IBM Software Group

# Enterprise Service Bus Patterns

Rick Robinson
Architecture Services, EMEA WebSphere Services
rick_robinson@uk.ibm.com

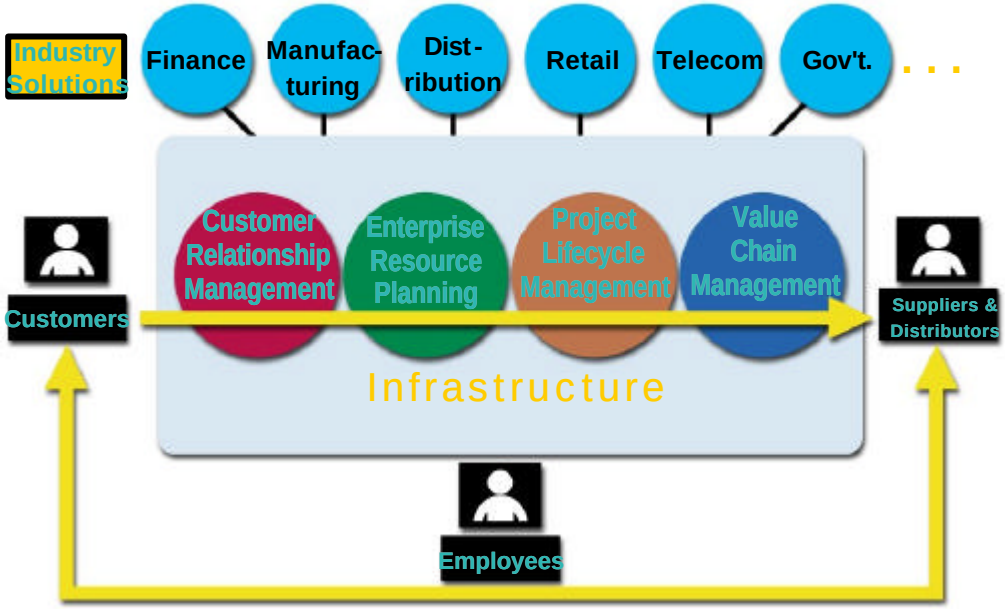
## Agenda

- Why am I here to talk about the Enterprise Service Bus?
  - If we want to achieve business flexibility, intermediaries such as the ESB will be crucial components of a Service Oriented Architecture
  - If we ask what the “loose coupling” specified by Service Oriented Architecture really means, we drive infrastructure requirements.
  - If we want to implement SOA in a *manageable* way, we drive infrastructure requirements.

IBM Software Group

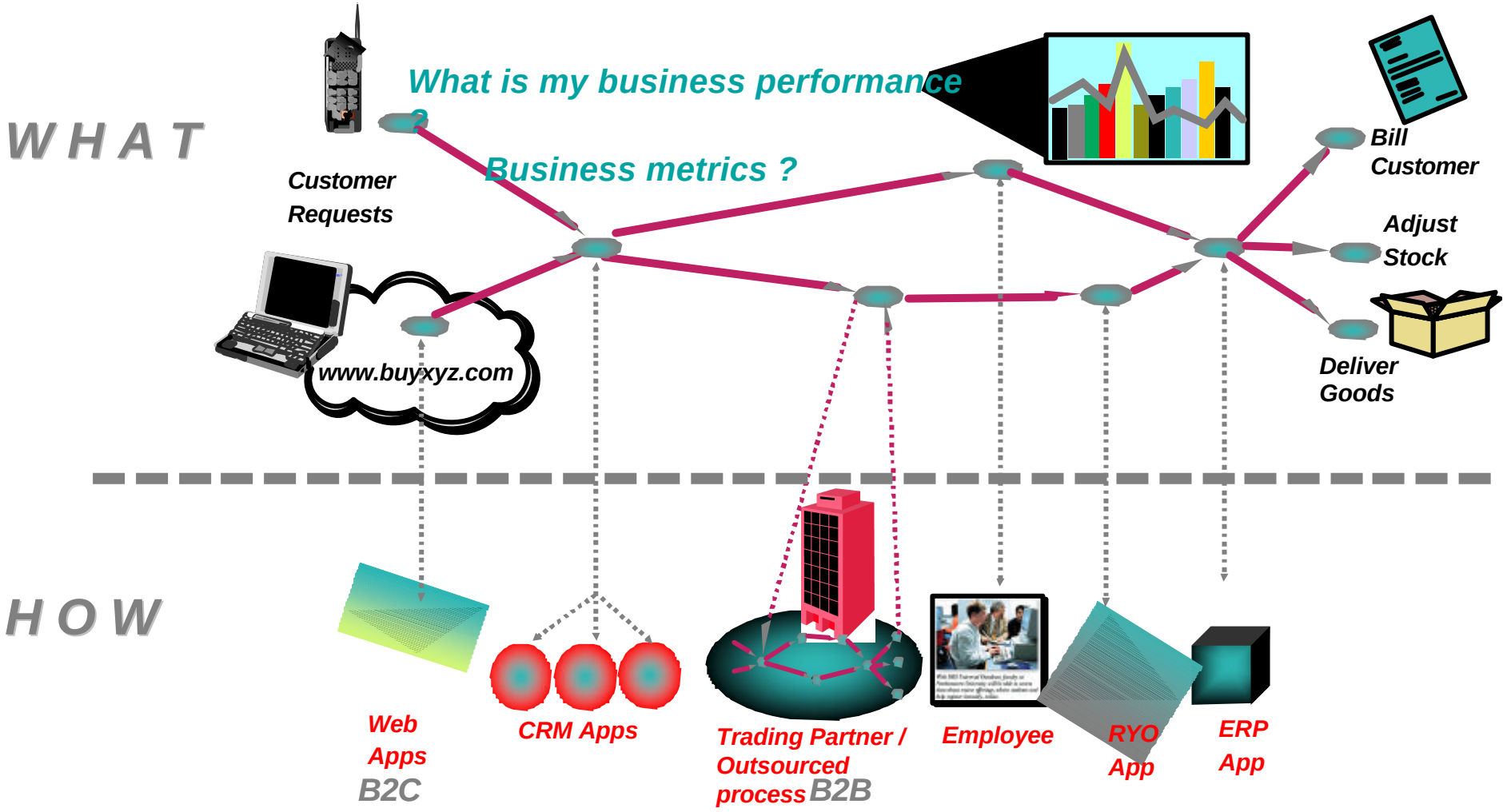
# What is Service Oriented Architecture and why do I care?

# Business are faced with some rather familiar problems ...




- Reduce cycle time and costs for external business processes
  - Move from manual transactions with suppliers towards automated transactions
  - Facilitate flexible dealings with partners with minimal process or IT impact
- Support an agile business model
  - The marketplace is changing - businesses need to change too
  - Many existing IT systems are inhibitors to change: complex and inflexible
  - Existing *integrations* can be inhibitors to change: multiple technologies, point-to-point integration, inflexible models
- Drive down cost
  - Eliminate duplicate systems
  - Re-use, don't re-build
  - Simplify skills base
- Integrate across the enterprise
  - Integrate historically separate systems
  - Completion of mergers and acquisitions
  - Across physical and technology barriers

... and solving them implies implementing flexible, automated business processes across the enterprise.



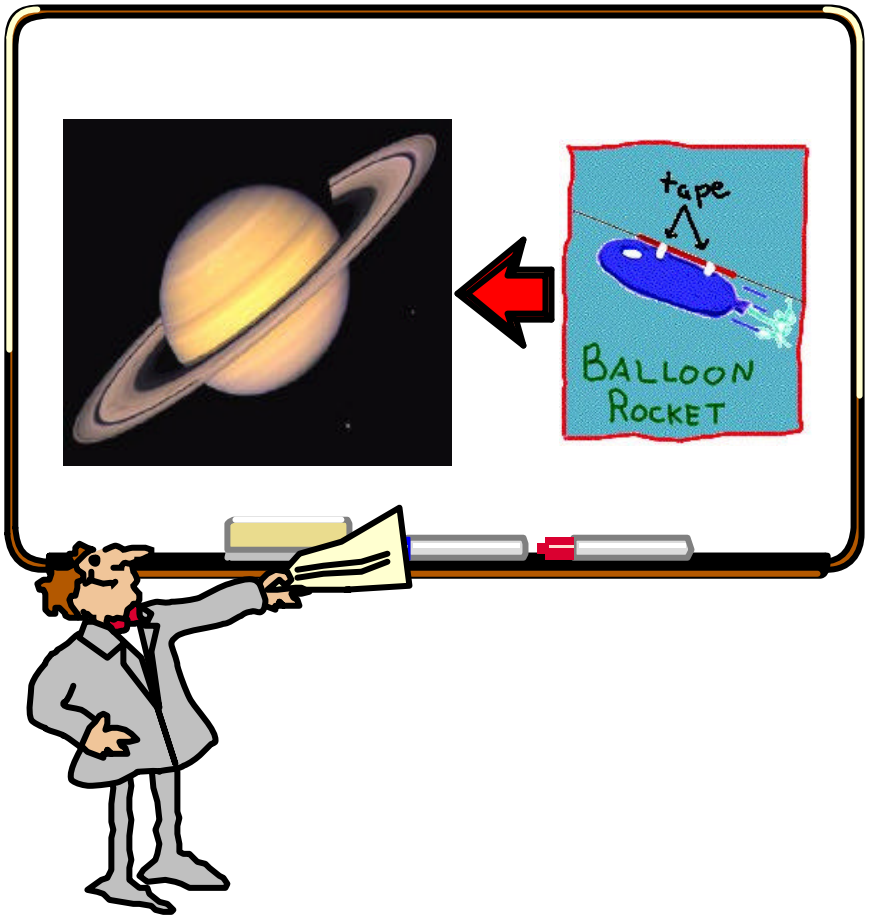
# There are many difficult problems in the way.

- **Technical Diversity**
  - A *multitude of technologies and platforms* support your business systems.
  - Processes may need to include the systems of *existing and future partners*, which are not only varied but beyond your control.
  - No single, fully functional integration solution will talk to them all.
- **Business Model Realisation**
  - Business process implementations are a combination of individual behaviour, business guidelines, application code and interactions between all of these.
  - It is *difficult to create and sustain* a consistent model of business data, function and processes across (or beyond) the enterprise.
  - Realising the business model in a flexible manner may involve refactoring the business, with all the attendant issues of transformation and change.
- **Undesirable Dependencies**
  - It is *difficult to isolate business behaviour* so that changing it to reflect business requirements tends to require changes to more than one system.
  - Traditional approaches to integration can suffer from the *same inflexibilities* as individual applications.

