Using DDS to Enable The Real-Time Enterprise Service Bus (RT-ESB)

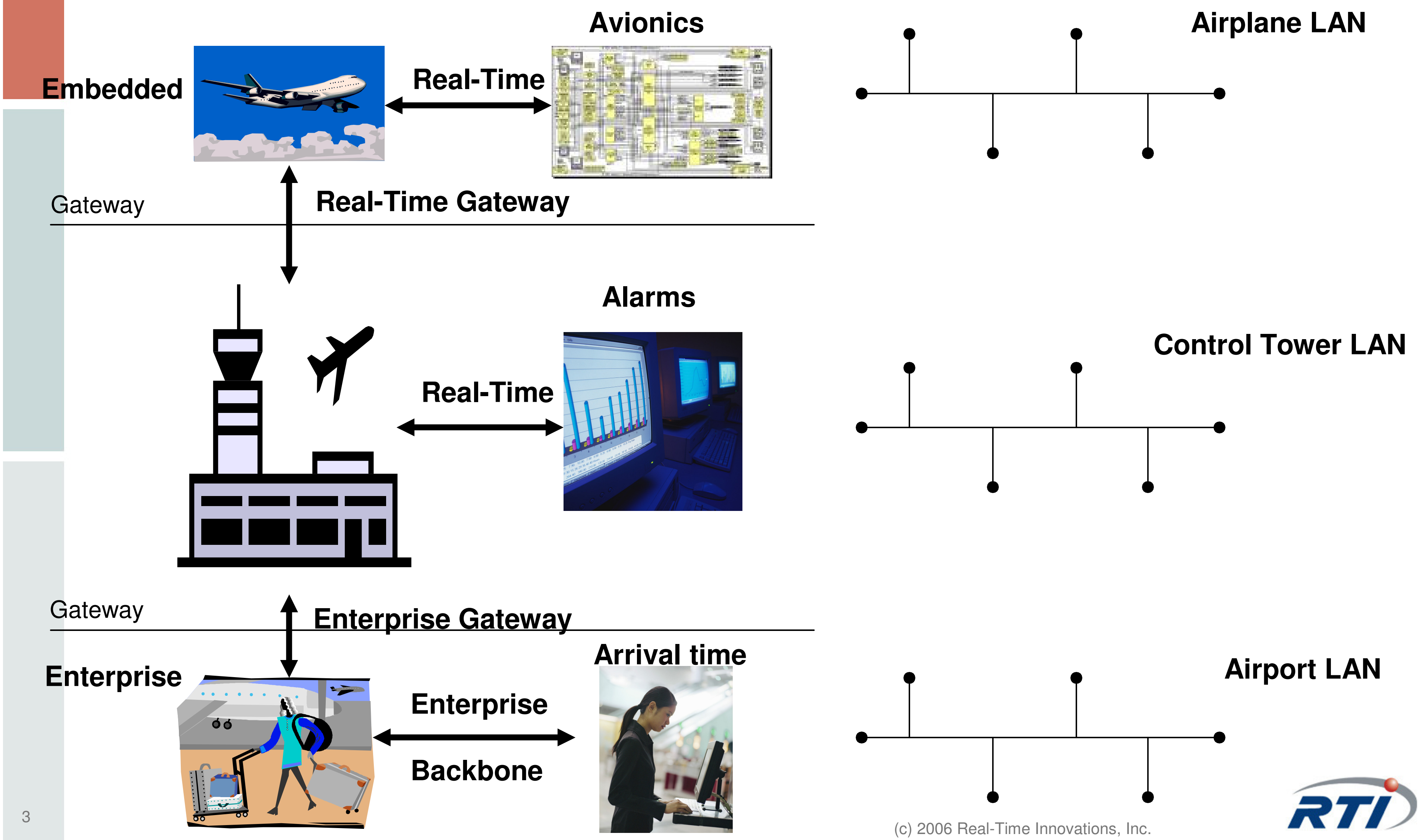
Rajive Joshi, Ph. D.
Gerardo Pardo-Castellote, Ph.D.
Real-Time Innovations, Inc

OMG Real-time and Embedded Systems Workshop
Arlington, VA, July 13, 2006

Agenda

- Life at the Edge
- The Gap between the Enterprise and the Edge
- Bridging the Gap: The Real-Time ESB
  - Critical technologies
  - Integrating DDS and JMS
  - Integrating DDS and SQL
  - Integrating DDS and Web Services

