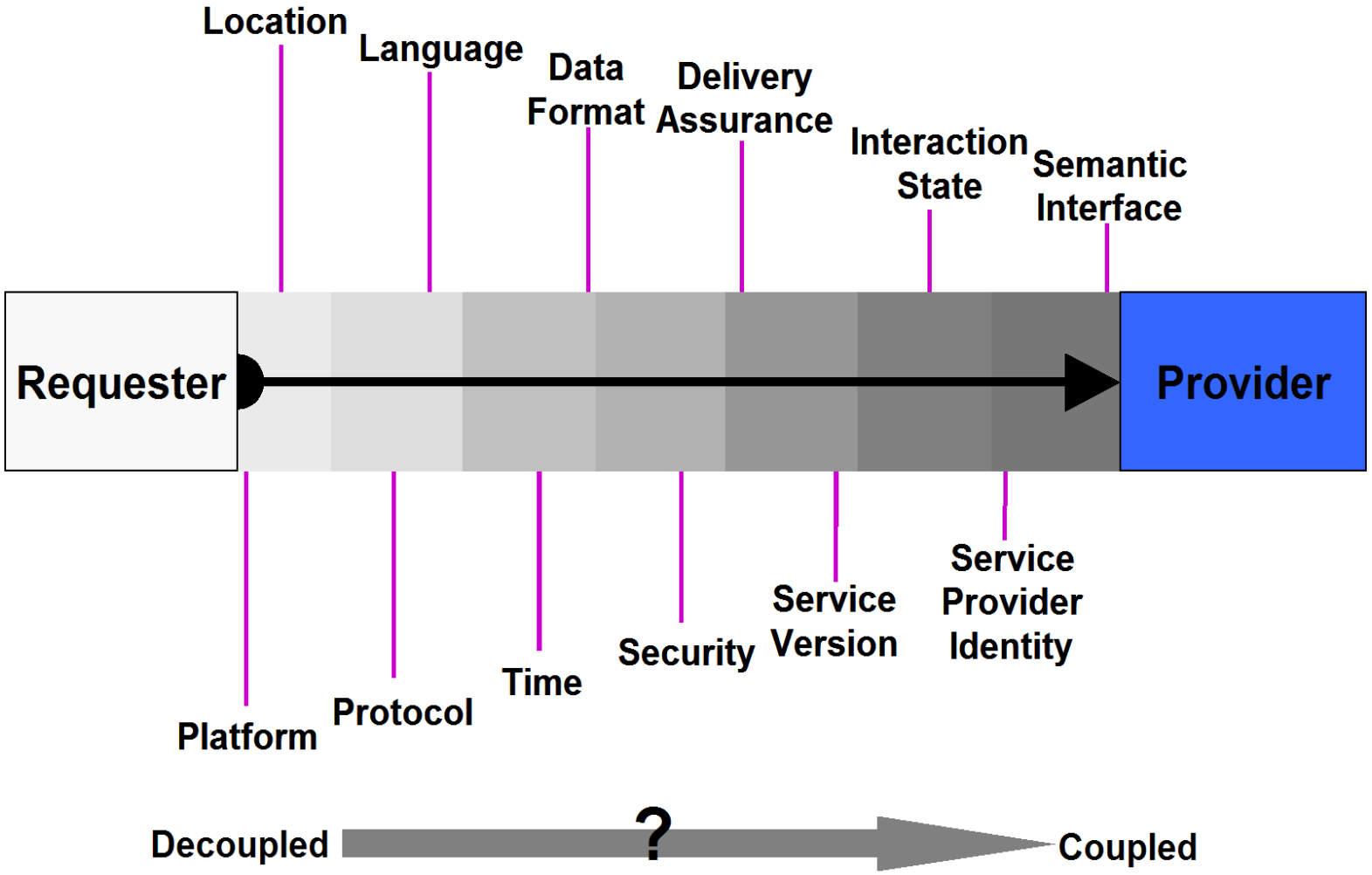
- An ESB enables standards-based integration between loosely-coupled applications and services within and across…
  - ***Services oriented architectures*** – distributed applications are composed of granular re-usable services with well-defined, published and standards-compliant interfaces
  - ***Message driven architectures*** - applications send messages through the ESB to apps
  - ***Event driven architectures*** - applications generate and consume messages anonymously

- An ESB enables **application integration** across different platforms, programming models & messaging standards
  - Underpinning Business Process and managed Business Partner integration

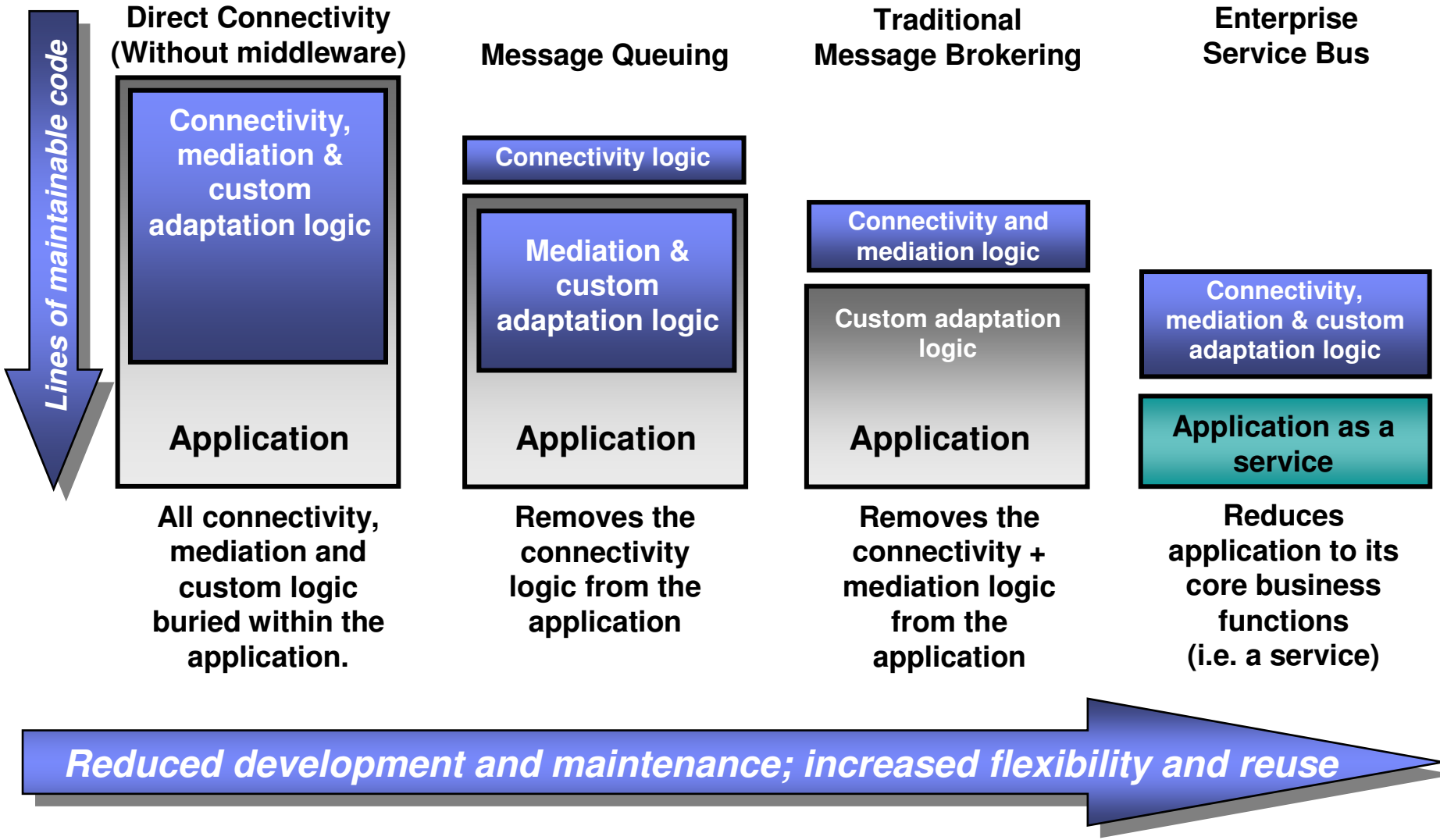
What is loose coupling?

- **Tighter coupling tends to cost more over time**
  - Synchronizing multiple organizations on change
  - Hard to move, hard to scale, hard to distribute, hard to replace
  - Making changes is hard and expensive, or impossible:
    - Knowledge is distributed throughout the code
    - Same people are solving business and infrastructure problems
  - Different parts of the solution are difficult to manage separately
  - Adapting, redeploying updated components without affecting others
- **Looser coupling requires greater investment up front**
  - More design work
  - More implementation work