# SOA is the answer to everything!!!

Service Oriented Architecture describes:

- Viewing business as the delivery of services to consumers.
- Decoupling the packaging and delivery of those services from their implementation.
- Implementing those services as business processes that use a highly flexible combination of other business and technical services.
- Applying this view to the supporting IT systems:
  - by directly modelling and mapping services and processes to them.
  - by implementing interactions between systems as services.



# A more formal definition

(With thanks to Richard Whyte)

- A service-oriented architecture consists of an arbitrary number of services constructed to communicate through a consistent, integrated communication service.
- The communication service should insulate service interactions from the details of underlying communication protocols, and should support a variety of interaction types.
- A given SOA implementation may add but not remove constraints to the basic definition to facilitate additional benefit.
- Such an architecture specifies that all participating systems are fully encapsulated within properly formed services such that any exposed function forms part of a public interface described in a formal terminology.
- These services must support backward compatibility.

# An example service: roadside assistance

- Imagine the fictitious "Wildcat" luxury car manufacturer that offers a "roadside assistance" service to its customers.
- The package includes roadside and home assistance, vehicle recovery, the provision of replacement cars and covers the cost of temporary accommodation.
- However, Wildcat are not expert in any of these areas, and so they sub-contract delivery:
  - Roadside assistance - to a roadside assistance specialist
  - Vehicle recovery - to local garages with towing facilities
  - Replacement cars - to a car rental company
  - Temporary accommodation - to a motel chain.
- Wildcat therefore offer a comprehensive service to their customers without delivering any elements of that service themselves.
  - They can subcontract to the lowest cost or highest standard service provider in each area.
  - If a new provider becomes available offering lower cost or higher standards, Wildcat can switch to them.
  - If Wildcat need to offer a new package of slightly different services to some of their customers, it is easy for them to do so.

# Applying SOA to the roadside assistance service

- SOA Automates the provision of business services and the aggregation of service suppliers.
  - In the example, sourcing and aggregation of services could take place through face-to-face negotiation, telephone calls etc.
  - In SOA, services are increasingly located, selected and bound to through interactions between systems rather than people.
- SOA applies the same principles further down the technology stack to achieve flexibility in the way technology is applied to support the business.
  - The ability of a business to change is constrained by the agility of its IT systems
  - By treating IT capabilities as services and providing similar de-coupling and flexibility, SOA seeks to improve the overall agility of a business.
- SOA enables outsourcing or re-sourcing of business services at various levels of granularity.
  - In the example, relatively large-grained business services such as replacement cars are outsourced.
  - SOA enables more flexible outsourcing, such as:
    - Finer grained services, such as payments, credit checks etc.
    - More dynamic outsourcing large grained services, e.g. selecting between multiple hotel chains by geographical area or depending on current discount programmes etc.

# Implementing an SOA is a combination of many things

- The use of appropriate technologies to support process models, service interactions etc. with appropriate loose coupling.
  - XML, Web Services, EAI middleware etc.
- Modelling business to the level of detail required for automation.
  - Process Modelling, Component Business Modelling etc.
- Refactoring business capabilities and processes to achieve re-use and flexibility.
  - Business Process Reengineering, Business Transformation etc.
- Deploying a manageable supporting infrastructure.
  - Enterprise Service Bus, Service Management, Web Services etc.

# SOA is not ...

- EAI, OO, On-Demand or CBD
- Marketing rubbish
- Web services
- Dynamic discovery of services at runtime
- Particularly new
- Agreed by everyone
- ESB
- A product
- Composed of "AD services", "architecture services", "desktop support services"

# Reality Check – Some Service Oriented Architectures I Have Seen

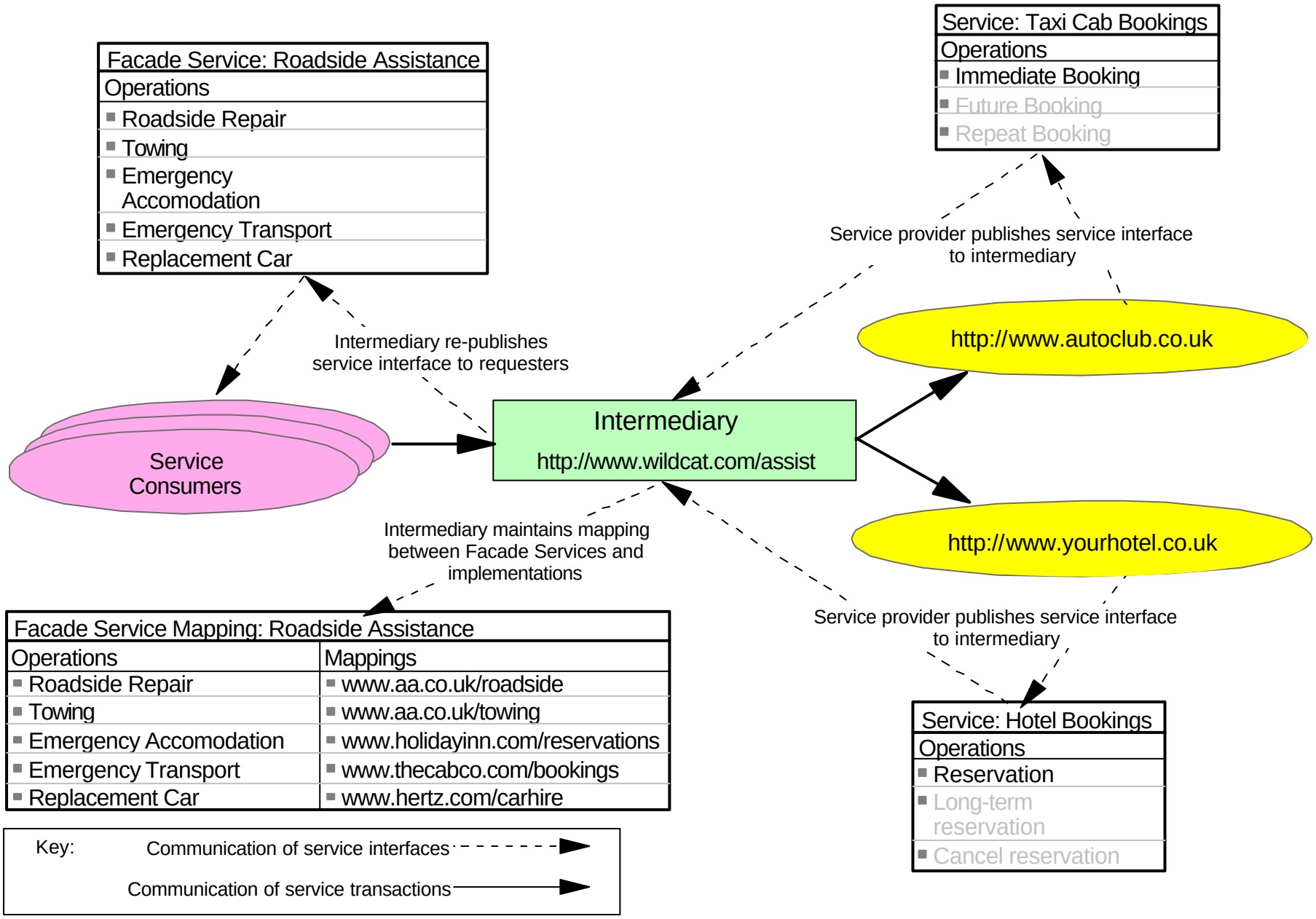
- CORBA
  - ✓Custom common business object model
  - ✓Stateless business service model
  - ✓IDL or XML interface definitions supporting IIOP and HTTP communication protocols
- EAI
  - ✓XML interface definition language and data model.
  - ✓EAI routing and communications infrastructure, e.g. WebSphere MQ or WBI Message Broker
  - ✓Framework APIs (e.g. Java, C etc.) to discover, bind to and invoke services
- Web Services
  - ✓Sparkassen Informatik, Charles Schwab – Evolving EAI-based SOAs to Web Services

IBM Software Group

# Achieving Flexibility – Façade Services

# Intermediaries and Service Facades are vital to the way SOA promotes agility.

- SOA promotes business agility by decoupling the packaging and delivery of a service from its implementation.
- This implies that *more than one interface is involved in delivering each service:*
  - The service consumer binds to an interface published by a delivery channel.
  - The delivery channel binds to an interface published by the service provider.
- SOA therefore requires intermediaries to provide these "Facade Services":
  - Service Gateways
  - Enterprise Service Bus
  - Portals
  - etc.


Facade Service: Roadside Assistance
Operations
Roadside Repair
Towing
Emergency Accomodation
Emergency Transport
Replacement Car
Service: Taxi Cab Bookings
Operations
Immediate Booking
Future Booking
Repeat Booking
Service provider publishes service interface to intermediary
Intermediary re-publishes service interface to requesters
http://www.autoclub.co.uk
Service Consumers
Intermediary
http://www.wildcat.com/assist
http://www.yourhotel.co.uk
Intermediary maintains mapping between Facade Services and implementations
Service provider publishes service interface to intermediary
Facade Service Mapping: Roadside Assistance
Operations
Mappings
Roadside Repair
www.aa.co.uk/roadside
Towing
www.aa.co.uk/towing
Emergency Accomodation
www.holidayinn.com/reservations
Emergency Transport
www.thecabco.com/bookings
Replacement Car
www.hertz.com/carhire
Service: Hotel Bookings
Operations
Reservation
Long-term reservation
Cancel reservation
Key:
Communication of service interfaces
Communication of service transactions