Life at the Edge: Air traffic control example



Embedded

Avionics

Real-Time

Airplane LAN

Gateway

Real-Time Gateway

Alarms

Real-Time

Control Tower LAN

Gateway

Enterprise Gateway

Enterprise

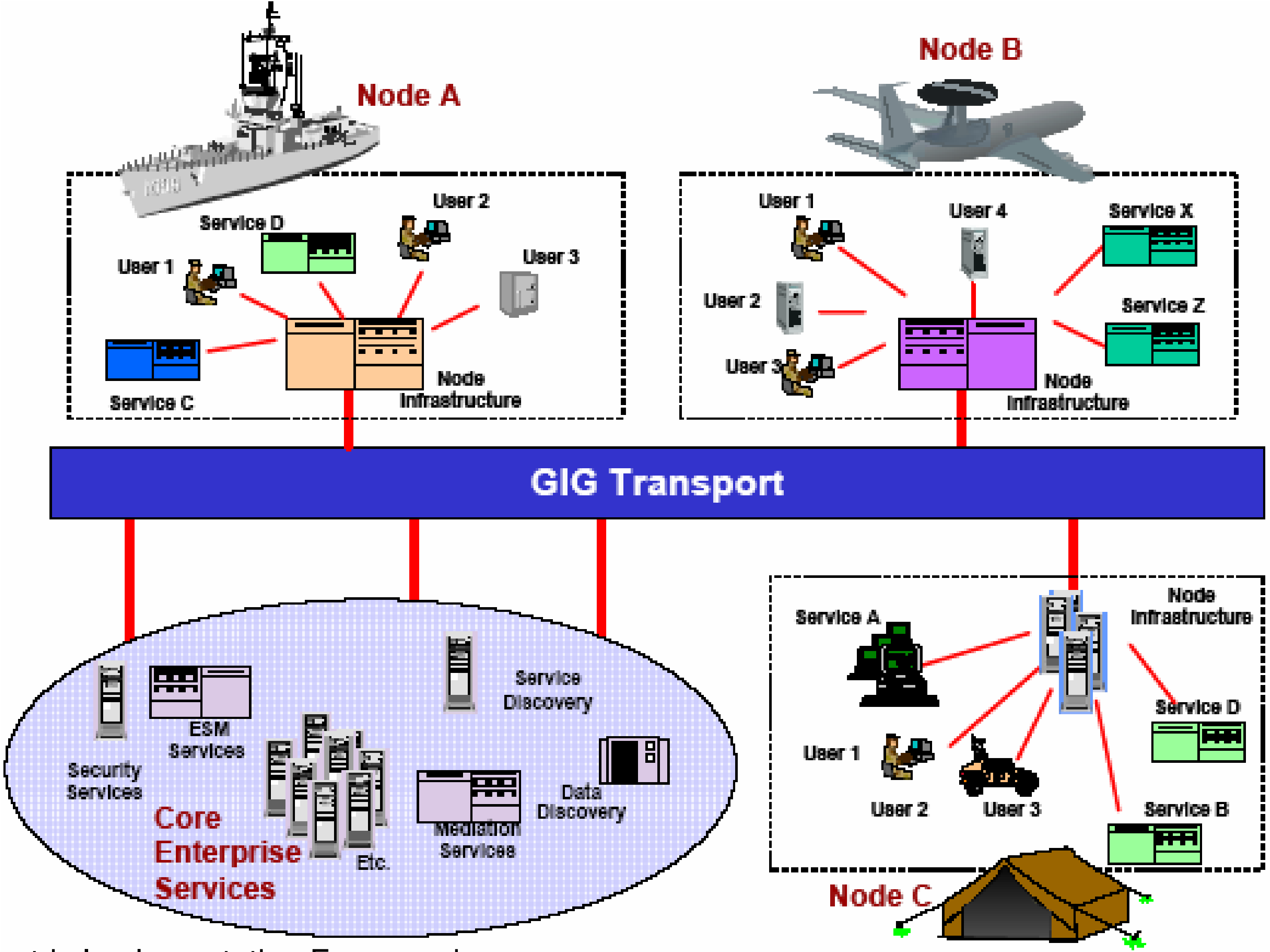
Arrival time

Enterprise

Backbone

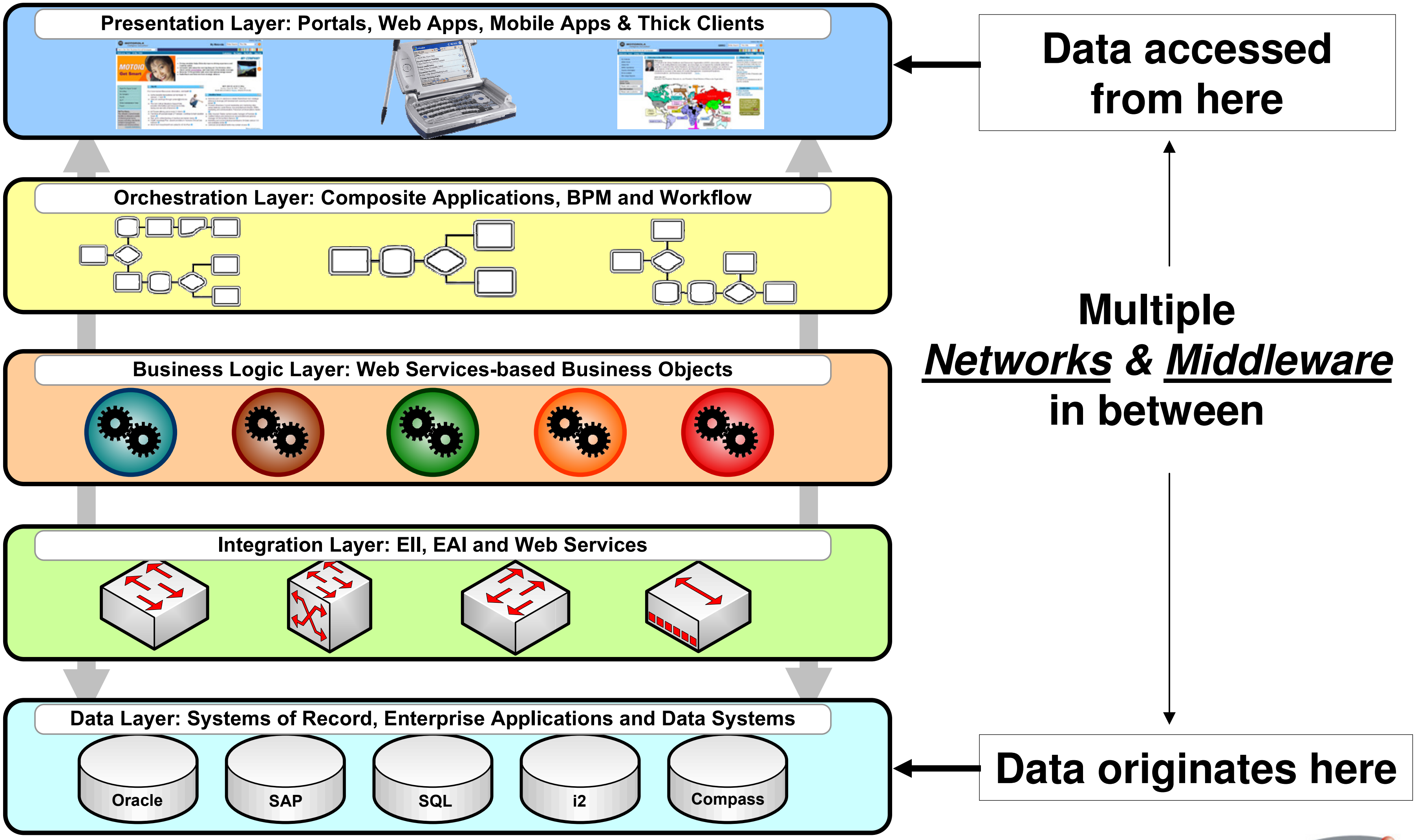
Airport LAN

Life at the Edge: Net-Centric "Node"



Net-Centric Implementation Framework
Part 4: Node Design Guidance Figure 1. c

The Gap: Enterprise surrounds the Edge



Presentation Layer: Portals, Web Apps, Mobile Apps & Thick Clients

Orchestration Layer: Composite Applications, BPM and Workflow

Business Logic Layer: Web Services-based Business Objects

Integration Layer: EII, EAI and Web Services

Data Layer: Systems of Record, Enterprise Applications and Data Systems

Data accessed from here

Multiple *Networks* & *Middleware* in between

Data originates here

Diagram Source: Redshaw, Motorola

The Gap: Differing Requirements

Edge

Enterprise

High performance a must
- **low latency**
- **high throughput**

Same middleware stack on both sides acceptable

LAN environment / Trust boundary

Scalability is important

Effective Interoperability a must

Different stacks on either side

Lower performance acceptable

May cross LAN/trust boundaries

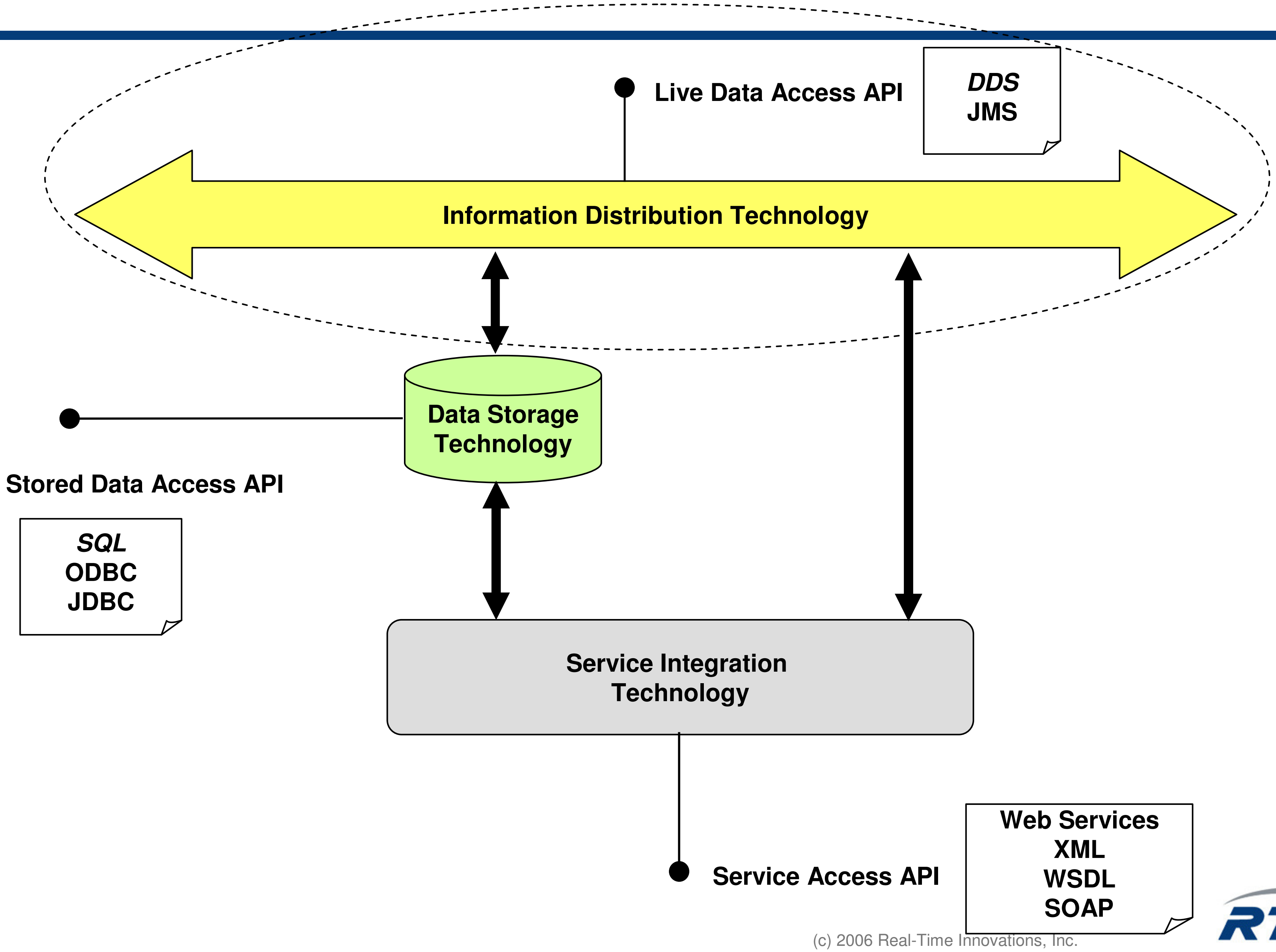
Scalability is important

Impedance mismatch!