Loose coupling: aspects of service interaction

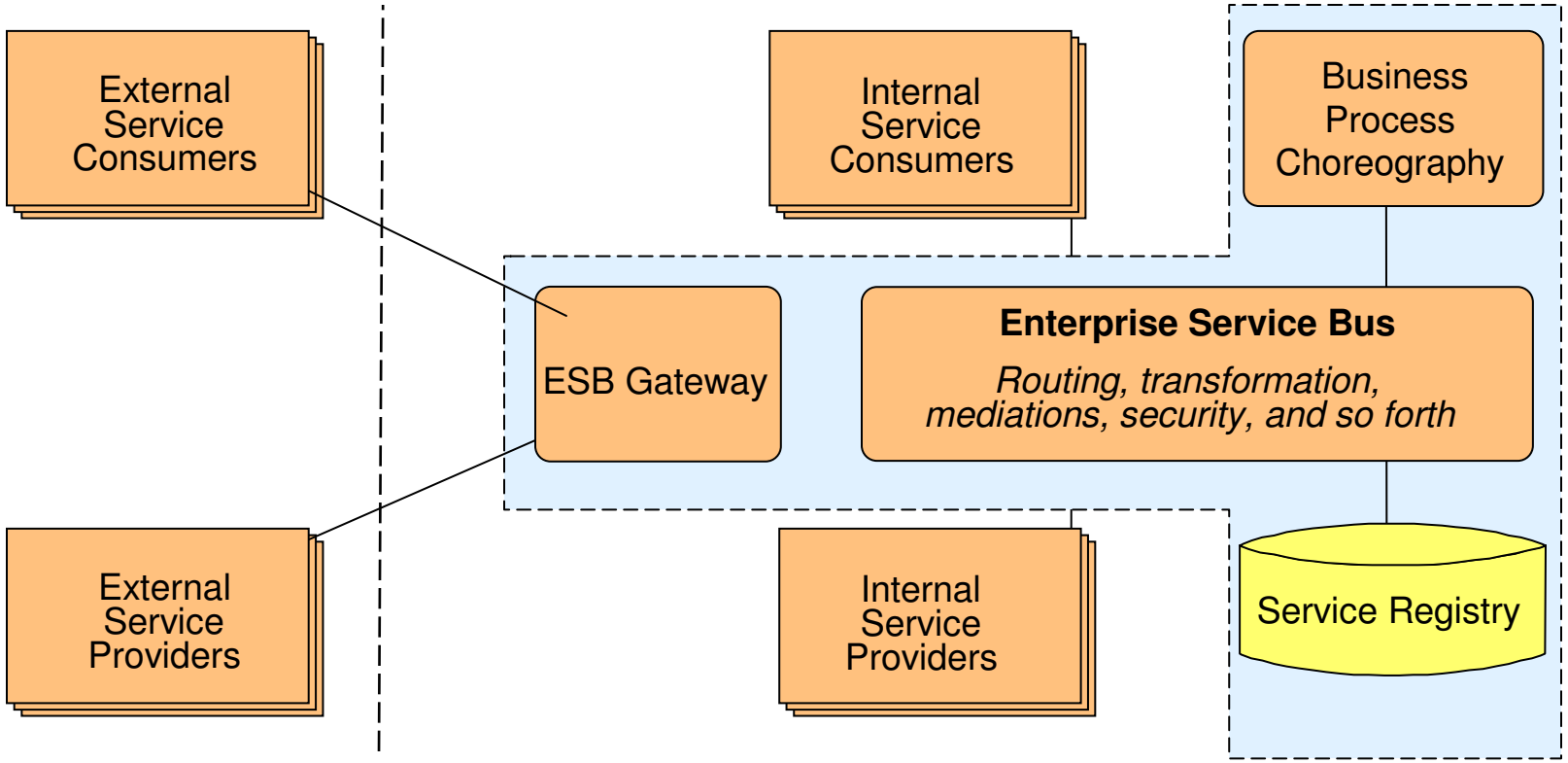


ESB – an evolution of existing technology



The Enterprise Service Bus (ESB) Pattern

The SOA Runtime Pattern Family



Infrastructure components for service-oriented architecture

Enterprise Service Bus Pattern

ESB's need to provide support for…

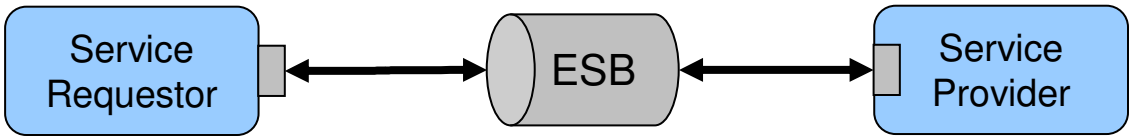
- Large numbers of service interactions in a manageable way
- Web Services and traditional EAI communication standards and technologies
- A variety of interaction styles such as synchronous request/response, messaging, publish/subscribe, and events
- Advanced service interaction capability, for example, transactions, store and forward, infrastructure services, security, and quality of service
- Service routing and substitution, protocol transformations, and other message processing

Implementations

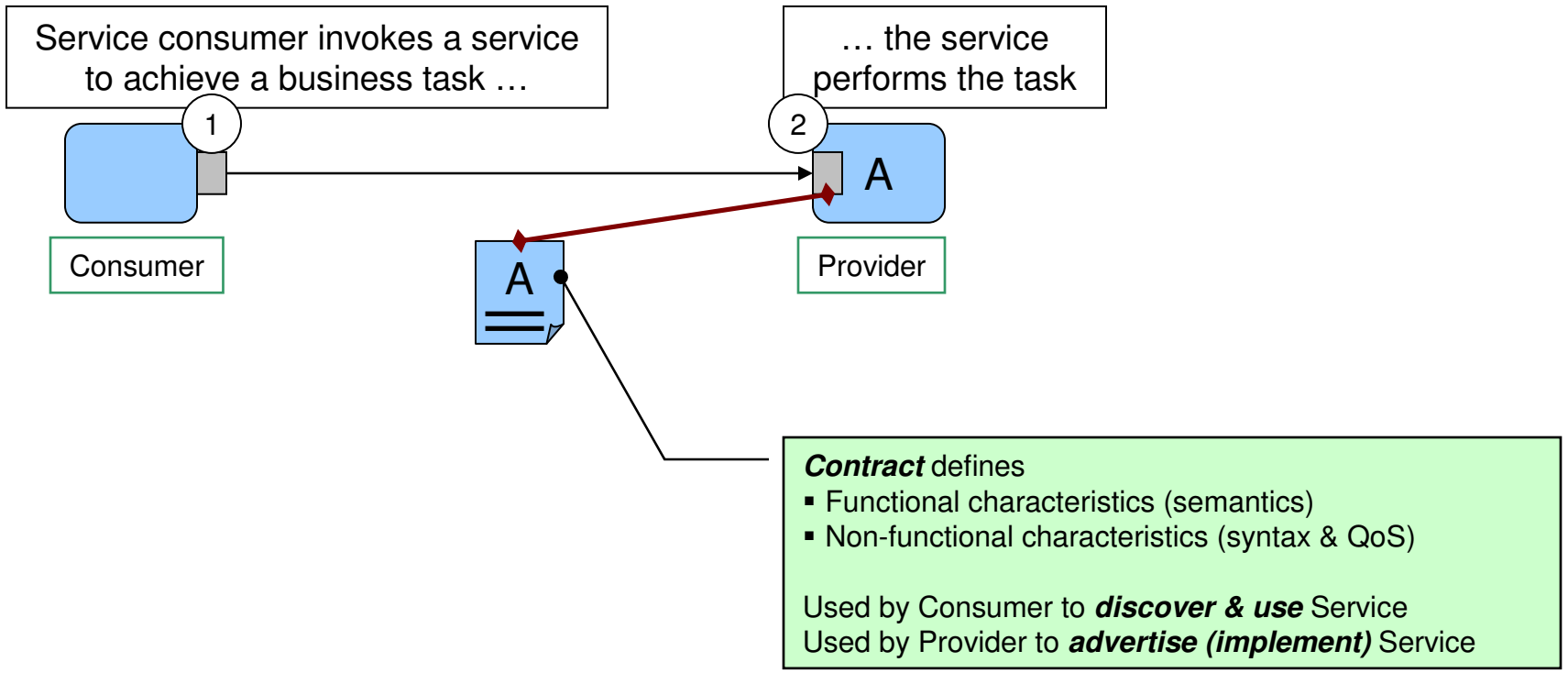
- EAI and messaging technologies
- "Gateway" technologies
- ESB products

Core ESB Principal – Virtual Service Provider Pattern

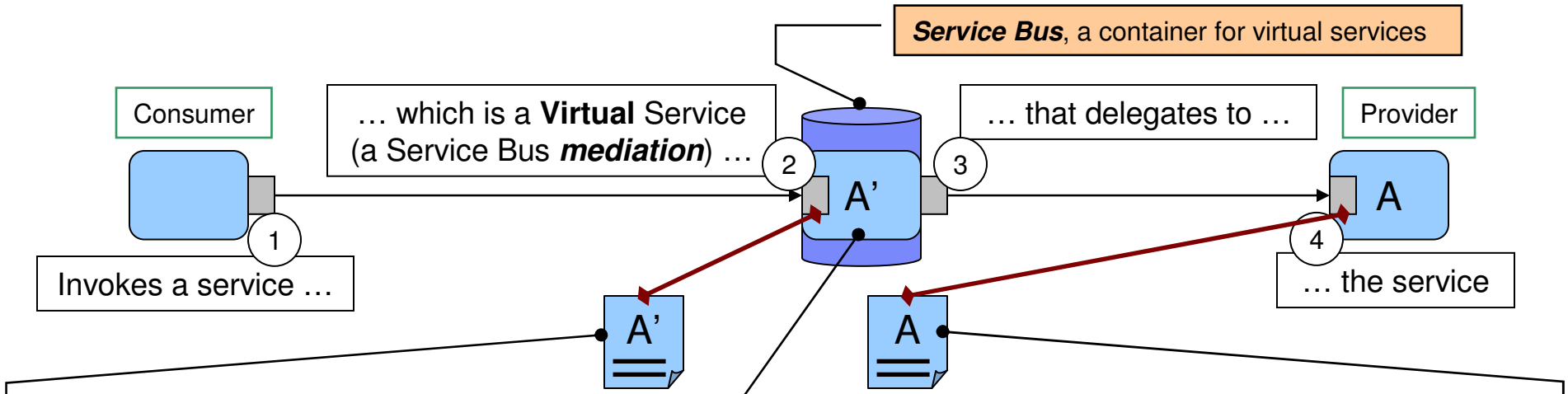
- **ESB inter-connects requestor and provider**
  - Interactions are *decoupled*
  - Supports key SOA principle – *separation of concerns*
- **ESB provides *Service virtualization* of**
  - Location and identity
  - Interaction pattern and protocol
  - Interface
- **ESB also enables Aspect-oriented connectivity or mediation**
  - Security
  - Management
  - …



Direct Service Connectivity in SOA



Mediated Service Connectivity in SOA



Consumer Contract defines
- Functional characteristics
- Non-functional characteristics

Used by Consumer to ***discover & use*** "Service"
Used by Domain Owner to ***advertise*** "Service"
Used by Bus Owner to ***implement*** Virtual Service

Service Contract (may be informal) defines, for Provider
- Functional characteristics
- Non-functional characteristics

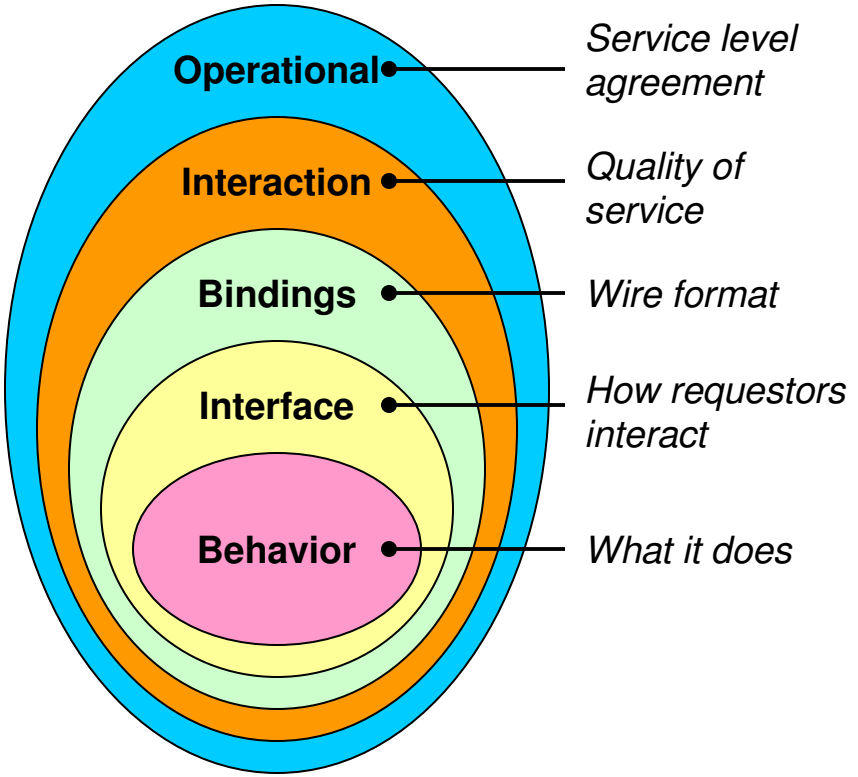
Used by Provider to ***advertise (& implement)*** Service
Used by Bus Owner to ***implement*** Virtual Service

Virtual Service/Mediation
- Enforces "policy" regarding interaction
  - Management, security, log, …
- Is in effect a governance control point for service access
- ***Is*** the service from a Consumer (business) perspective