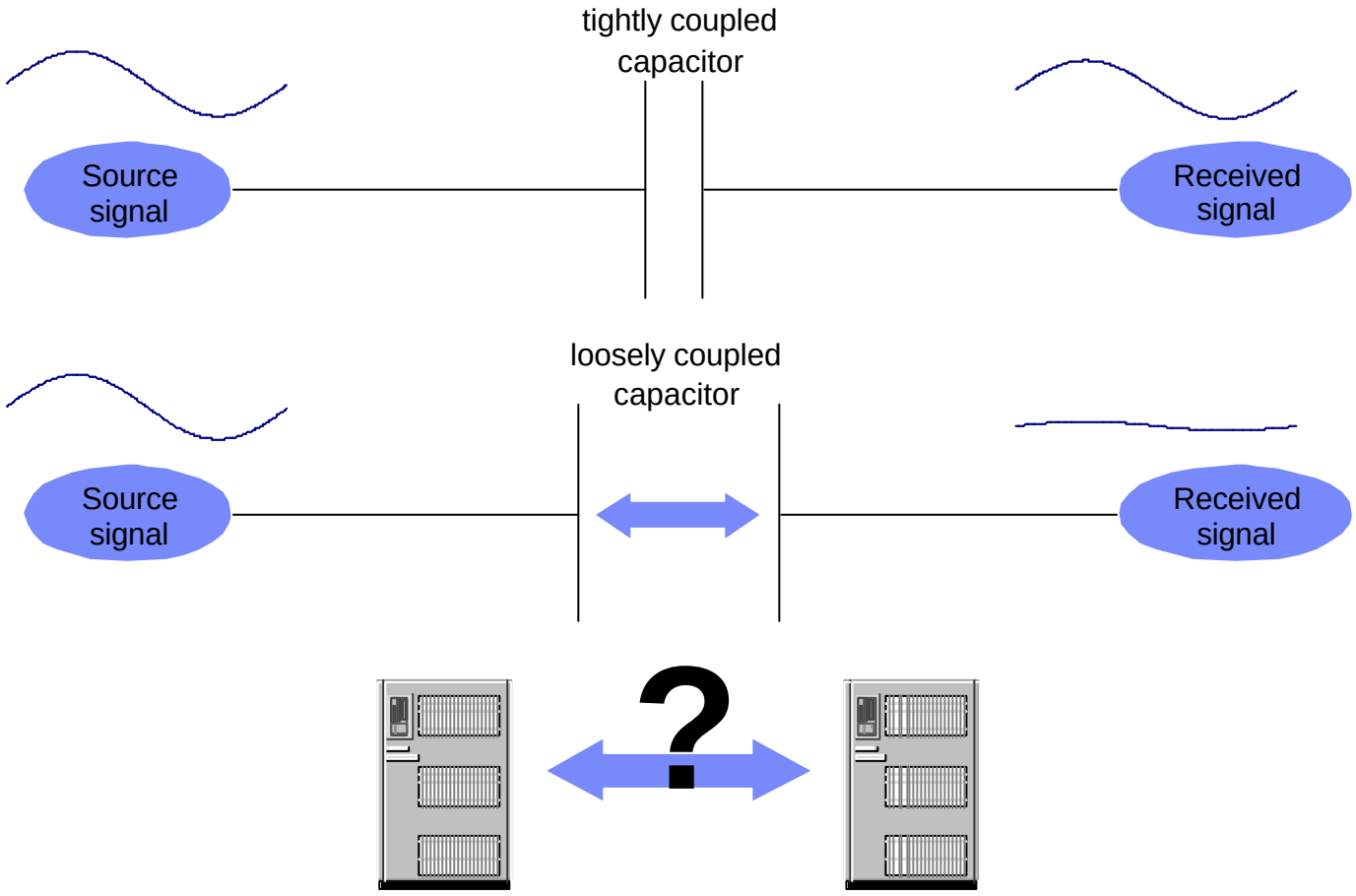
# Service Facades give us a new way of considering cohesion and encapsulation

- In existing techniques (e.g. OO and CBD) that use interfaces, the cohesion between operations in the interface is driven by ***implementation*** concerns.
  - Component and Object interfaces are composed of behaviours that it makes sense to implement in one place.
- For facade services, the cohesion between operations in the interface is driven by ***consumption*** concerns.
  - Facade services interfaces are composed of behaviours that it makes sense to access in one place.
- Examples:
  - In the "roadside assistance service" example, the services available were those that made sense to someone with a broken-down car. They do not necessarily make no sense implement together, which is why the car manufacturer has to source them from elsewhere.
  - In the world of banking, many supermarkets now sell financial services - insurance, current accounts, mortgages etc. Each supermarket could source these services from a variety of financial providers. Again, it makes more sense to *offer* these services together than to *implement* them together.

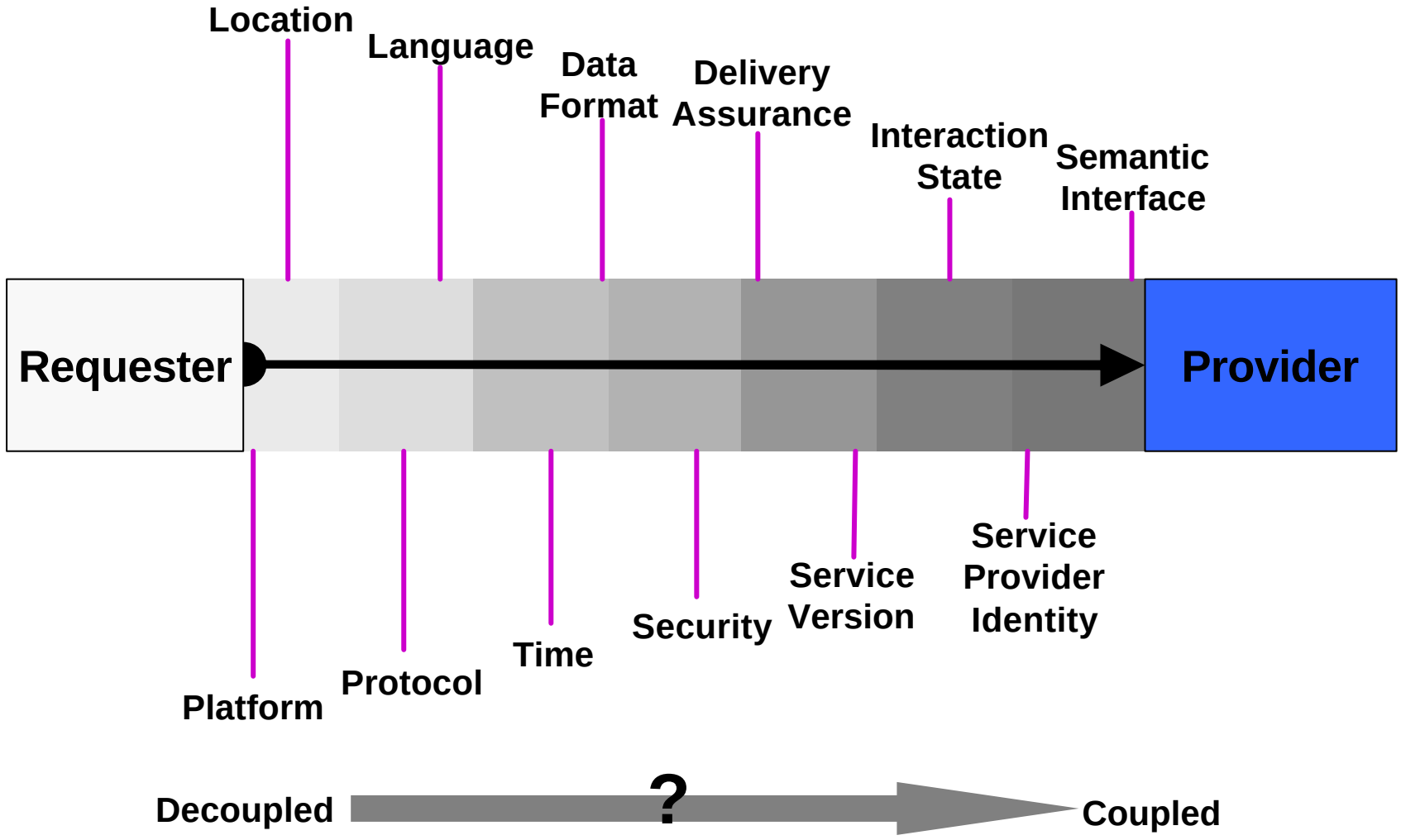

IBM
IBM Software Group
Achieving Loose Coupling
© 2004 IBM Corporation

# The “loose coupling” required by Service Oriented Architecture is not something I know how to implement for IT systems …

- *“Services are loosely coupled and invoked through communication protocols that stress location transparency and interoperability.”*



## … but I can consider whether I would like to couple various *aspects* of service interactions …



## … and I can use more sophisticated *coupling styles* than "coupled" or "decoupled".

- **Coupled**
  - Directly manipulated by service requester and provider application code.
  - e.g. business data model

- **Declared**
  - Clients and providers declare matching behaviour in interfaces.
  - e.g. WS-Security
  - **Transformed**
    - Specified in the service interface, but not manipulated by application code. Services declare different characteristics but the service infrastructure can mediate between requester and provider.
    - e.g. data format
  - **Negotiated**
  - Requester and provider interfaces declare a spectrum of behaviours, infrastructure negotiates an agreed behaviour for each interaction.
  - e.g. proposed WS-Policy, service provider identified dynamically through UDDI

- **Decoupled**
  - Entirely independent between client, provider and infrastructure.
  - e.g. platform independence through use of XML and HTTP

# Applying coupling styles to aspects of service interactions implies infrastructure capabilities as well as standards and protocols.

| Aspect | Coupling Style | Techniques |
|---|---|---|
| **Semantic Interface** | Coupled | WSDL, Application code, common objects, data dictionary |
| **Language** | Decoupled | Open standards, Web Services, J2EE, HTTP, XML |
| **Platform** | Decoupled | As above |
| **Data Format** | Declared or Transformed | WSDL, XML, adaptors, stylesheets, AD tools, middleware |
| **Protocol** | Declared or Transformed | HTTP, SOAP, Middleware, adaptors |
| **Location** | Decoupled | Communication protocols, HTTP, SOAP, middleware |
| **Service provider identity or implementation** | Declared, transformed or negotiated | Service routing, middleware, "broker"s, UDDI etc. |
| **etc. ...**<br>Time, Delivery Assurance, Error Handling, Security, Service Version, Interaction State | etc ... | etc ... |