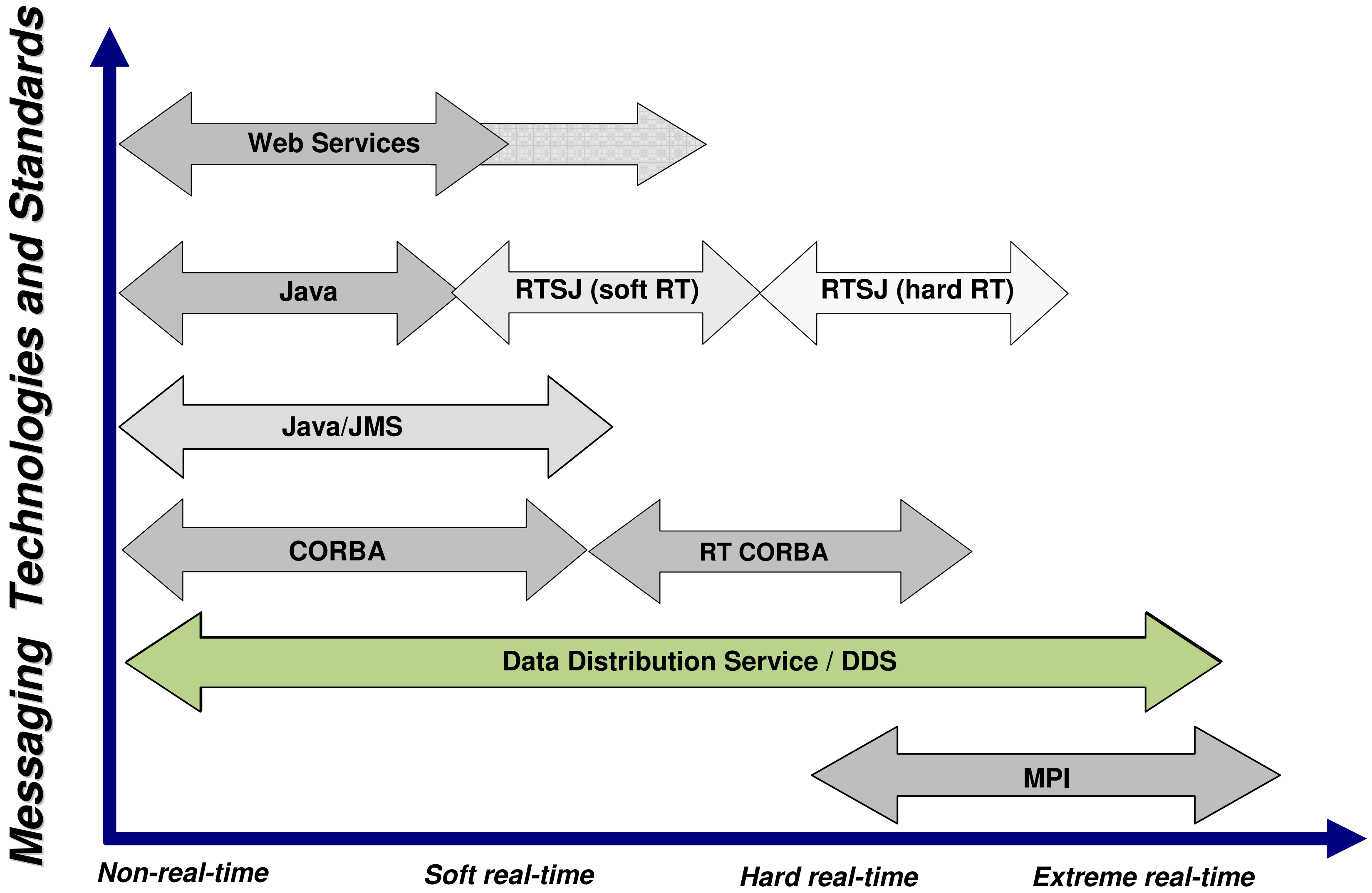
Agenda

- Life at the Edge
- The Gap between the Enterprise and the Edge
- Bridging the Gap: The Real-Time ESB
  - Critical technologies
  - Integrating DDS and JMS
  - Integrating DDS and SQL
  - Integrating DDS and Web Services