Service Virtualization

Service Contract

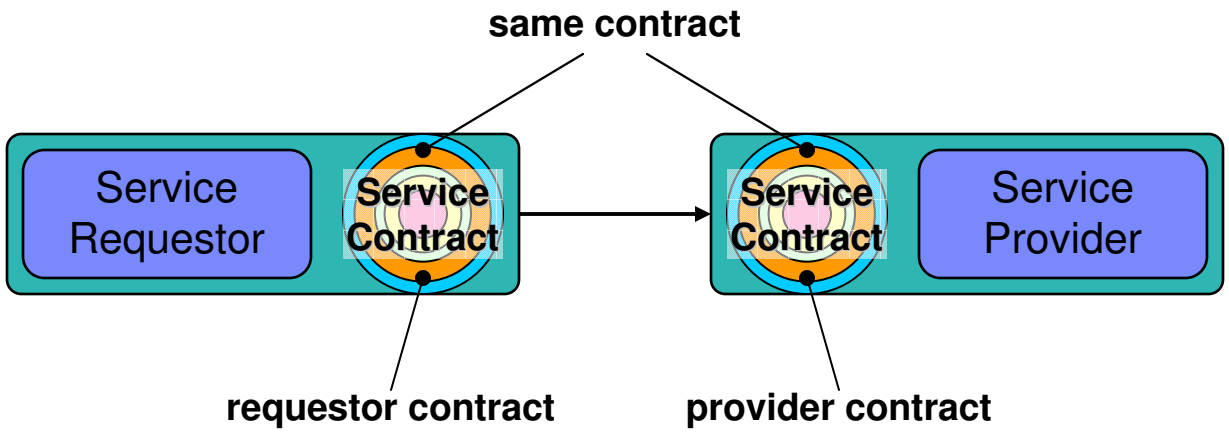
- Documents everything requestors and providers must agree upon
  - Behavior
    - What it does
  - Interface
    - How requestors interact
  - Bindings
    - Wire format
  - Interaction
    - Quality of service (QoS)
  - Operational
    - Service level agreement (SLA)
- All externally visible characteristics of a service



Service Contract

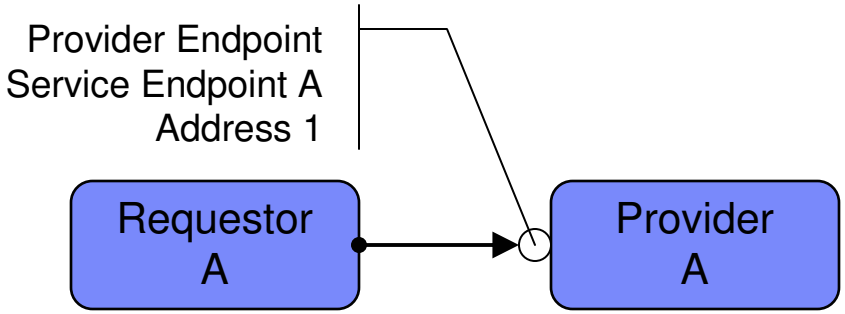
Service Connectivity

- Connects requestors and providers
  - Enables requestors to invoke providers
- Must agree on service contract
  - Requestor contract - the one the requestor delegates to
  - Provider contract – the one the provider implements
- Integration of contracts which don't match requires mediation



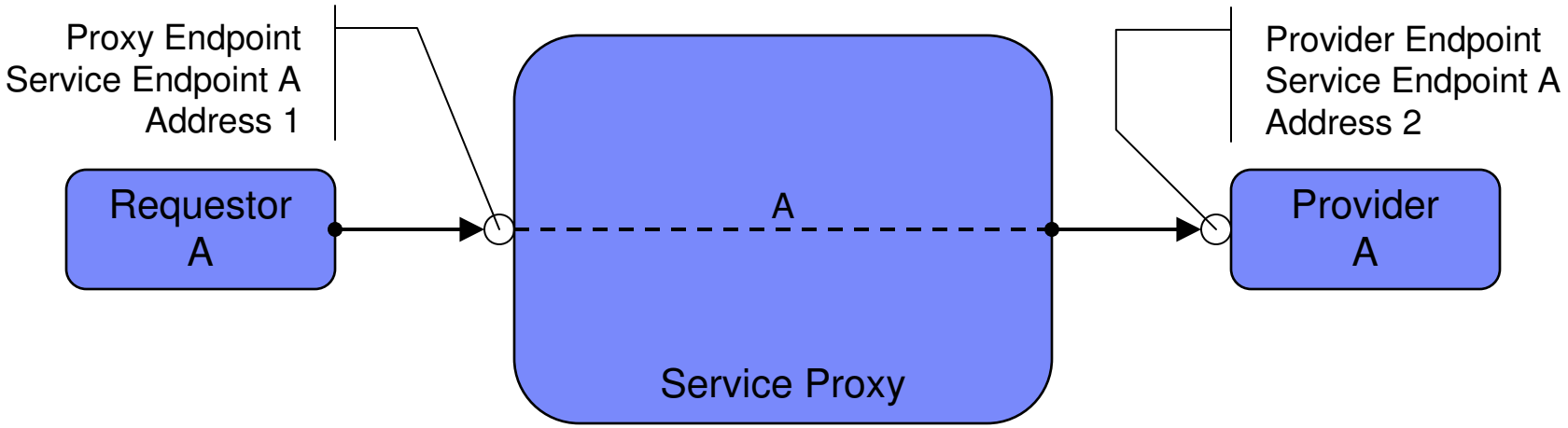
Direct Service Connectivity

- Requestor gets/knows endpoint address of the provider
  - Uses that to invoke the provider
  - Standard Web services UDDI approach
- Consequences of this design
  - Requestor can only use a single provider
  - Requestor contract and provider contracts must match
  - Unreliable: busy provider, flakey network
  - Tighter coupling: Changes in provider affect requestor



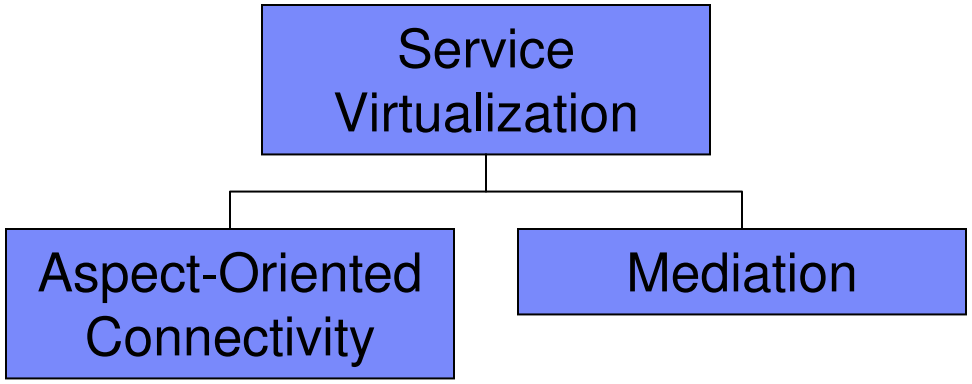
Indirect Service Connectivity

- Service proxy (i.e. Proxy pattern) between requestor and provider
  - Service virtualization – separate proxy and provider endpoints
- Consequences of this design
  - Requestor can use multiple providers
  - Requestor and provider contracts do not have to match
  - Increases reliability of invocations
  - Looser coupling: Changes in provider affect proxy, not requestor



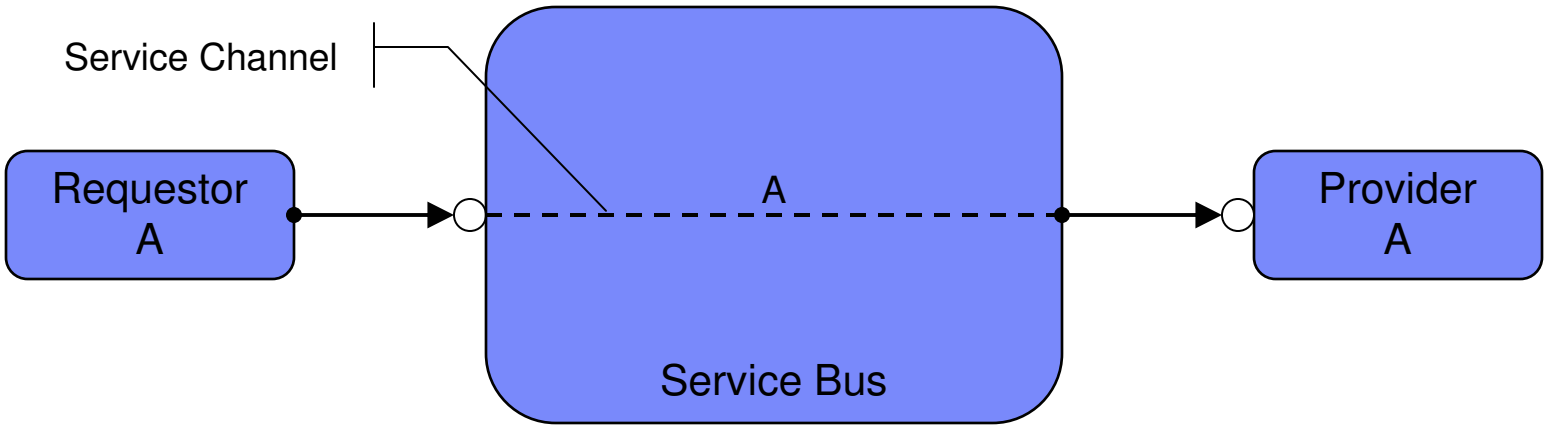
Service Virtualization

- By-product of indirect service connectivity
  - Provided by the service proxy
  - Result of the proxy's endpoint being different from provider's endpoint
- Provides opportunity for aspect-oriented connectivity and/or mediation
  - Aspect-oriented connectivity to improve the quality of the connectivity
  - Mediation to bridge the differences between service contracts



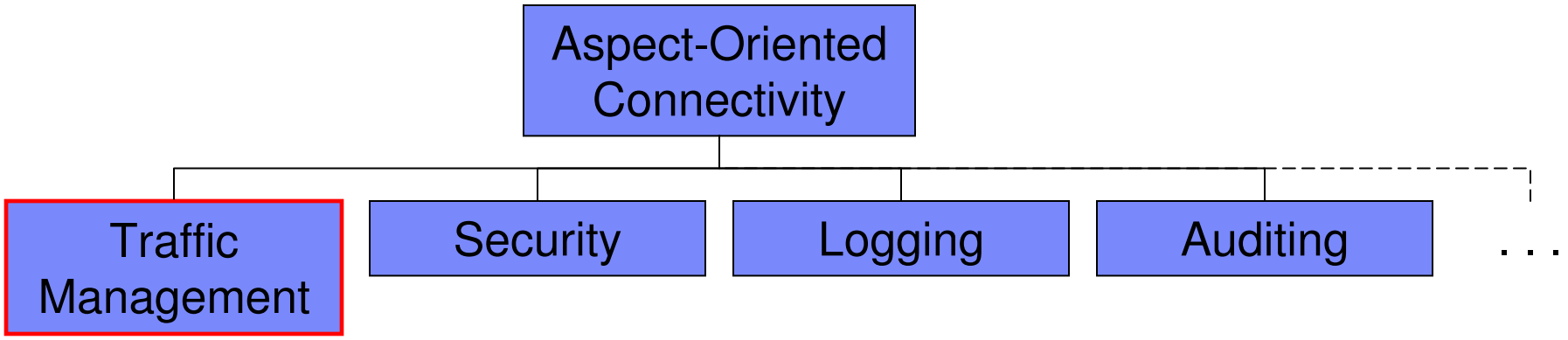
Service Channel

- Implements a service endpoint within a service proxy
  - Connects all requestors of a service to all providers of that service
- Requestor invokes proxy's endpoint; proxy invokes provider's endpoint
  - Structure within the proxy that manages the invocations
- Opportunity to increase the quality of the connection
  - Aspect-oriented connectivity



Aspect-Oriented Connectivity

- Cross-cutting aspects of connectivity
- Can be added or removed without affecting requestors and providers
- Applied per service channel
  - Zero or more aspects can be combined
- Various kinds of aspect-oriented connectivity
  - Traffic management, security, logging, auditing, etc.