IBM Software Group

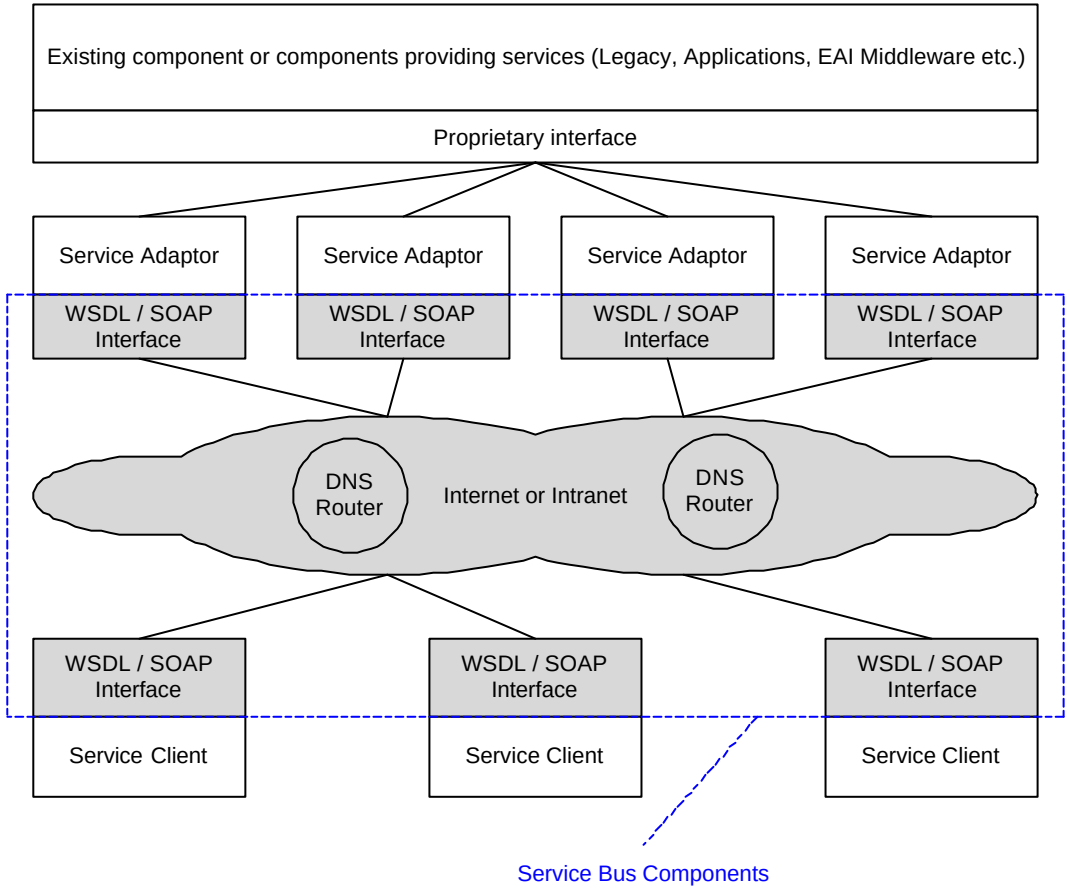
# A manageable infrastructure for SOA

# Infrastructure capabilities are also required to implement a manageable SOA in the enterprise.

**As a counter-example, consider the use of simple SOAP/HTTP wrappers**

- SOAP/HTTP and WSDL leverage URL addressing and the existing HTTP and DNS infrastructure for Service Oriented Architecture.
- But they do not address enterprise requirements for SOA:
  - Multiple qualities of service
  - Widespread integration and interoperability
  - Transformations and mediations
  - Substitution of service implementations

Critically, management of an SOA “infrastructure” provided in this way is distributed between service endpoints and the existing infrastructure.




IBM
IBM Software Group
Enterprise Service Bus
Patterns and Implementations
© 2004 IBM Corporation

# The marketplace uses the term “Enterprise Service Bus” to describe infrastructure technology providing these capabilities.

- Analysts describe a middleware capability …
  - “…a new architecture that exploits Web Services, messaging middleware, intelligent routing and transformation…” Roy Schulte, Gartner, Predicts 2003: Enterprise Service Buses Emerge
- … but some are more sceptical
  - ""ESBs have no defining advantage over products such as WebSphere MQ, and they lack its credentials.“ … Andrew Binstock (Integration Watch), December 2003 http://sdtimes.com/cols/integrationwatch.htm
- You have built your own … often using XML with EAI technology (e.g. message brokers or adaptors) or HTTP.
- A unifying view:
  - ESB is a logical infrastructure component that can be described by patterns
  - The ESB is defined by capabilities required to support SOA in the Enterprise in combination with other integration styles (messages and events)
  - The ESB provides a means to concentrate control and distribute topology

# ESB capabilities can be categorised and summarised based on the drivers described previously … but not all capabilities are required in all scenarios.

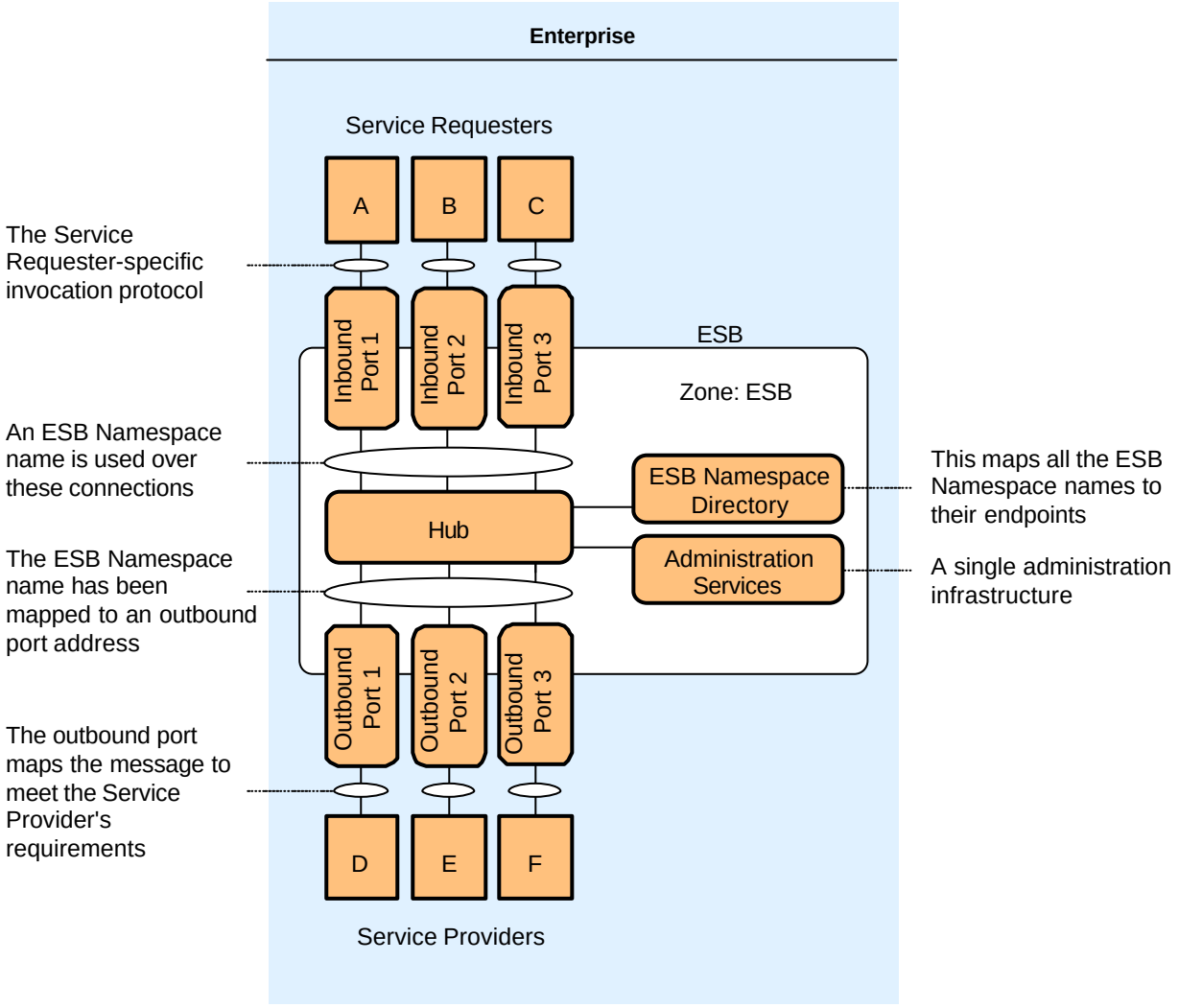
| | |
|---|---|
| **Communications, e.g.**<br>▪ Routing, addressing, protocols, pub/sub, async. | **Service Interaction, e.g.**<br>▪ Interface definition, service substitution, messaging model, SOAP, WSDL, directories |
| **Integration, e.g.**<br>▪ Database, legacy, middleware connectivity, service aggregation, app server connectivity, protocol transformation. | **Quality of Service, e.g.**<br>▪ Transactions, delivery assurance |
| **Security, e.g.**<br>▪ Authentication, authorisation, non-repudiation, confidentiality, standards support (WS-Security, Kerberos etc.) | **Service Level, e.g.**<br>▪ Performance, throughput, availability, scalability. |
| **Message Processing, e.g.**<br>▪ Encoded logic, content-based logic, message and data transformations, intermediaries etc.. | **Management and Autonomic, e.g.**<br>▪ Service provisioning and registration, logging, metering, monitoring, systems management etc.. |
| **Modelling, e.g.**<br>▪ Object modelling, common formats and libraries, public vs. private etc.. | **Infrastructure Intelligence, e.g.**<br>▪ Business rules, policy driven behaviour, pattern recognition, etc.. |

## *Minimum* ESB capabilities can be defined from the requirement to support SOA in the enterprise, and define which technologies are consistent with the ESB.