Critical Technologies



Information Distribution

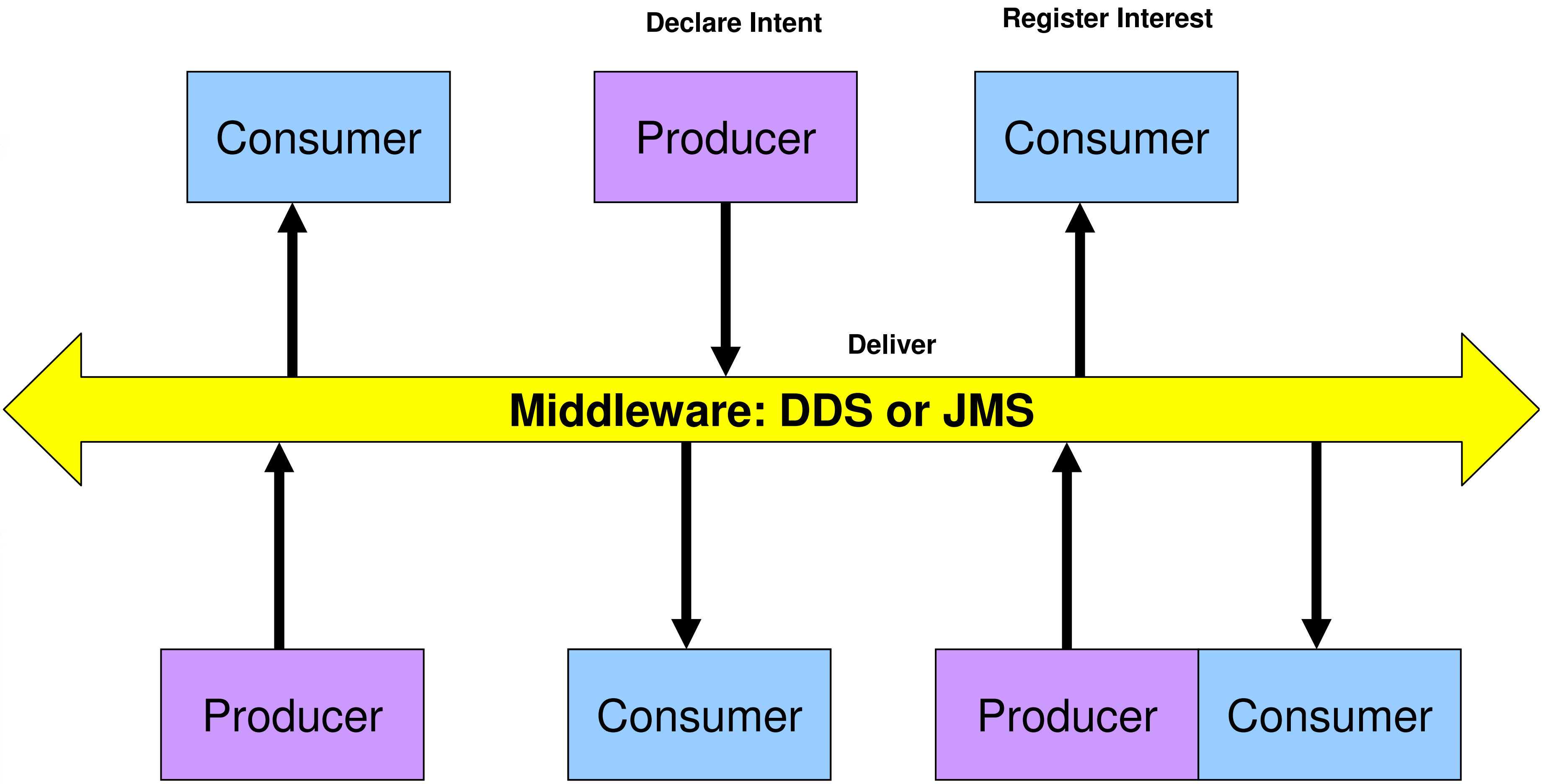


Adapted from NSWC-DD OA Documentation

Information Distribution Middleware Choices

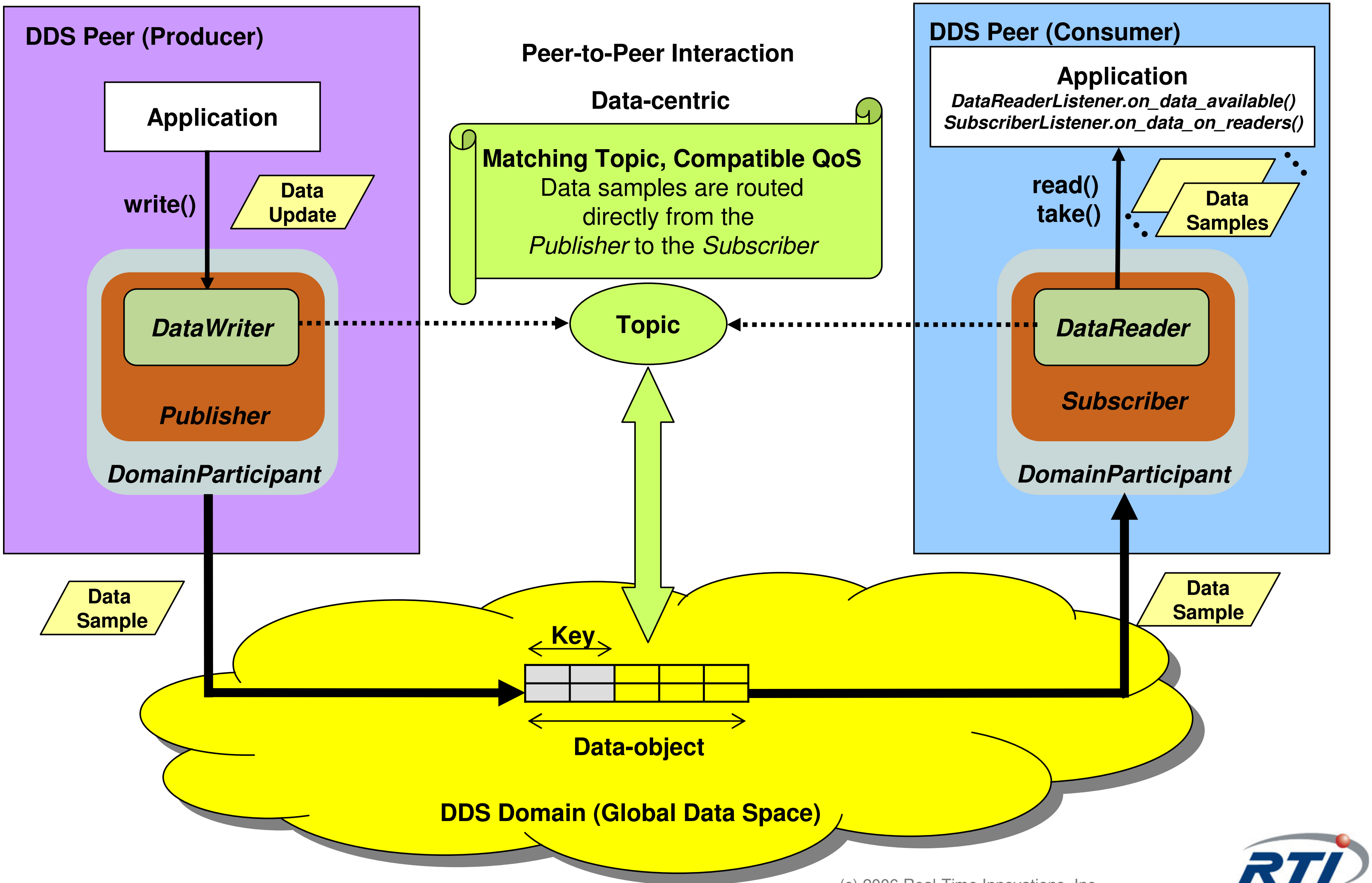
- Really only three choices
  - Use proprietary middleware
    - Eg. MQ Series, Tibco, In house, …
  - JMS
    - Standards based
    - Popular in Enterprise
      - Java only
      - API only, no wire interoperability spec
  - DDS
    - Standards based
    - Popular in “Edge” applications
      - Multiple platforms and languages
      - Wire interoperability
      - Real-Time QoS
      - High performance