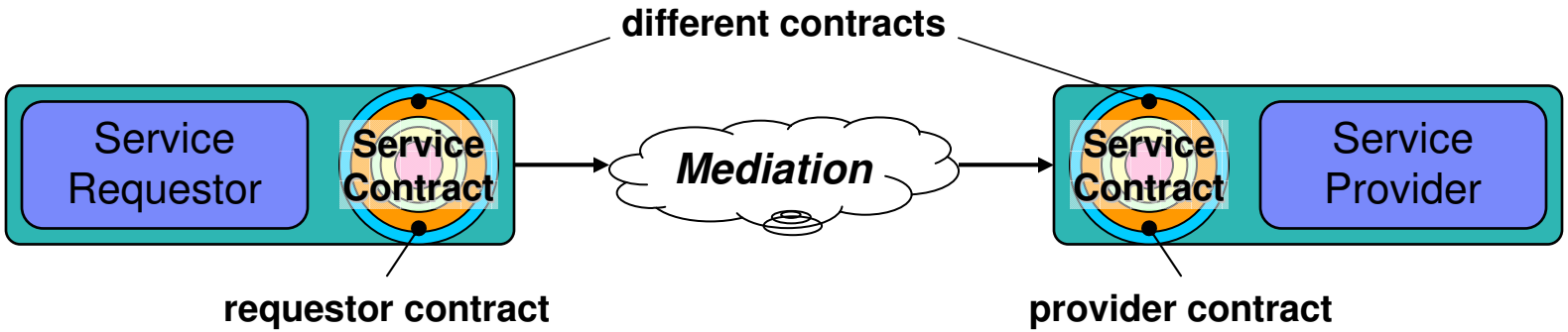


Traffic Management Patterns

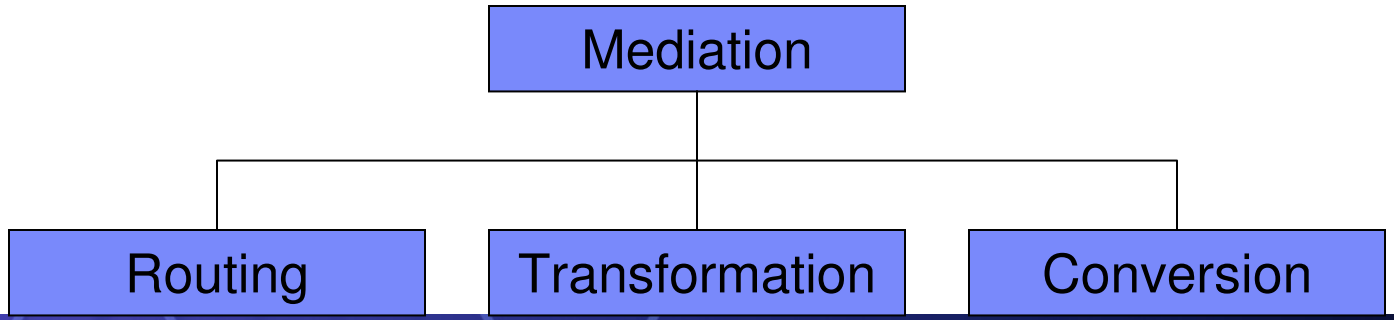
- Queuing
  - Stores invocation messages until they can be transmitted
  - Enables asynchronous invocation
- Prioritization
  - Process more important invocations before lesser ones
- Load distribution
  - Spread invocations across equivalent target endpoints
- Service invocation throttling
  - Rejects invocations past a specified frequency threshold
- Provider invocation retry
  - Reattempt a failed invocation until it succeeds

Mediation

- Mediation
  - Used when requestor contract and provider contract do not match

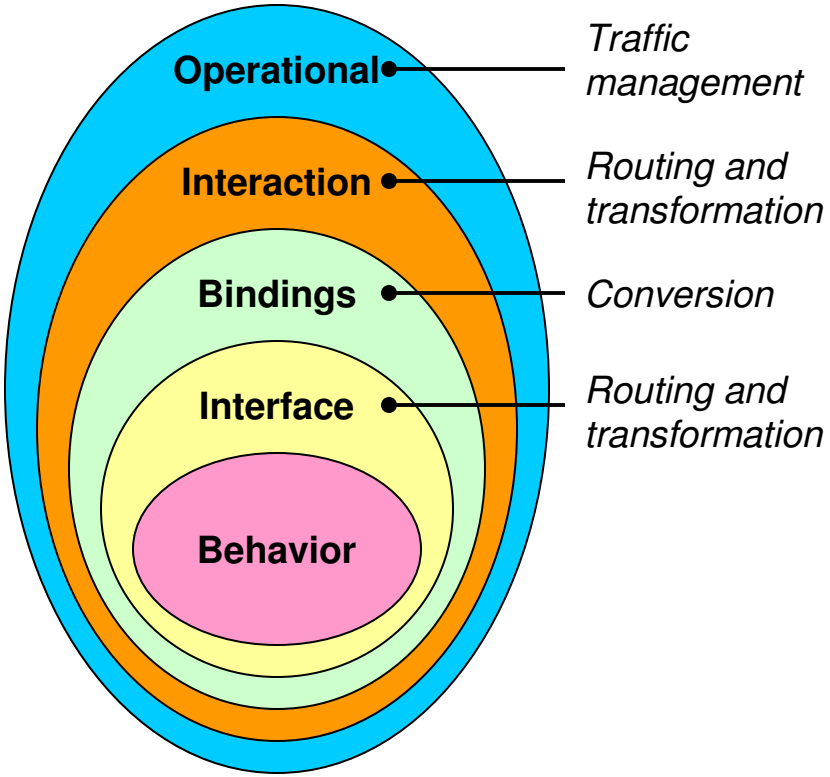


- Three kinds of mediation
  - Routing, transformation, and conversion



Service Contracts and Mediation

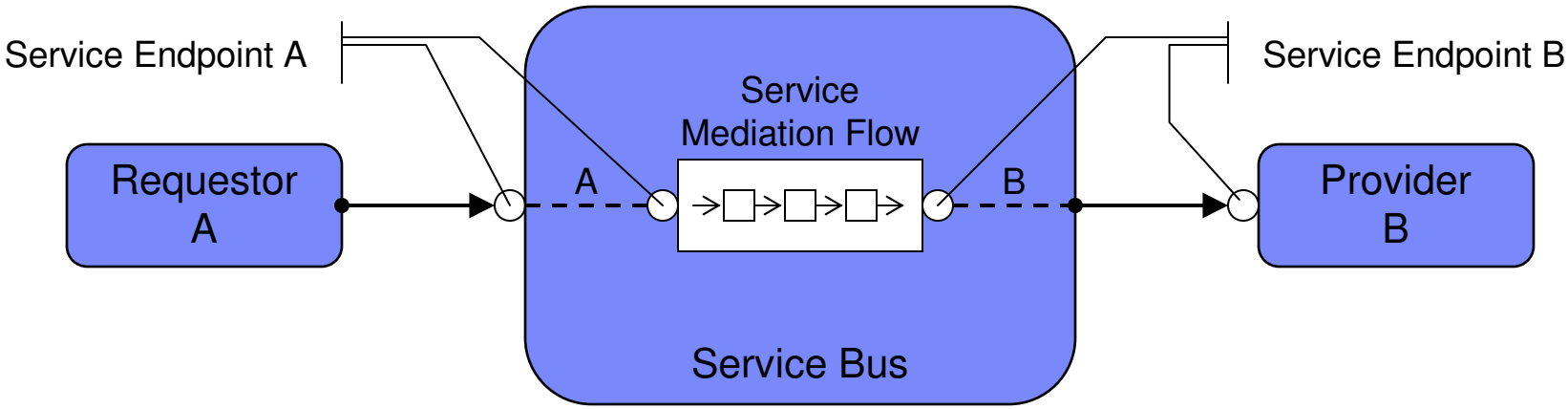
- Service connectivity connects service contracts
  - Differences between contracts bridged by mediation
- Different parts of contract bridged by different kinds of mediation
  - Behavior must be equivalent
  - Interface – Routing and transformation
  - Bindings – Conversion
  - Interaction – Routing and transformation
  - Operational – Traffic Management



Service Contract and Mediation

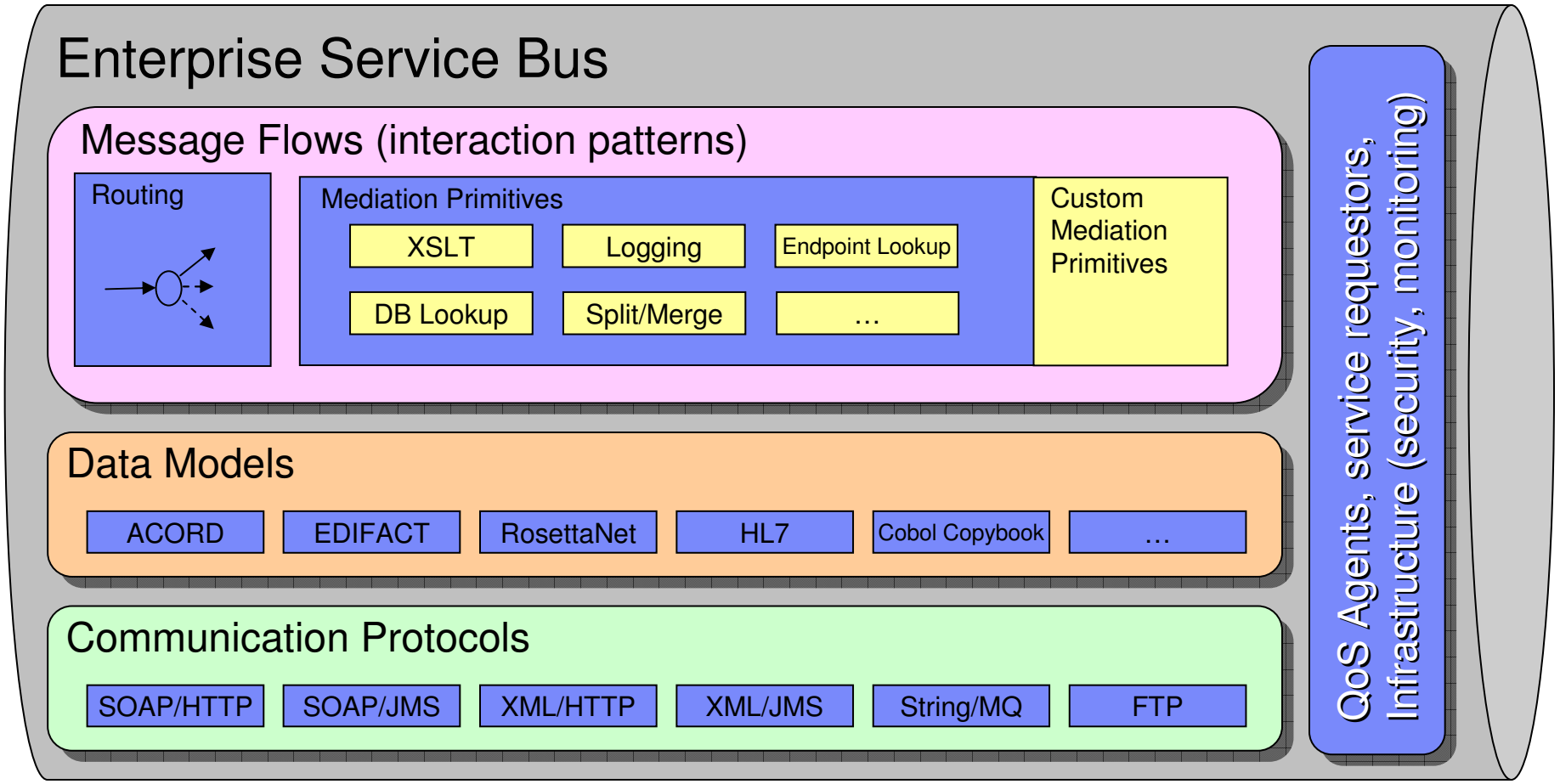
Mediation vs. Traffic Management

- Change the invocation during transmission
  - Proxy endpoint and provider endpoint have different contracts
  - Requestor contract and provider contract differ
  - Mediation bridges the differences
- Traffic management changes the handling of the invocation
  - Mediation changes the content and destination of the invocation



What is inside an Enterprise Service Bus?