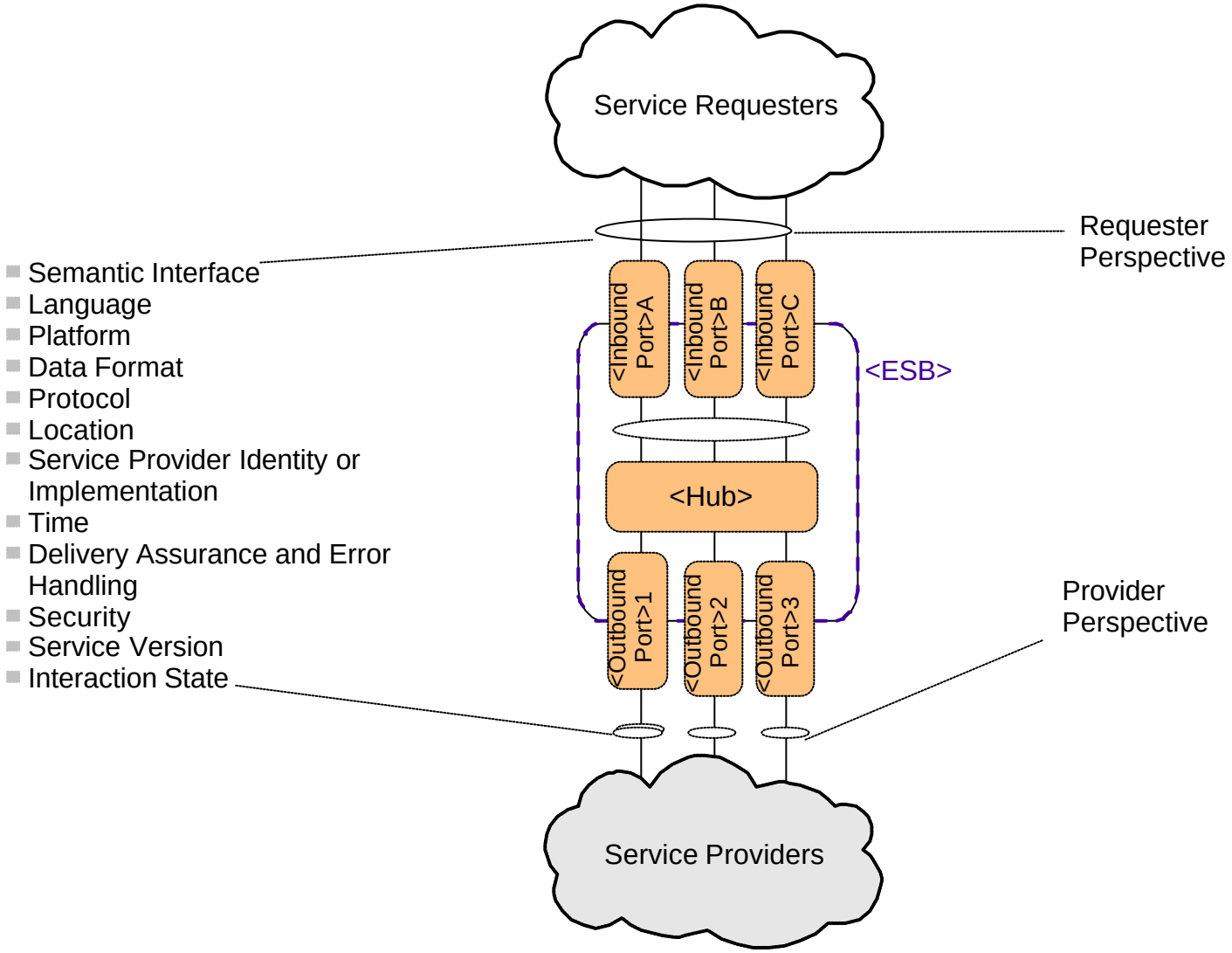
- If we require ESB to support our definition of SOA and add :
  - *The ESB provides the means to manage the service infrastructure and the capability to operate in today's distributed, heterogeneous environment.*
- We can define a minimum capability ESB implementation:

| Communication | Integration |
|---|---|
| ▪Routing and addressing services providing location transparency.<br>▪An administration capability<br>▪At least one messaging paradigm (e.g. request / response, pub/sub, etc.).·<br>▪At least one transport protocol that is or can be made widely available. | ▪Support for multiple means of integration, e.g. Java 2 Connectors, Web Services, etc. |
| **Service Interaction** | |
| ▪Service messaging and interfacing model. | |

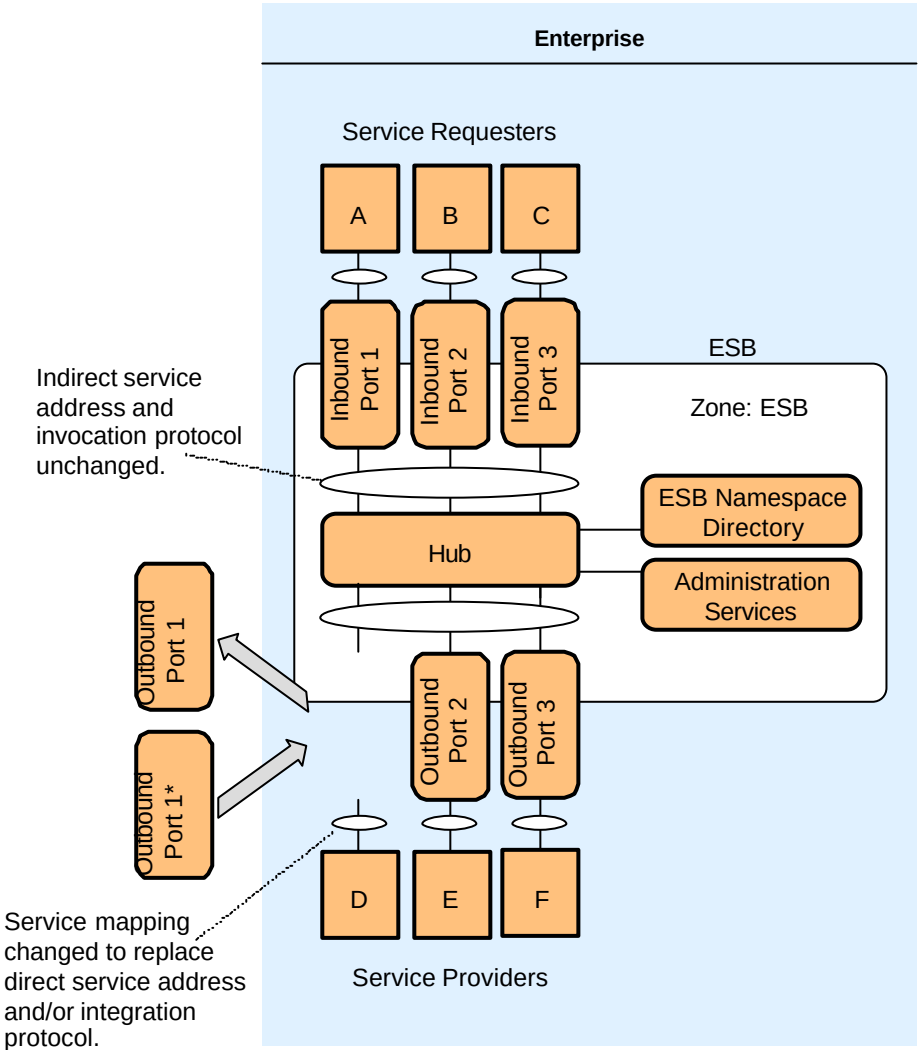
# In the IBM Patterns for e-business we model the ESB as a “Zone” supported by several “hubs”