Publish-Subscribe Middleware



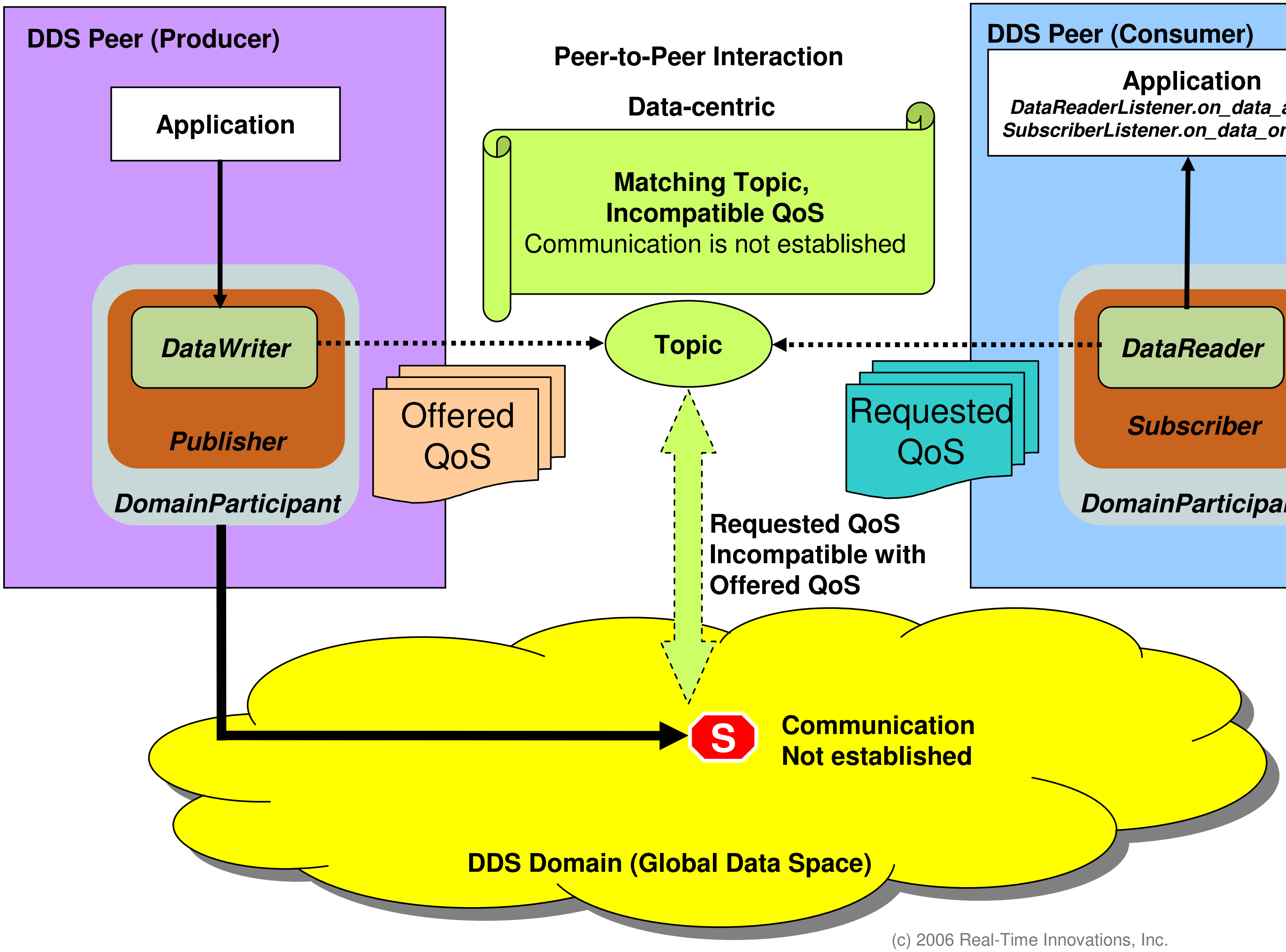
Decouples Producers and Consumers

Data Distribution: DDS Synopsis

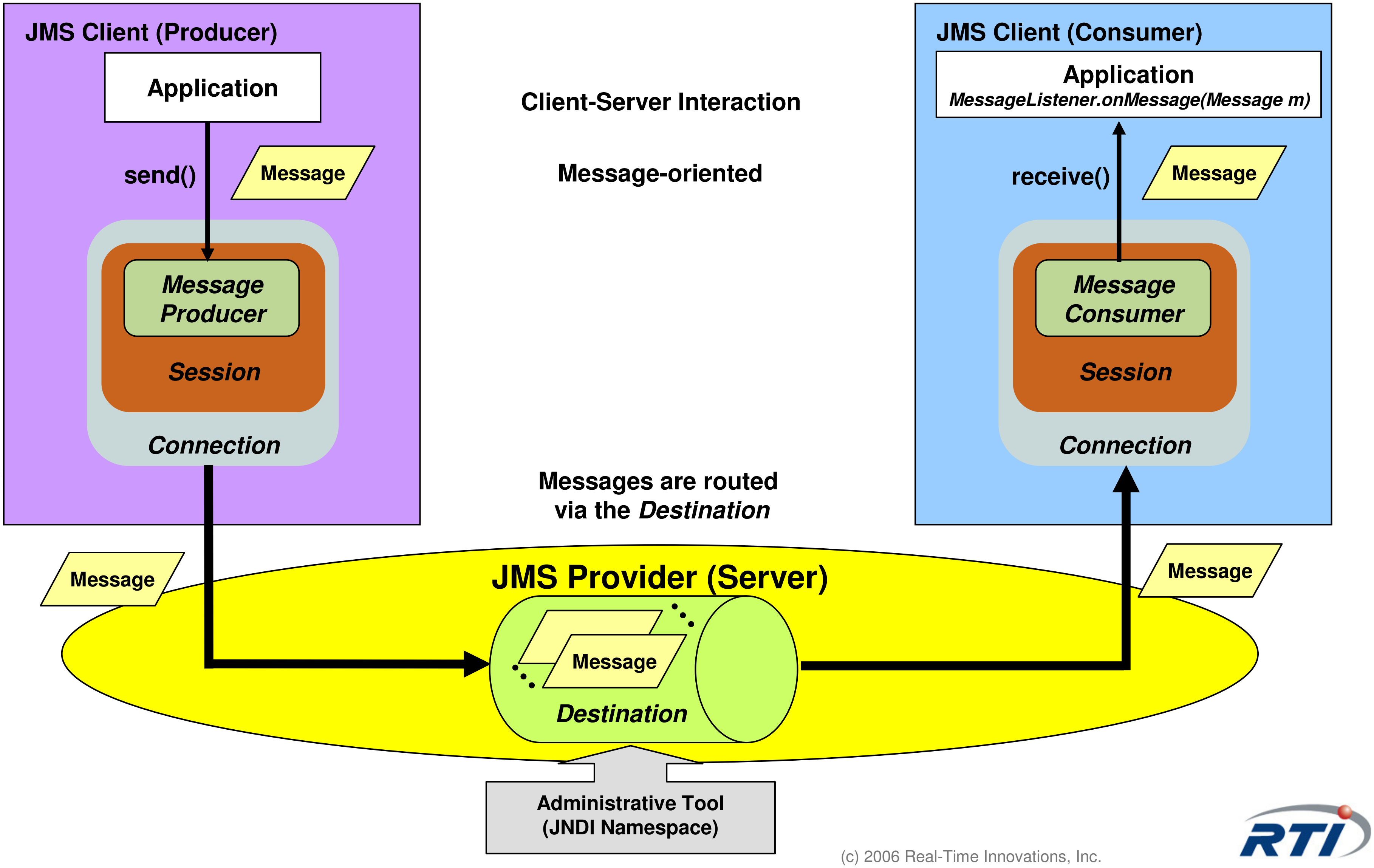


Matching Topic, Compatible QoS
Data samples are routed directly from the *Publisher* to the *Subscriber*

DDS Synopsis: Requested/Offered Semantics



JMS Synopsis



JMS Client (Producer)

Application

send()

Message

Message Producer

Session

Connection

Client-Server Interaction

Message-oriented

Messages are routed via the *Destination*

JMS Client (Consumer)

Application

MessageListener.onMessage(Message m)

receive()

Message

Message Consumer

Session

Connection

Message

JMS Provider (Server)

Message

Destination

Message

Administrative Tool
(JNDI Namespace)

JMS-DDS Semantic Mapping: Equivalents

JMS	DDS
Client	Application
Provider = client runtime + server (if any)	Middleware, Service
Domains are PtP and Pub/Sub	Domain represents a global data space, comprised of a set of communicating user applications
ConnectionFactory	DomainParticipantFactory
Connection start()	DomainParticipant enable()
Session	Publisher, Subscriber
	FooTypeSupport extends TypeSupport Used to register a user type ‘Foo’ with a DomainParticipant
Destination A named physical resource that gathers and disseminates messages addressed to it	Topic (of type “Foo”) An abstraction with a unique name, data-type, and QoS, used to connect matching DataWriters and DataReaders
Message	Foo (data-object) An instance of type ‘Foo’
MessageProducer	FooDataWriter extends DataWriter
MessageConsumer	FooDataReader extends DataReader

JMS-DDS Semantic Mapping: Differences

	JMS	DDS
Data Modeling	Autonomous messages, Message-oriented, Payload is opaque	Data-objects, Data-centric, Relational data model
Dataflow Routing	Specific destinations	Matching Topics and Compatible QoS
Discovery	Administered	Spontaneous, Access to endpoint meta-data
Data Typing	Predefined message types	Arbitrary user data types

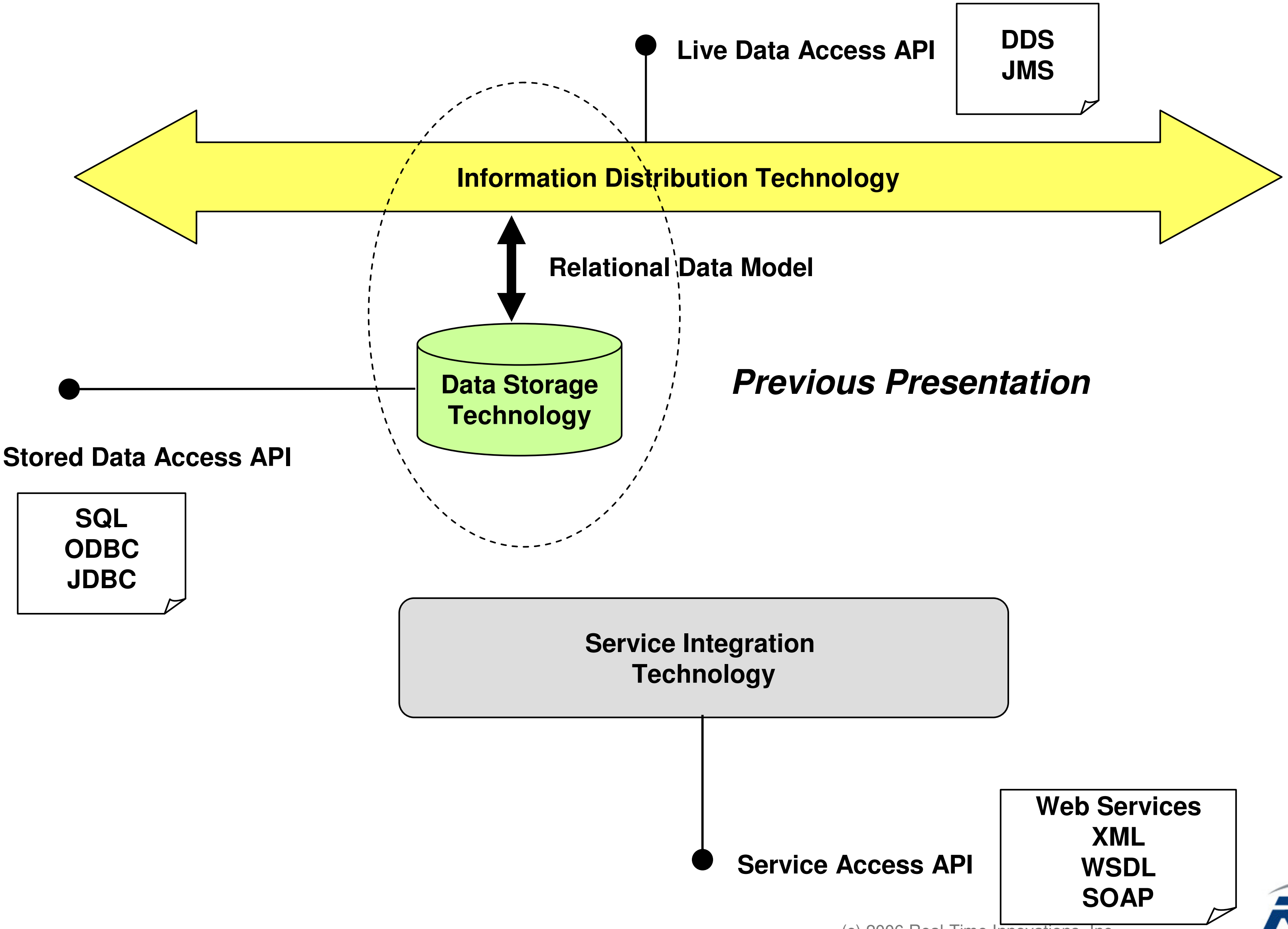
JMS and DDS: Practical Considerations

	JMS	DDS
Architecture	Client/Server, Centralized, Asymmetric	Peer-to-Peer, Decentralized, Symmetric
Platforms	Java	Multiple languages (C, C++, Java, …) & platforms
Interoperability	None. No plans.	Wire protocol standardization in progress
Transports	Connection oriented. Requires in-order acknowledged delivery.	Connection-less unreliable transport like UDP
Security	(Username, Password)	Extensible mechanism
Administration	JNDI, External to JMS	None: plug-n-play
Performance		Lower latency, higher throughput. Independent 10x better claims
Scalability	Limited by: PtP, Destinations	
Real-Time support		Many capabilities designed for real-time, low latency apps
Enterprise support	Transactions, Application acknowledgements, Java EE	