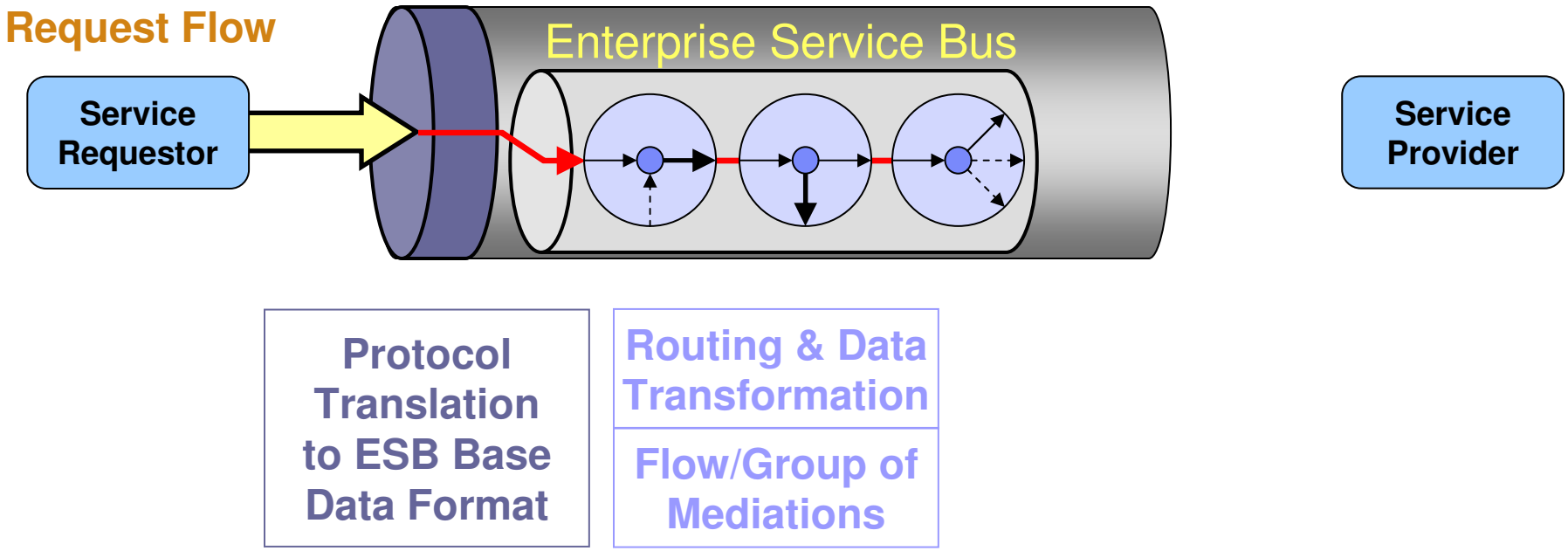
Enterprise Service Bus Reference Architecture