# We can model the role played by the ESB in mediating coupling in service interactions.

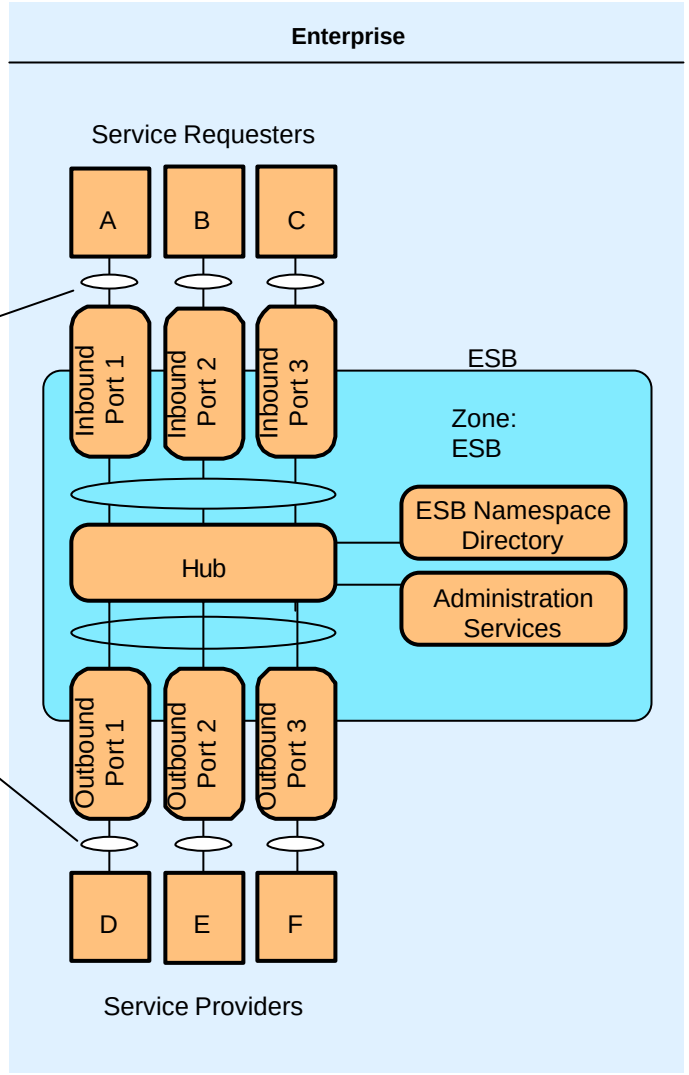


# The ESB supports substitution of service implementation

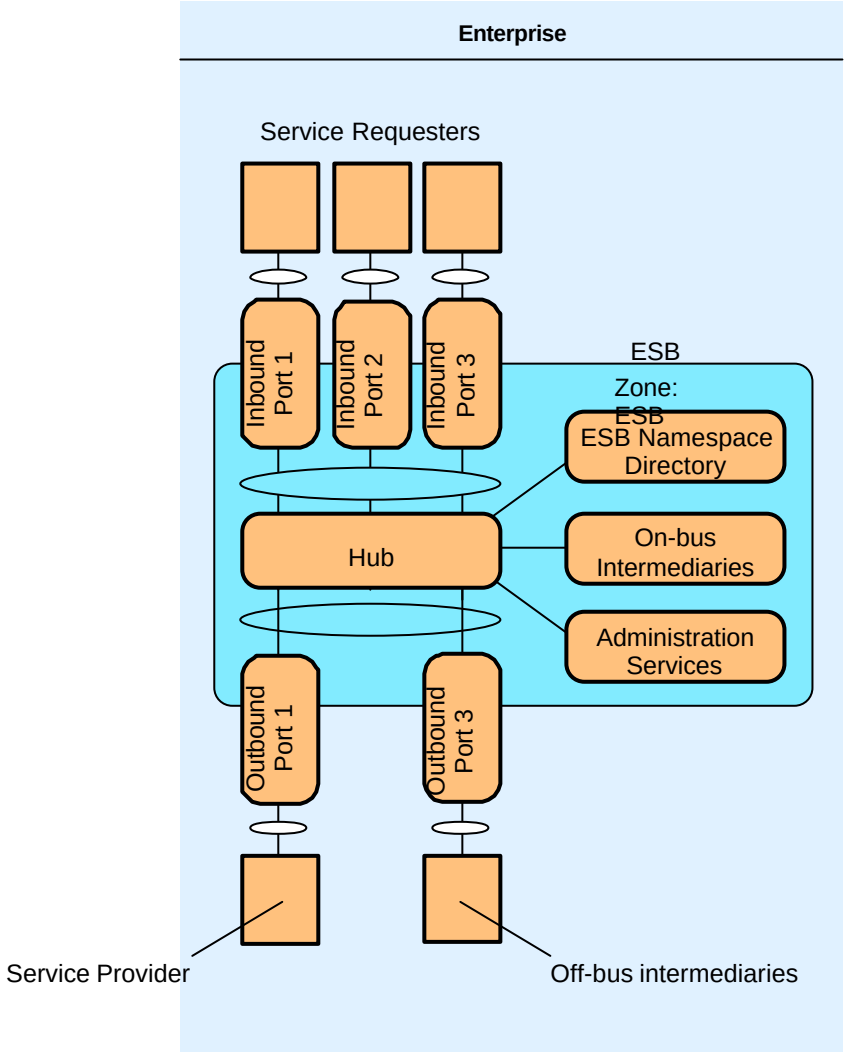


# The ESB supports a variety of connection styles

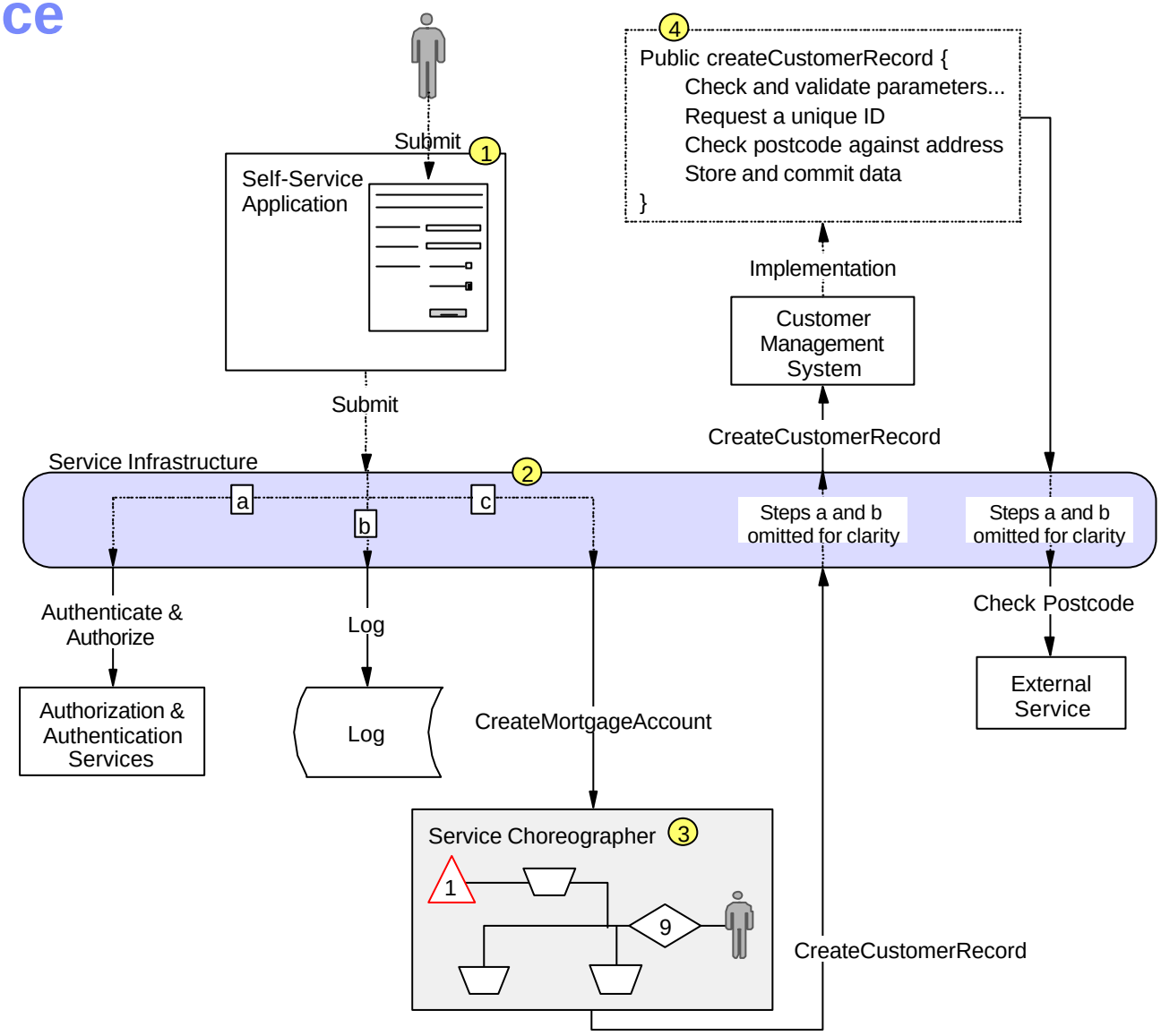
- API
- Protocol
- Adaptor



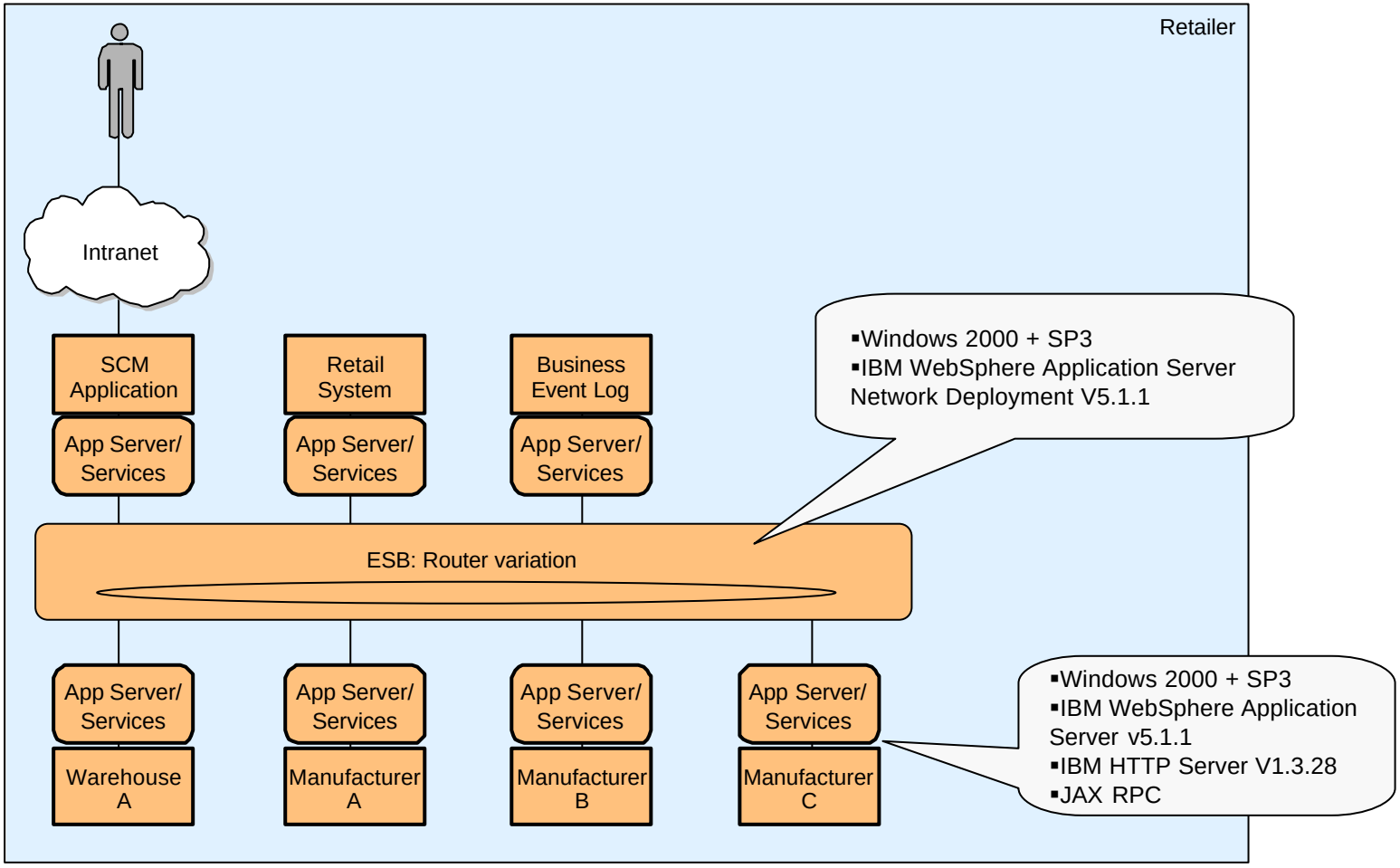
# The ESB allows on- and off-bus intermediaries to be sequenced around service invocations