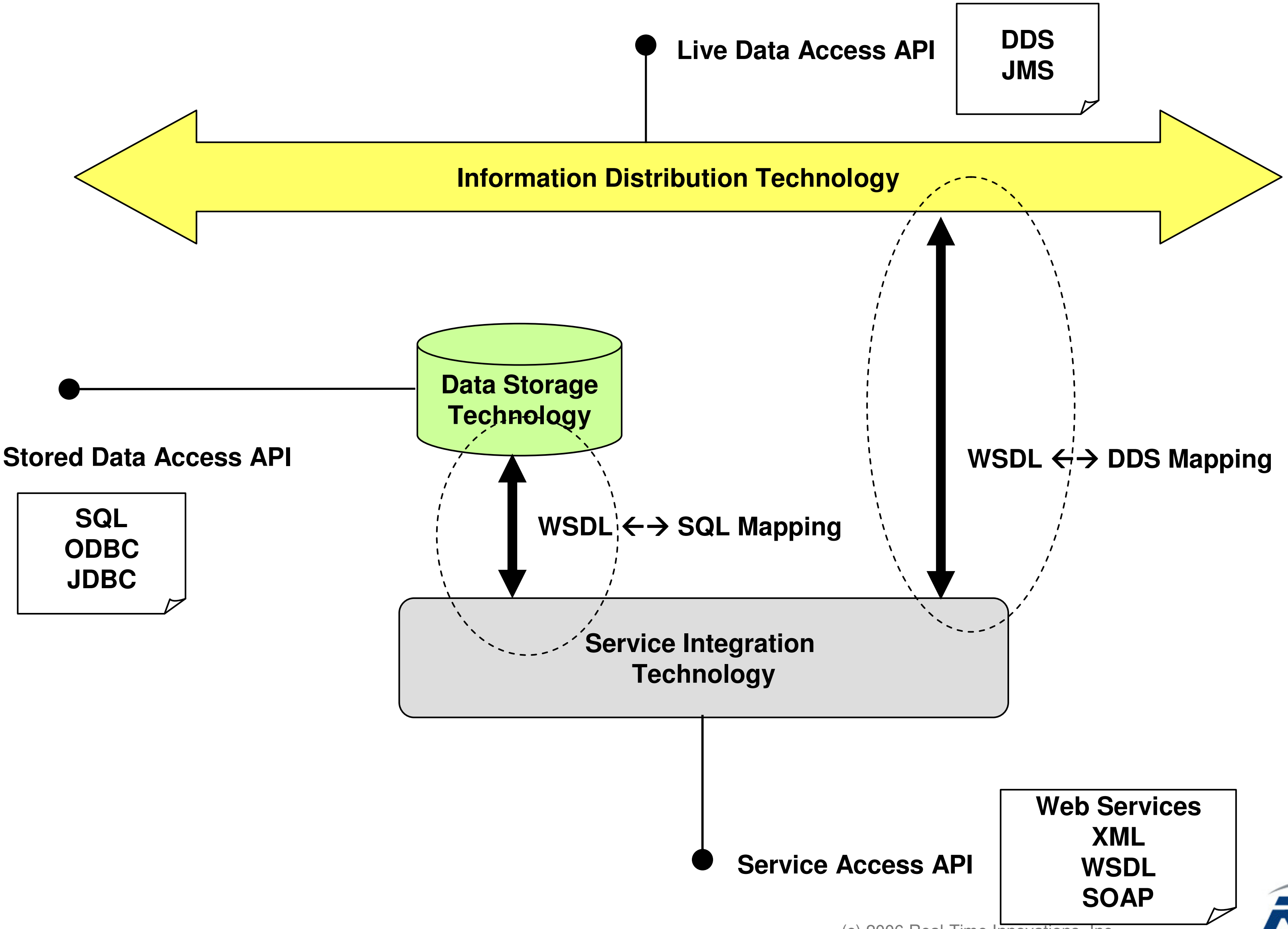
Using DDS and JMS together

- JMS-DDS Bridging
  - Translate between DDS updates ←→ JMS messages
  - Data conversion overhead
  - Least common denominator
  - Integrate different sub-systems
- JMS/DDS Bindings
  - Wrap DDS middleware with JMS APIs
  - Port JMS applications to DDS
  - JMS apps interoperable with DDS
  - Higher performance JMS

Data Storage: Integrating DDS and SQL



Service Integration: Integrating DDS and Web Services



Live Data Access API

DDS
JMS

Information Distribution Technology

Data Storage Technology

Stored Data Access API

SQL
ODBC
JDBC

WSDL ←→ SQL Mapping

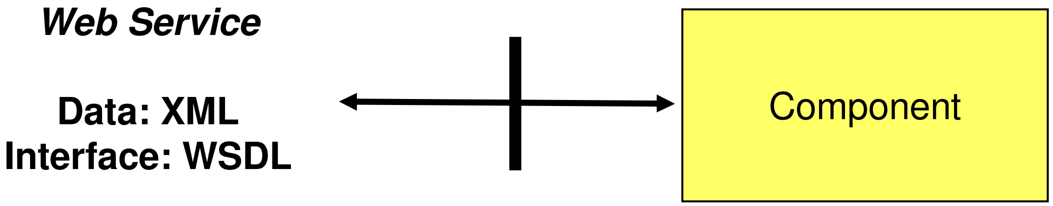
WSDL ←→ DDS Mapping

Service Integration Technology

Service Access API

Web Services
XML
WSDL
SOAP

What is a Web Service?



Web Service

Data: XML
Interface: WSDL

Component

- W3C Definition
  - A software system designed to support interoperable machine-to-machine interaction over a network
- Service Data Model: XML Schema
  - XML = “Extensible Markup Language”
  - XML Schema for describing XML documents (Data Types)
- Service Contract: WSDL
  - “Web Services Description Language”
  - Defines all details about a service contract (Interfaces)
- Encoding of Messages: SOAP
  - “Simple Object Access Protocol”
  - How to format XML documents for transmission between applications (Serialize/Deserialize)

Family of Extensible Specifications

An Example Web Service

Aircraft Tracking Service

Data: XML
Interface: WSDL

Aircraft Tracking Component

- **Service Data Model**
  - **TrackData**
    - **x**
    - **y**
    - **z**

XML Schema

```
<complexType name=“TrackData”>
    <element name =“x” type=“xsd:double”>
    <element name =“y” type=“xsd:double”>
    <element name =“z” type=“xsd:double”>
</complexType>
```

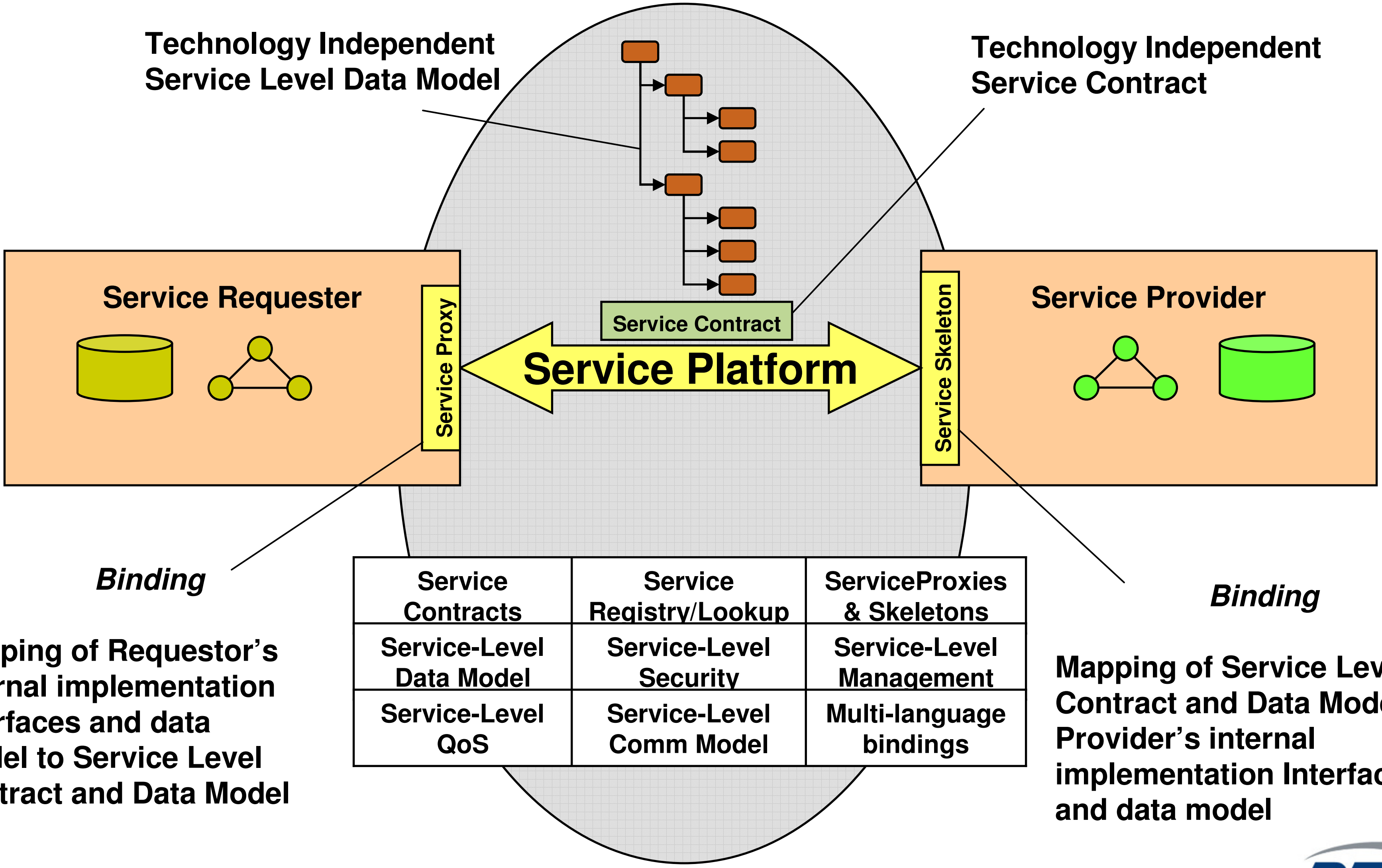
- **Service Contract**
  - **GetPosition**
    - **Message In**
      - **Aircraft Id**
    - **Message Out**
      - **TrackData**