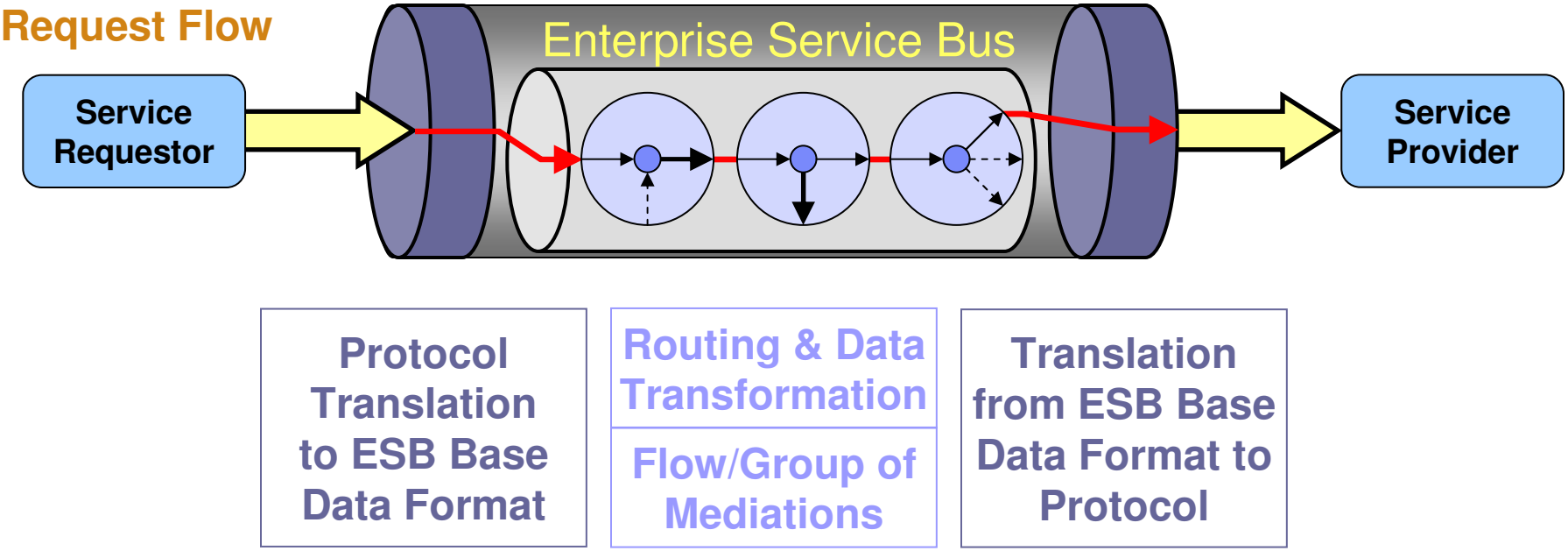
ESB Request Flow Decomposition

ESB Request Flow Decomposition

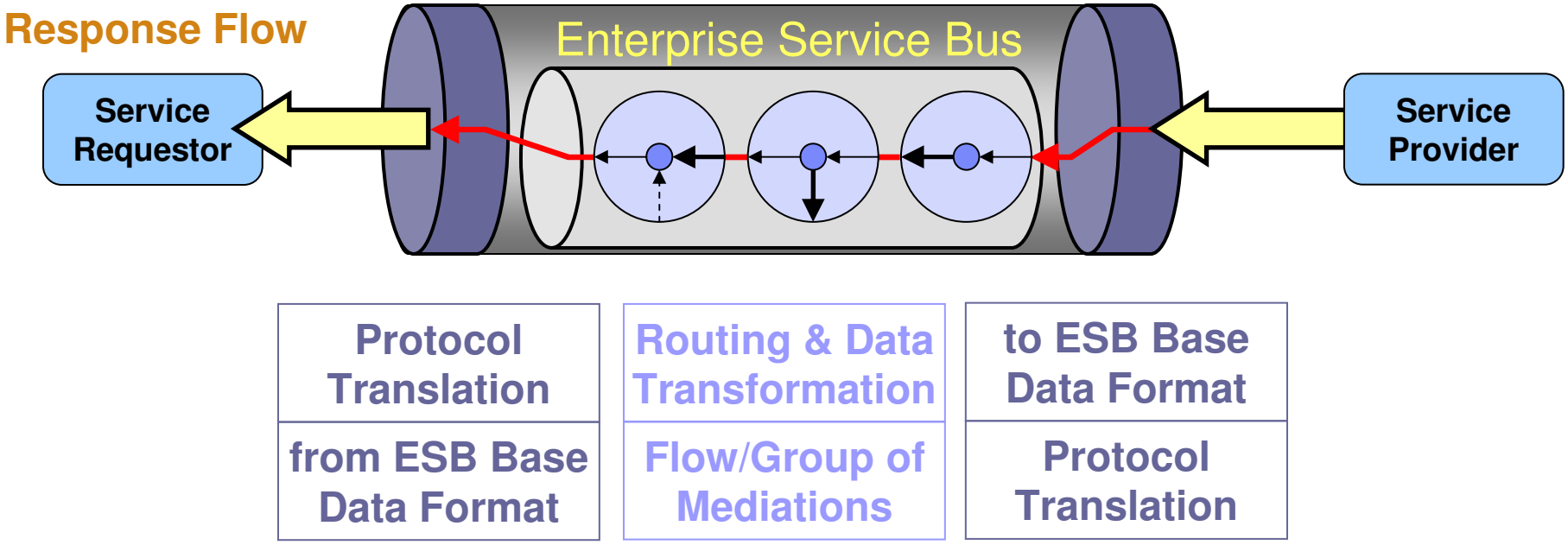
ESB Request Flow Decomposition



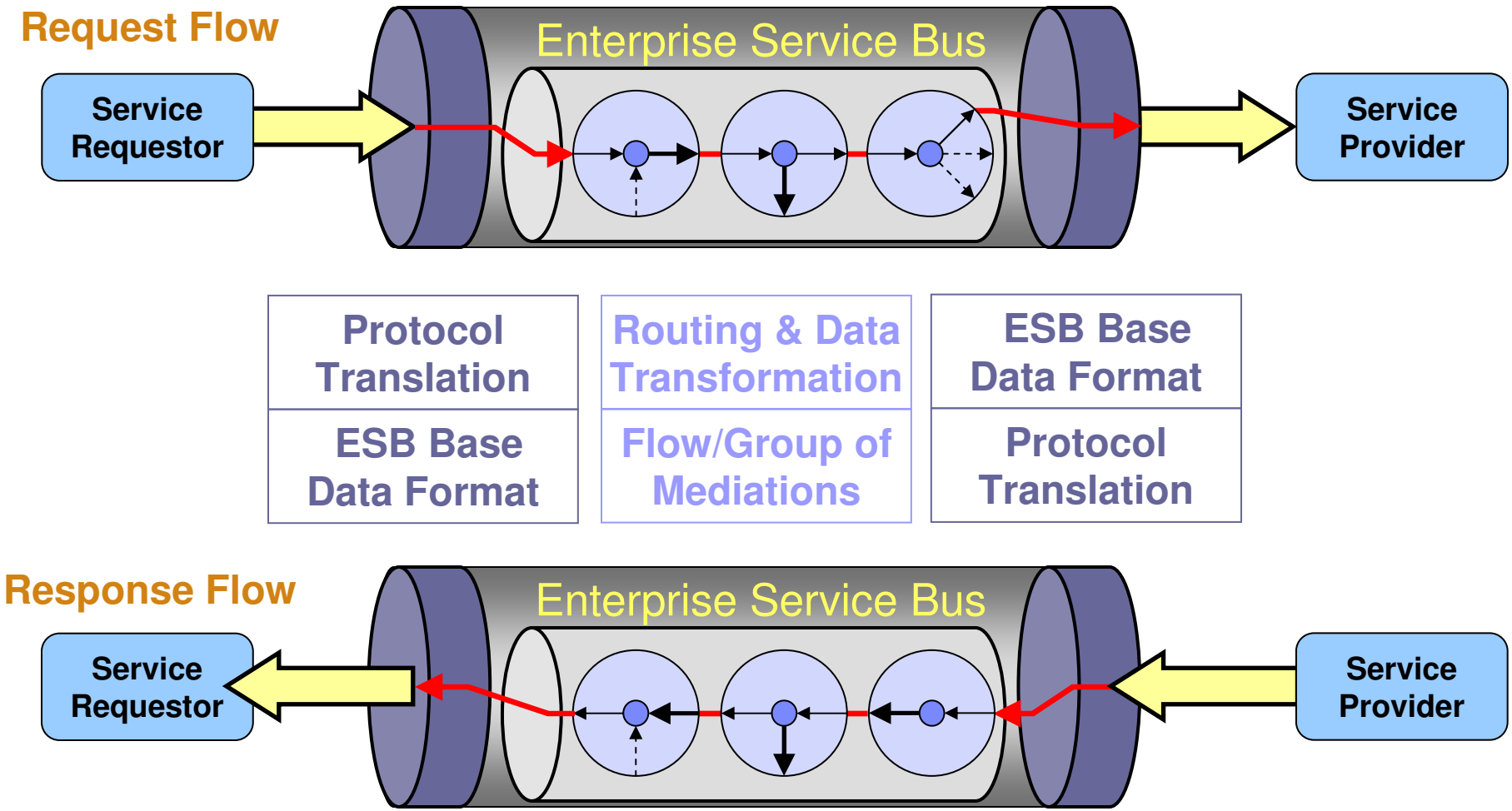
ESB Request Flow Decomposition



ESB Response Flow Decomposition



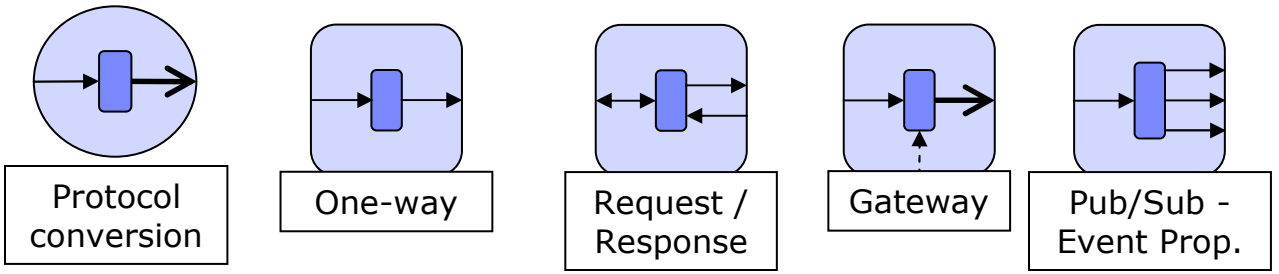
ESB Request-Response Flow Review



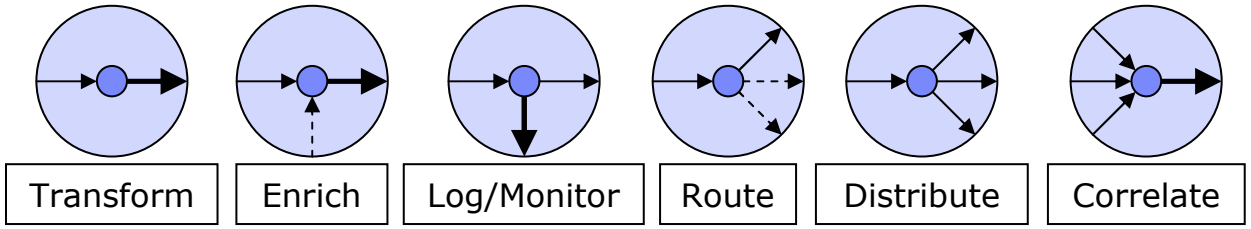
Typical Enterprise Service Bus Patterns?

Mediation Patterns - Examples

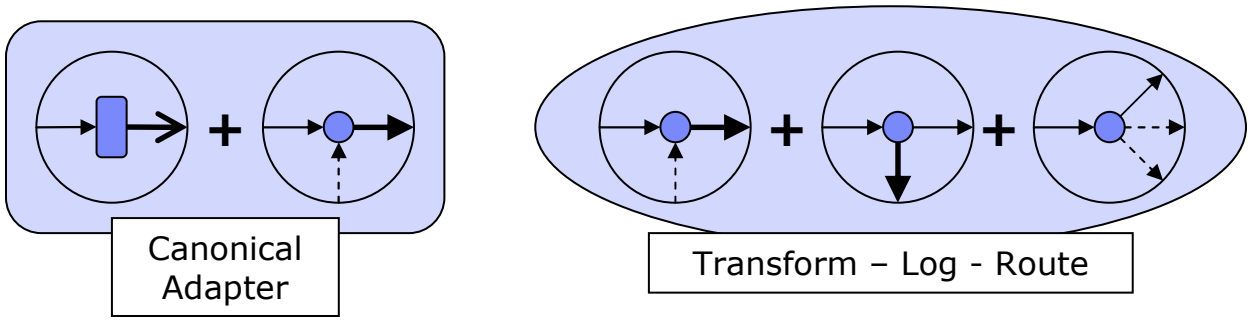
Conversion & Interaction Patterns



Transformation & Routing Patterns



Composite Patterns Examples



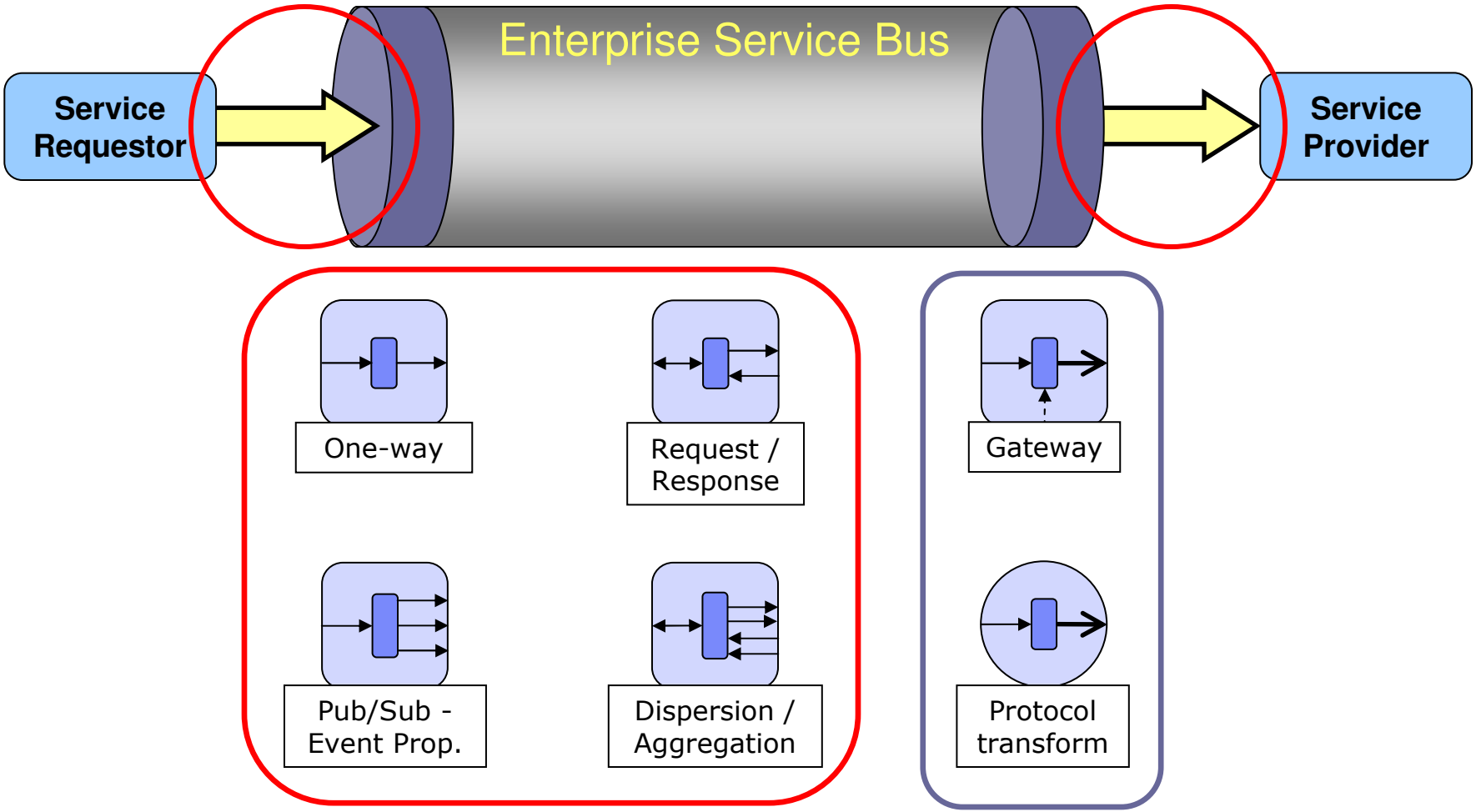
ESB Transport Protocols and Conversion

- Basic connectivity supported via one or more transport protocols
  - Dependent on underlying communication fabric(s)
- Conversion inherent with support for more than one transport protocol
- Enables
  - Virtualization of *interaction protocol*
  - Aspects of QoS (e.g., reliable delivery, transactions)
- Typical requirements
  - HTTP (SOAP/HTTP, XML/HTTP)
  - MQ (SOAP/JMS/MQ, XML/MQ, text/MQ, …)
  - Adapters (legacy, EIS)

ESB Interaction Patterns and Enhanced Routing

- Fundamental interaction patterns based on underlying communications fabric(s)
  - Point-to-point
    - Request/reply (synchronous and asynchronous)
    - One way
  - Pub/Sub
- Enhanced (dynamic) routing of messages
  - Via mediation patterns
- Enables
  - Virtualization of *location* and *identity*
  - Aspects of QoS (e.g., SLA, failover)
- Typical routing requirements
  - Round robin
  - Content based
  - Service registry driven

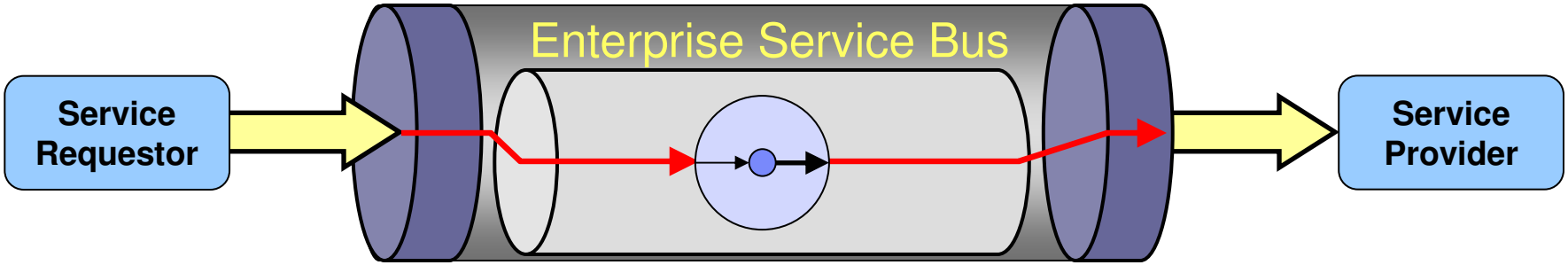
Mediation Patterns – Interaction Patterns