# The ESB acts as a bridge between different types of service

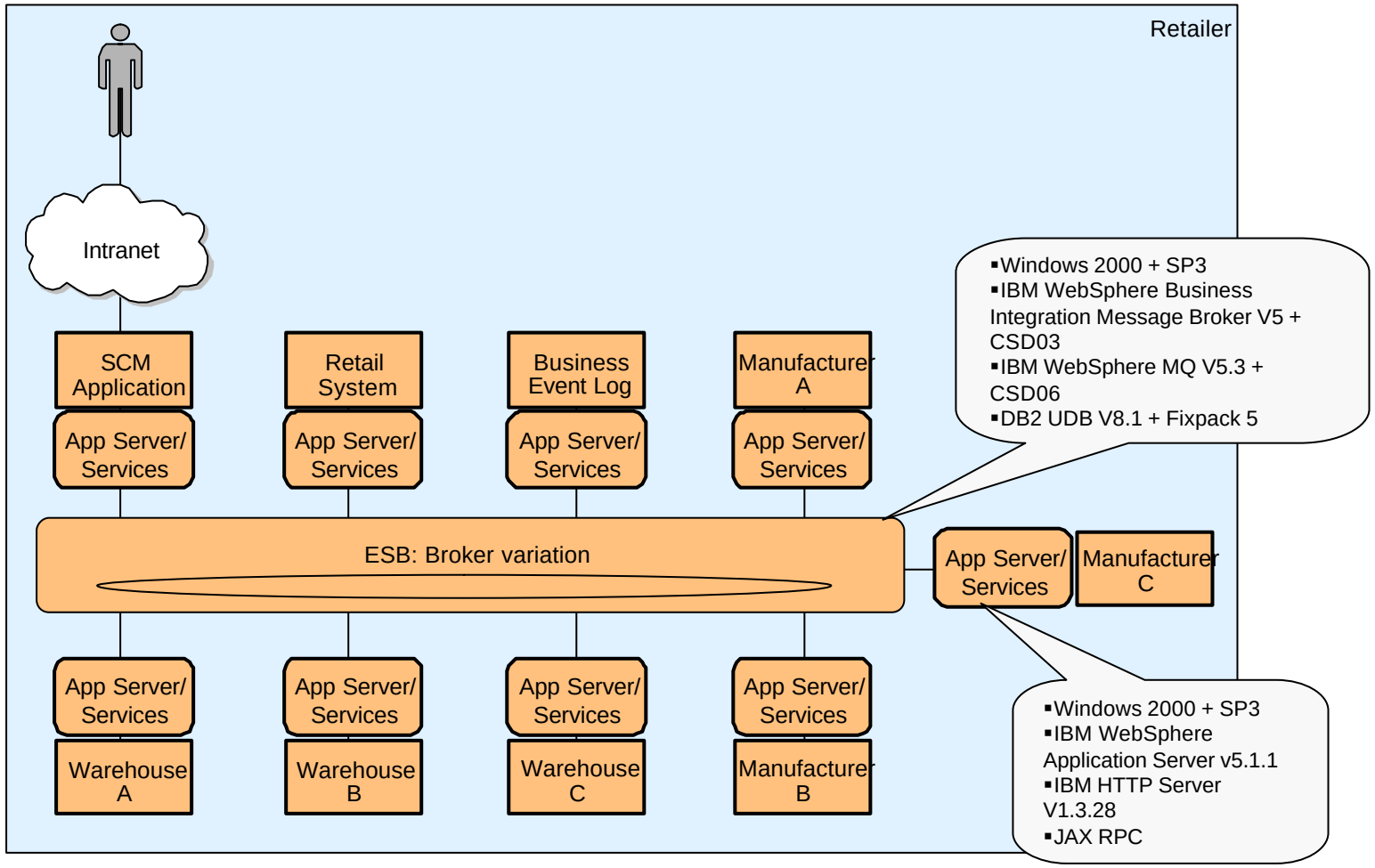


# ESB Today: a simple implementation with Web Services Gateway



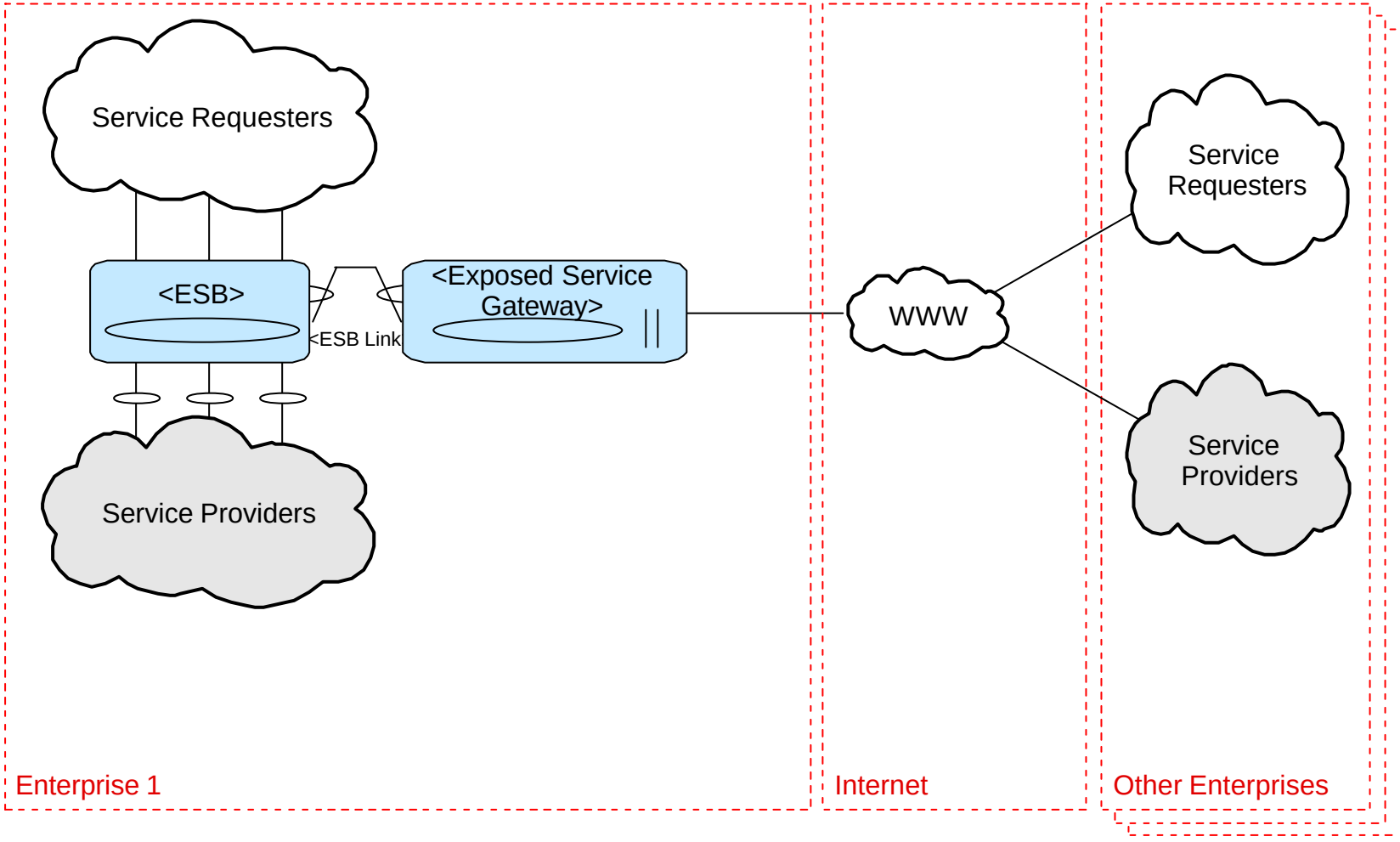

See redbook - Patterns: Implementing an SOA using an Enterprise Service Bus, SG24-6346

# ESB Today: a more sophisticated implementation with WBI Message Broker

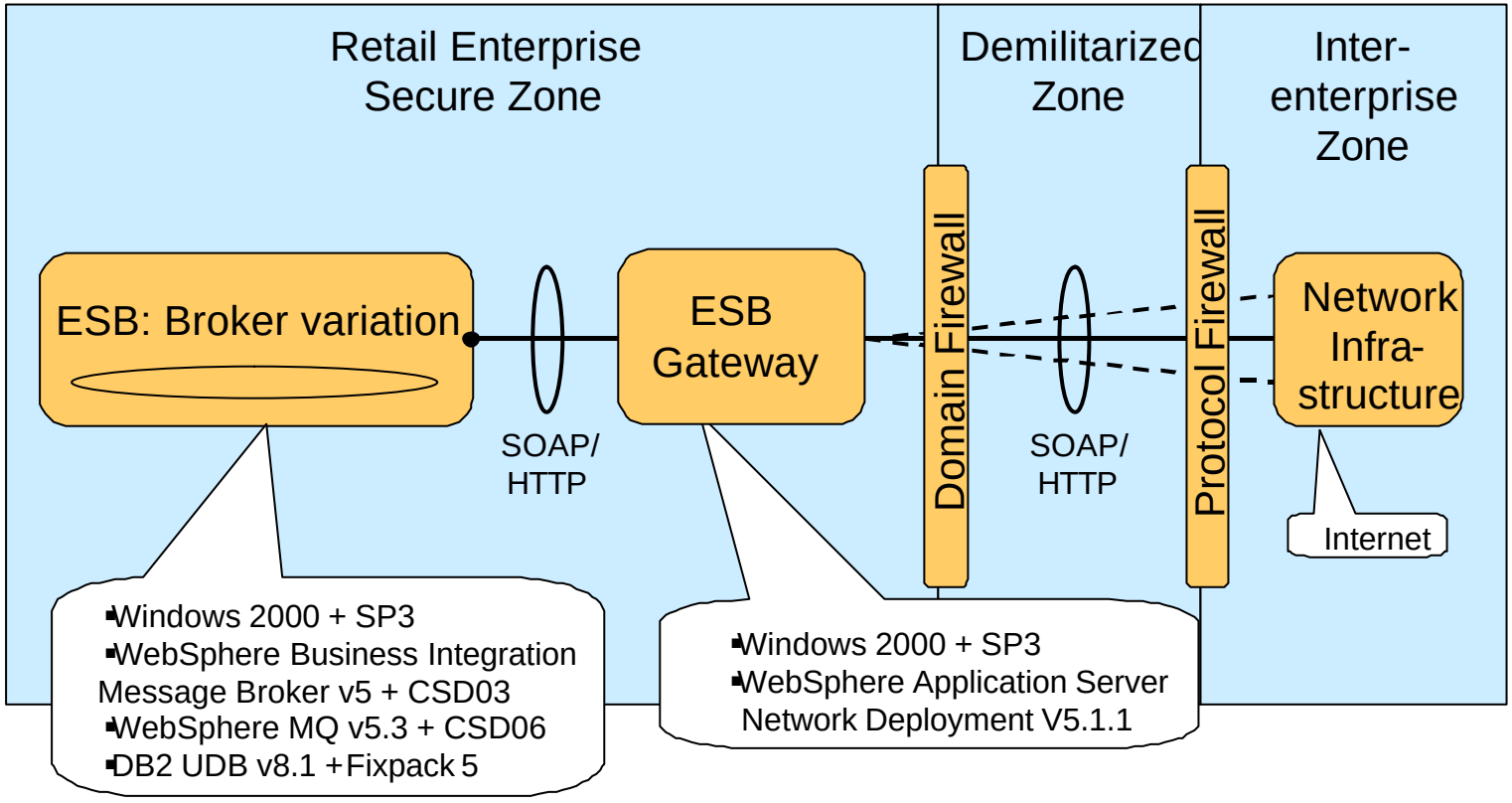



See redbook - Patterns: Implementing an SOA using an Enterprise Service Bus, SG24-6346

## Some scenarios will require more than one ESB pattern. Linking patterns in this way identifies requirements to match service interaction characteristics between infrastructure components.