WSDL

```
<portType name=“AircraftTrackingServicePort”>
  <operation name=“GetPosition”>
    <input message=“AircraftIdMessage” name=“aircraftId”>
    <output message=“TrackDataMessage” name=“trackData”>
  </operation>
</portType>

<message name=“AircraftIdMessage”>
  <part element=“xsd:int” name=“aircraftId”/>
</message>
<message name=“TrackDataMessage”>
  <part element=“TrackData” name=“trackData”/>
</message>
```

A Web Service is Just a Specification



Technology Independent Service Level Data Model

Technology Independent Service Contract

Binding

Mapping of Requestor’s Internal implementation Interfaces and data model to Service Level Contract and Data Model

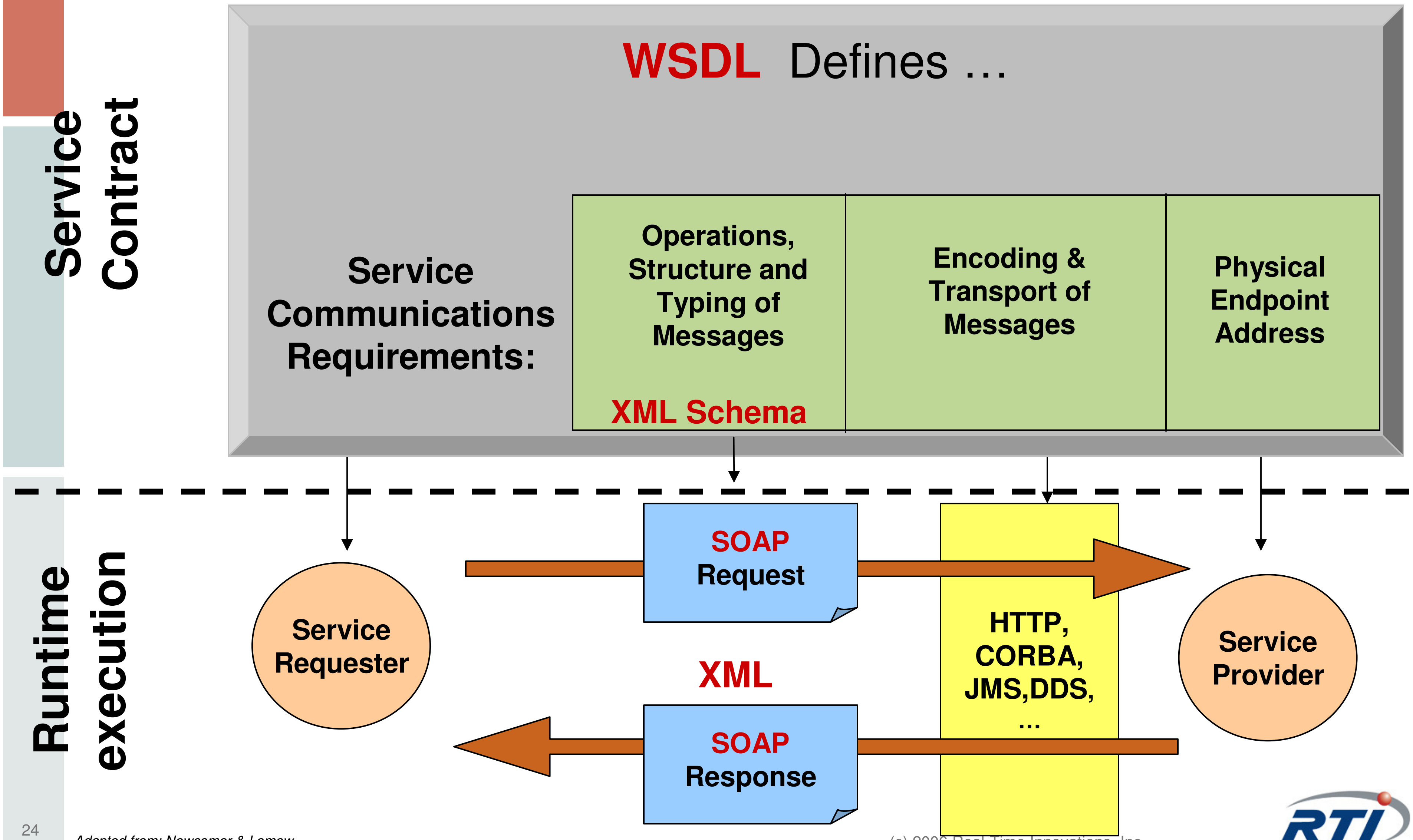
Service Contracts	Service Registry/Lookup	ServiceProxies & Skeletons
Service-Level Data Model	Service-Level Security	Service-Level Management
Service-Level QoS	Service-Level Comm Model	Multi-language bindings

Binding

Mapping of Service Level Contract and Data Model to Provider’s internal implementation Interfaces and data model

Adapted from: Newcomer & Lomow

Binding a Web Service to a Transport



Service Contract

WSDL Defines …

Service Communications Requirements:

Operations, Structure and Typing of Messages

XML Schema

Encoding & Transport of Messages

Physical Endpoint Address

Runtime execution

Service Requester

SOAP Request

XML

SOAP Response

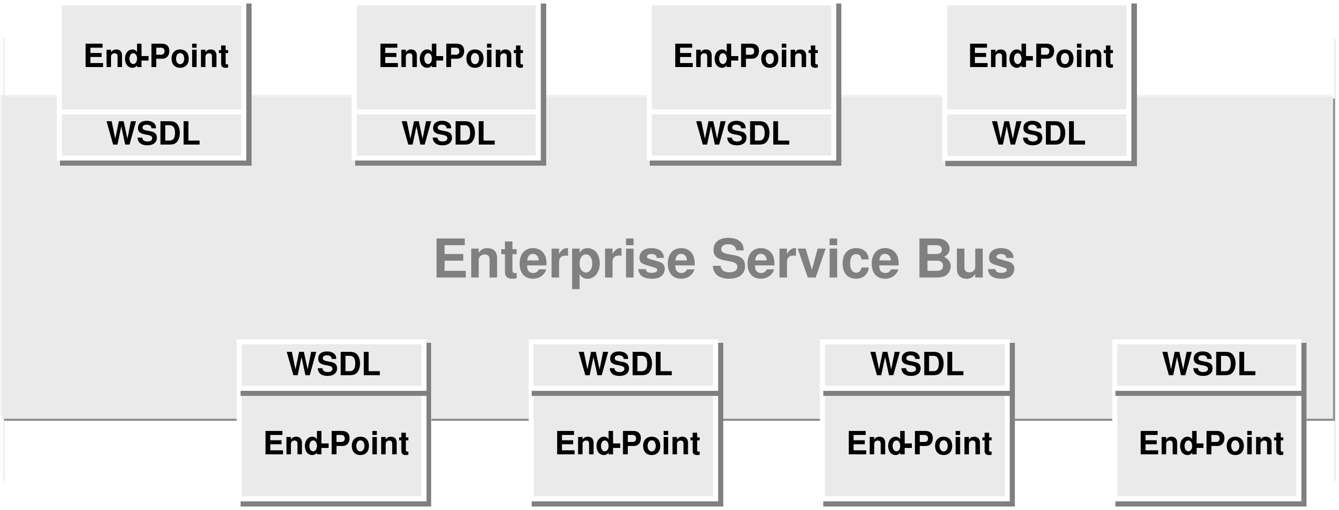
HTTP, CORBA, JMS,DDS, …

Service Provider

Adapted from: Newcomer & Lomow

Enterprise Service Bus (ESB)