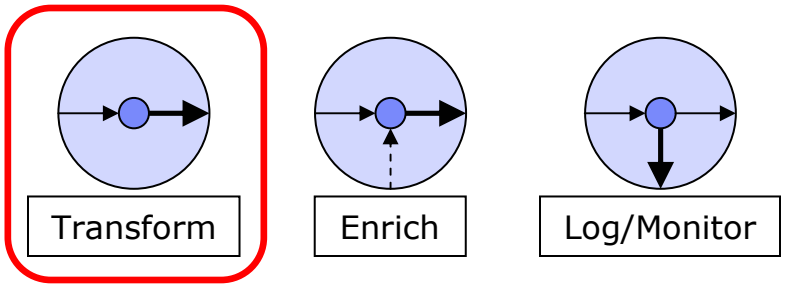
ESB Mediation Patterns and Message Processing

- Allow manipulation of messages during a message flow
  - Provided by a mediation framework enabling pattern construction
- Enhance the basic interaction patterns, e.g.,
  - Message enrichment
  - Monitoring and logging
  - Registry, security and management
  - Distribution/aggregation
- Enables
  - Aspects of QoS (security and management)
- Typical requirements beyond routing and transformation
  - Retry
  - Recipient list
  - Custom

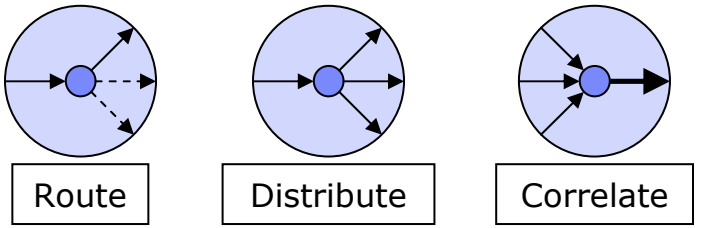
Transformation Pattern - Transform



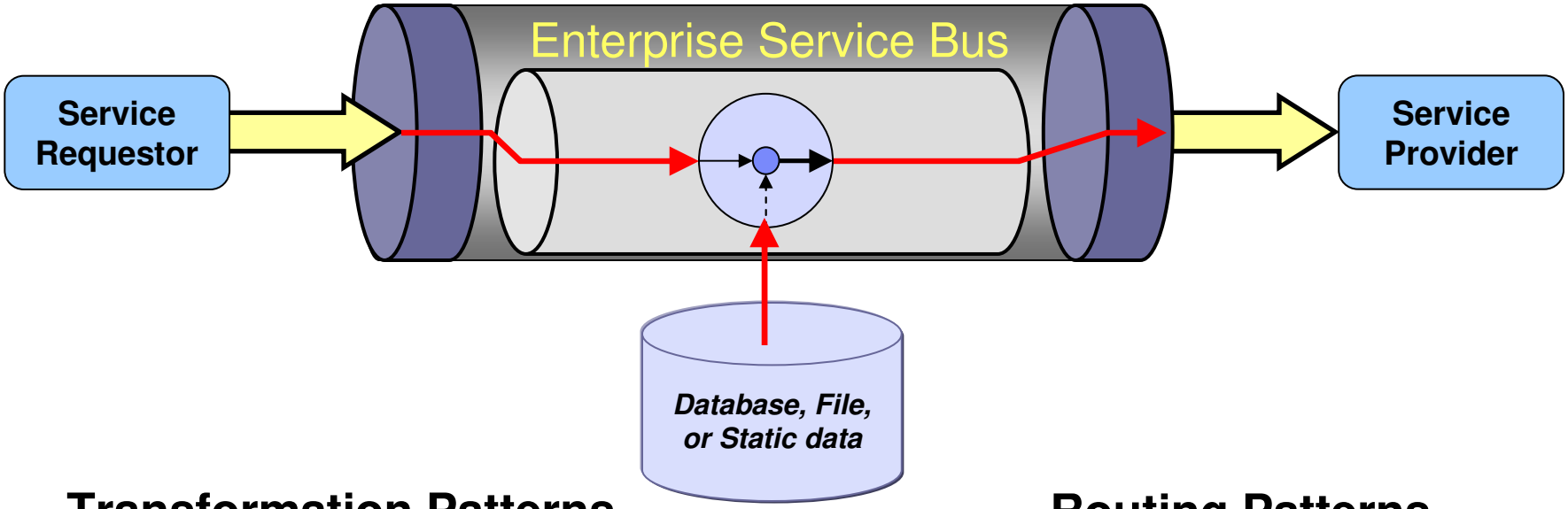
Transformation Patterns



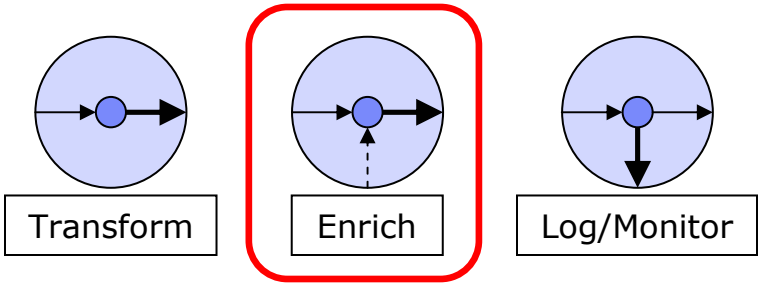
Routing Patterns



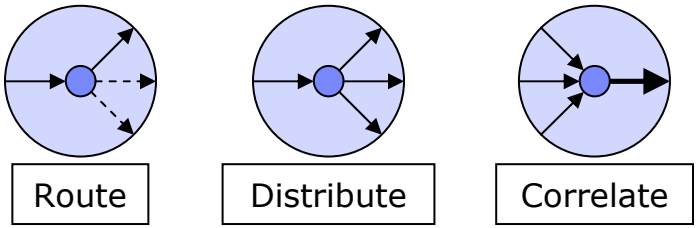
Transformation Pattern - Enrich or Augment