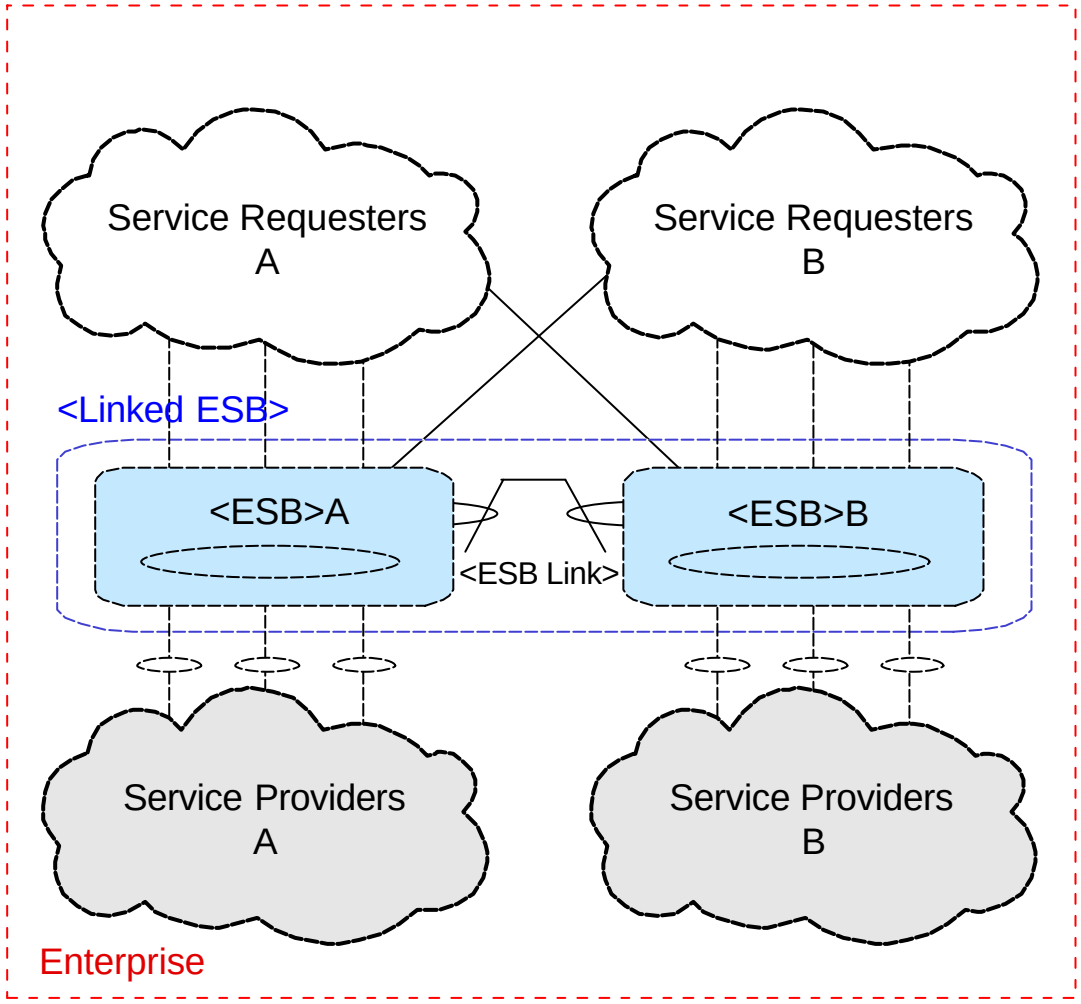


# Implementing an ESB and a Gateway with WBI Message Broker and Web Services Gateway




See redbook - Patterns: Implementing an SOA using an Enterprise Service Bus, SG24-6346

# In more complex situations ESBs will need to be linked



# Summary and Resources

# SOA and ESB are still evolving

- SOA is an architectural style that promotes architectural and business flexibility. Organisations have derived real business value from its principles.
- It is an evolution that takes succesful, proven techniques from previous architectural styles (OO, CBD) and technologies (CORBA, EAI etc.).
- SOA is not Web Services, but Web Services will become an increasingly important technology.
- The Enterprise Service Bus is an evolving conceptual architecture to support the loose coupling, business flexibility and manageability required by SOA in the Enterprise.
- ESBs can be constructed using current technology, but patterns and products will evolve to implement more sophisticated ESB capability in future.

# Plugs

- **Redbook: Patterns: SOA and Web Services, SG246303** http://publib-b.boulder.ibm.com/Redbooks.nsf/RedbookAbstracts/sg246303.html?Open
  - Mark Endrei, Jenny Ang, Ali Arsanjani, Sook Chua, Philippe Comte, Pål Krogdahl, Dr Min Luo, Tony Newling.
  - Introducing SOA concepts, standards and technologies
- **Redbook: Patterns: Implementing a Service Oriented Architecture using Enterprise Service Bus patterns, SG246346**
  http://publib-b.boulder.ibm.com/Redbooks.nsf/RedbookAbstracts/sg246346.html?Open
  - Martin Keen, Amit Acharya, Susan Bishop, Alan Hopkins, Sven Milinski, Chris Nott, Rick Robinson.
  - Further SOA concepts and implementation patterns for the Enterprise Service Bus
- **Enterprise Service Bus whitepapers**
  http://www-106.ibm.com/developerworks/webservices/library/ws-esbscen/

  http://www-106.ibm.com/developerworks/webservices/library/ws-esbscen2.html
  - http://www-106.ibm.com/developerworks/webservices/library/ws-esbscen3/
- **Perspectives on Web Services**
  http://publib-b.boulder.ibm.com/Redbooks.nsf/RedbookAbstracts/sg246346.html?Open
  - Olaf Zimmermann, Mark Tomlinson, Stefan Peuser
  - Architectural and development guidance on implementing SOA using Web Services and IBM software

# Resources

- IBM Web Services hompage
  - http://www-3.ibm.com/software/solutions/webservices/
- Web Services Interoperability organisation
  - http://www.ws-i.org/
- OASIS open standards organisation (ebXML etc.)
  - http://www.oasis-open.org/
- Specifications:
  - Business Process http://www-106.ibm.com/developerworks/library/ws-bpel/
  - Transactions http://www-106.ibm.com/developerworks/library/ws-transpec/
  - Security http://www-106.ibm.com/developerworks/webservices/library/ws-secmap/
  - SOAP http://www.w3.org/TR/SOAP/
  - WSDL http://www.w3.org/TR/wsdl
  - UDDI http://www.uddi.org/specification.html (public directory at https://uddi.ibm.com/ubr/registry.html)
  - WS-ReliableMessaging at http://www-106.ibm.com/developerworks/library/ws-rm/
- Tutorials, articles, technologies
  - http://www.ibm.com/webservices/
  - http://msdn.microsoft.com/webservices/
- Detailed presentations and LOTS more links
  - http://www.ibm.com/developerworks/speakers/colan
- Preview Technology
  - http://www.alphaworks.ibm.com/webservices
- Case Studies
  - http://www.ibm.com/software/ebusiness/jstart/casestudies/webservices.html

# Resources continued ...

- "Web Service Oriented Architecture - The Best Solution to Business Integration", Annrai O'Toole, Cape Clear Software CEO at
  - http://www.capeclear.com/clear_thinking1.shtml
- "SOA - Save Our Assets", Lawrence Wilkes, CBDI Forum (subscription required) at
  - http://www.cbdiforum.com/report_summary.php3?topic_id=2&report=623&start_rec=0
- The IBM series of articles "Migrating to a Service Oriented Architecture" by Kishore Channabasavaiah, Kerrie Holley and Edward M. Tuggle Jr. at
  - http://www-106.ibm.com/developerworks/library/ws-migratesoa/
  - http://www-106.ibm.com/developerworks/webservices/library/ws-migratesoa2/
- LooselyCoupled.com has a list of Enterprise Service Bus links at
  - http://www.looselycoupled.com/blog/2003_07_13_lc.htm - 105836683995371084
- The original Gartner article defining the Enterprise Service Bus requires a subscription, but can be found by searching their site http://www.gartner.com , and is entitled "Predicts 2003: Enterprise Service Buses Emerge", published 9th December 2002 by Roy W. Schulte.
- IBM Patterns for e-Business website at
  - http://www.ibm.com/developerworks/patterns/
- "The On Demand Operating Environment" at
  - http://www-3.ibm.com/e-business/doc/content/evolvetech/operating_environment.html
- "Using Web Services for Business Integration", Geert Van de Putte et al, IBM Redbook SG246583 at
  - http://publib-b.boulder.ibm.com/Redbooks.nsf/RedpieceAbstracts/sg246583.html?Open
- "WebSphere Version 5.1 Application Developer 5.1.1 Web Services Handbook", Ueli Wahli et al, IBM Redbook SG246891-01 at
  - http://publib-b.boulder.ibm.com/Redbooks.nsf/9445fa5b416f6e32852569ae006bb65f/d336dbf7a0ae01c385256d5000578477?OpenDocument
- "Coarse-Grained Interfaces Enable Service Composition in SOA", Jeff Hanson, Builder.com at
  - http://builder.com/5100-6386-5064520.html