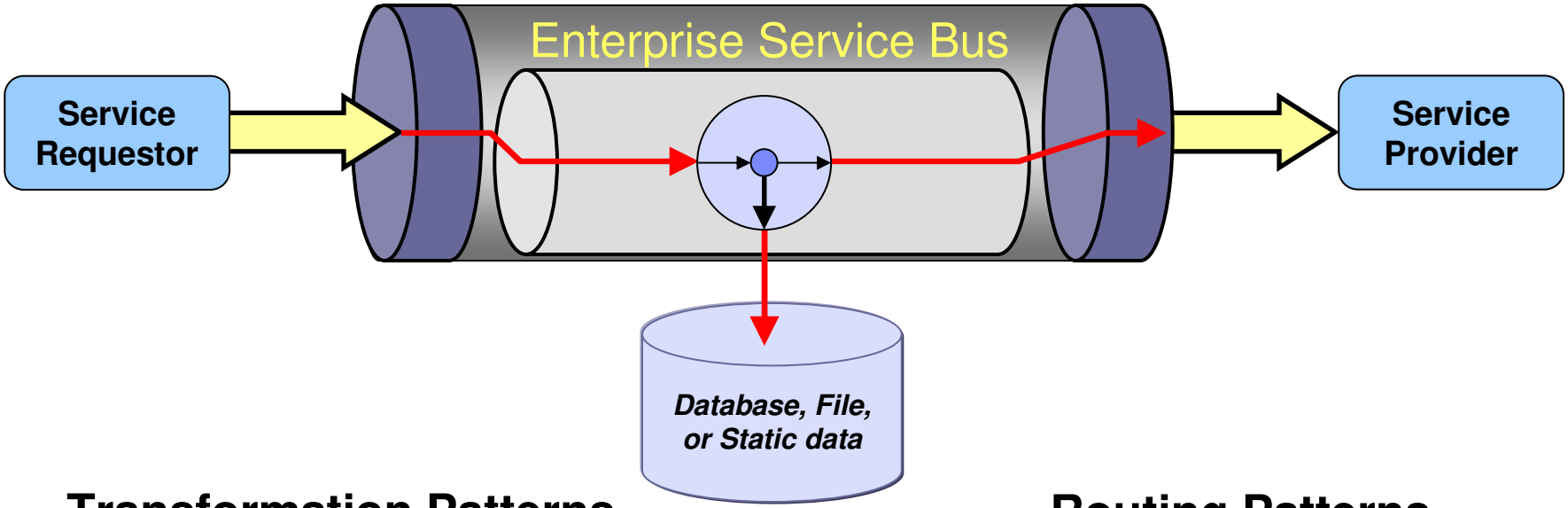
Transformation Patterns



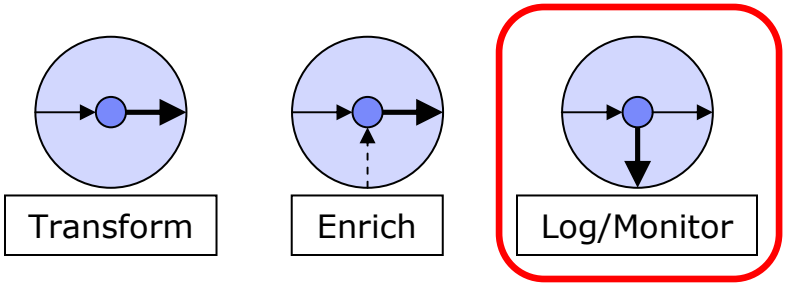
Routing Patterns



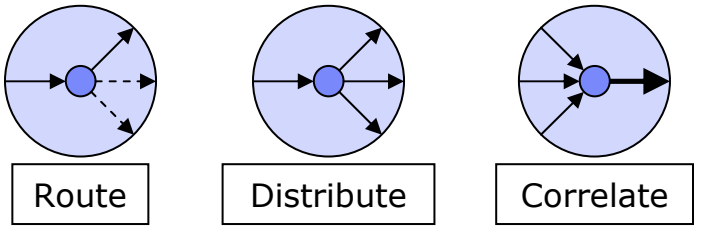
Transformation Pattern – Log or Monitor



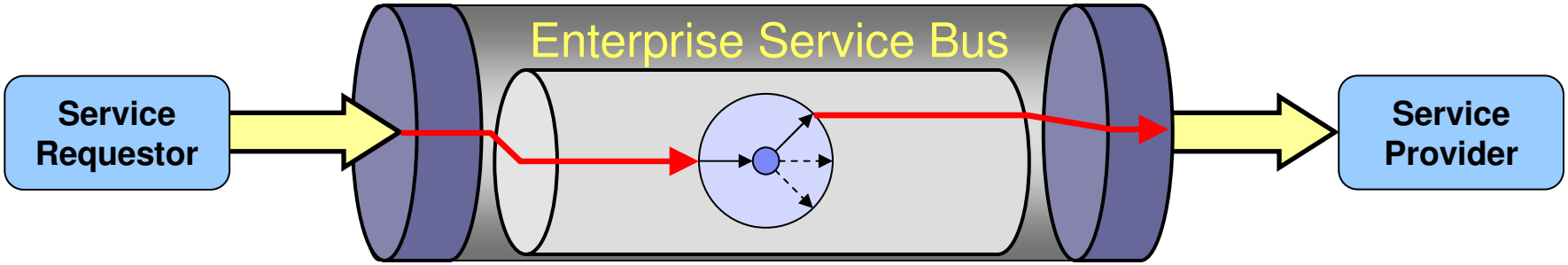
Transformation Patterns



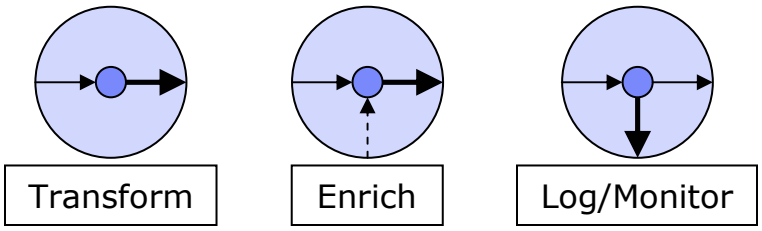
Routing Patterns



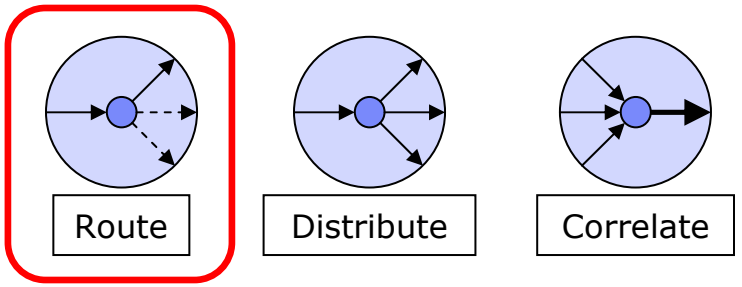
Routing Pattern - Route



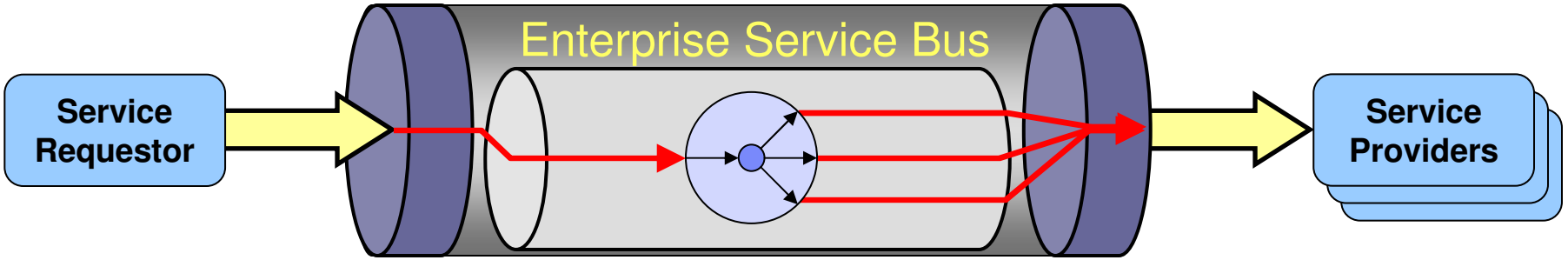
Transformation Patterns



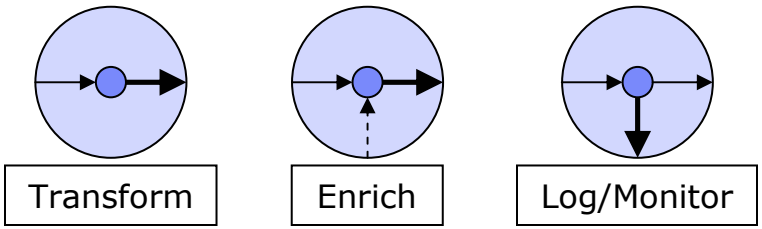
Routing Patterns



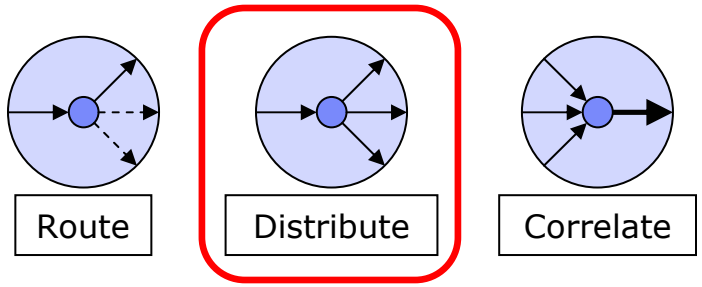
Routing Pattern - Distribute



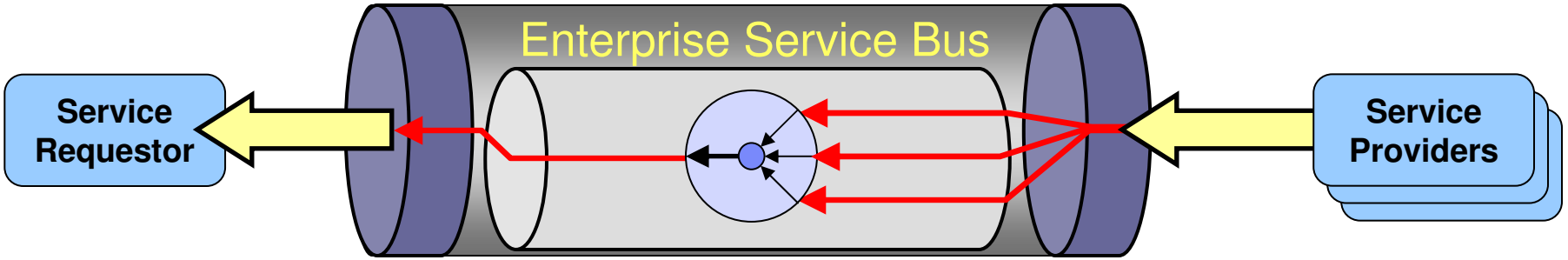
Transformation Patterns



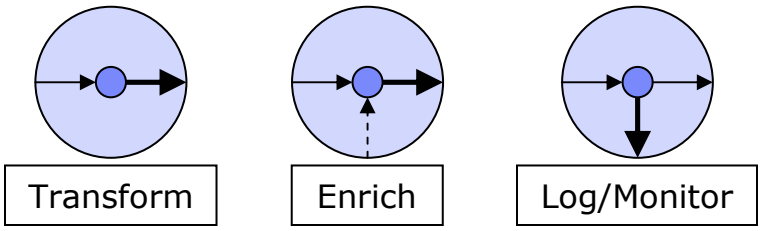
Routing Patterns



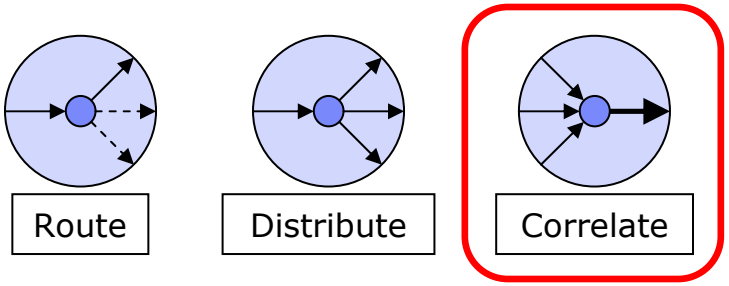
Routing Pattern - Correlate



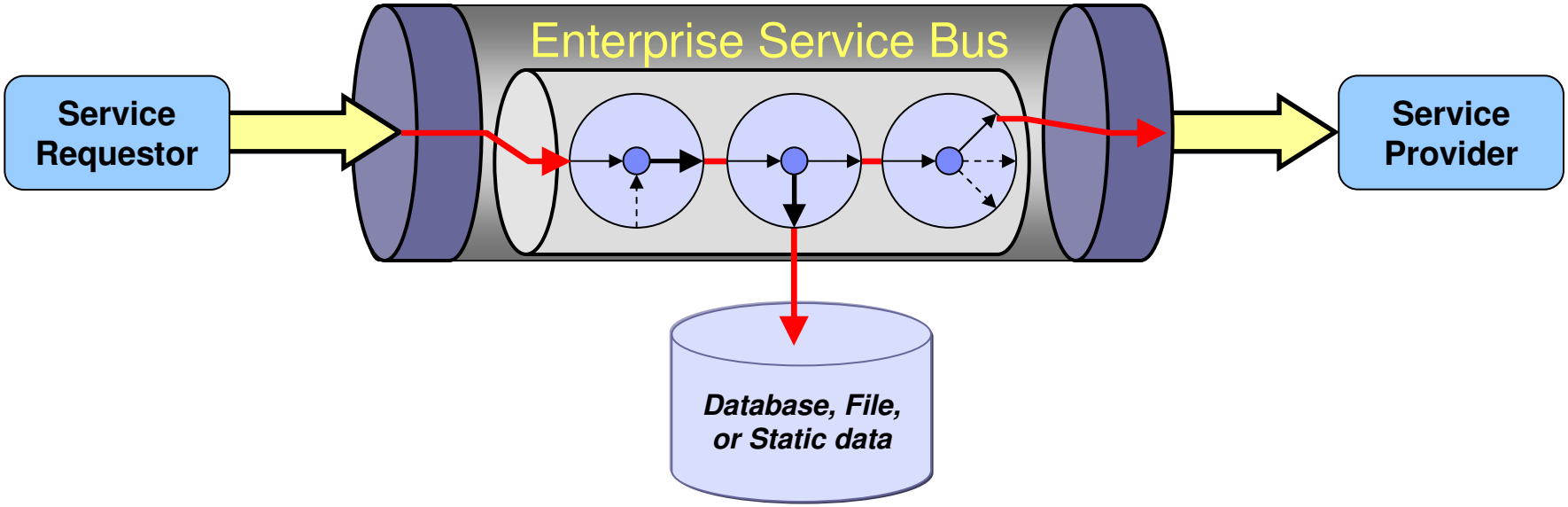
Transformation Patterns

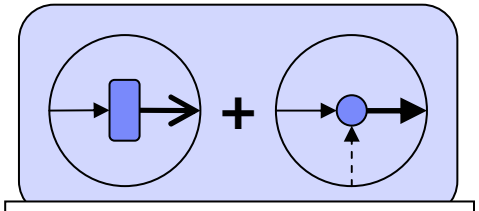


Routing Patterns

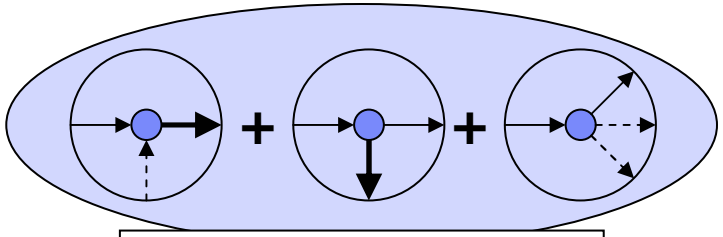


Mediation Patterns – Common Composite Pattern Example





Canonical Adapter
Protocol+Data Transform



Transform – Log - Route

Review

- ✓ **What is an Enterprise Service Bus?**
- ✓ **The Enterprise Service Bus Pattern**
- ✓ **Service Virtualization**
- ✓ **What's inside an Enterprise Service Bus?**