The Enterprise Service Bus (ESB) Pattern



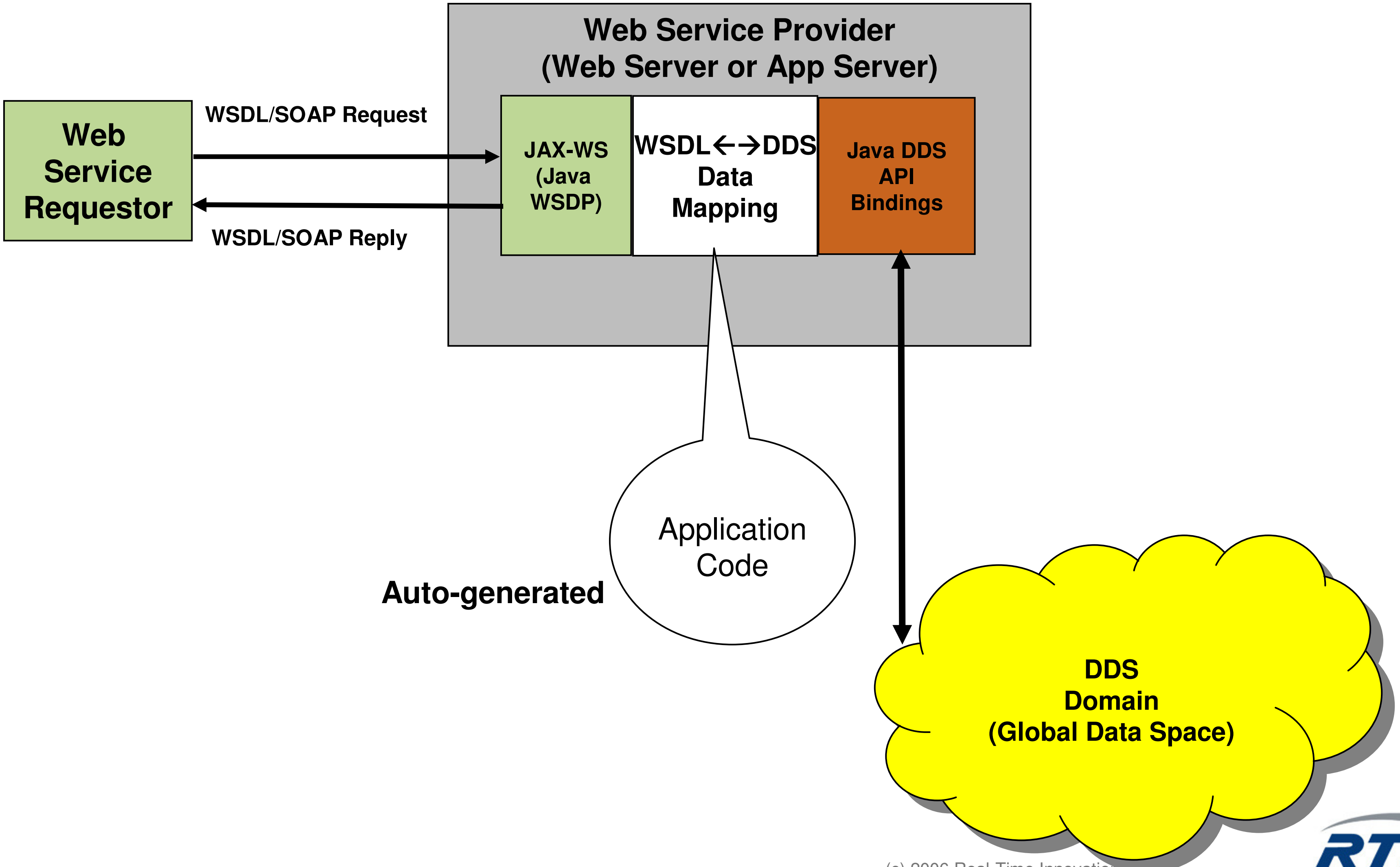
- Collection of distributed, interconnected end-points
- Support multiple transports / middleware
- No internal canonical data format
- Instead: support transformation and routing

Adapted from: Iona, “Next Generation ESBs’

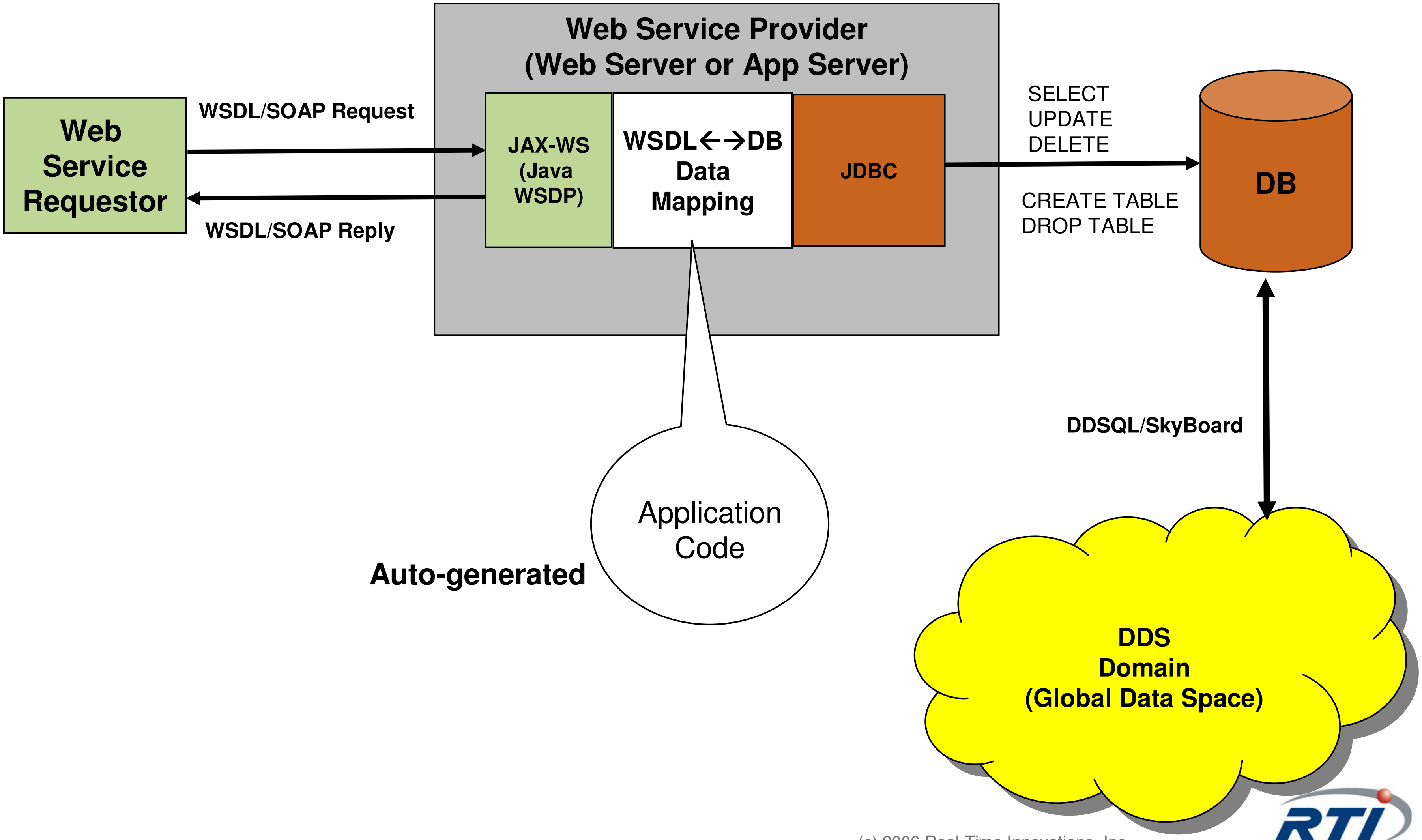
Enabling the Real-Time ESB

- Using DDS as a Transport for a Web Service
  - WSDL/DDS Bindings
  - WSDL/SQL Bindings
    - Using DDS-SQL Integration to access the global data space
- Exposing DDS as a Web Service
  - Exposes the underlying relational data model

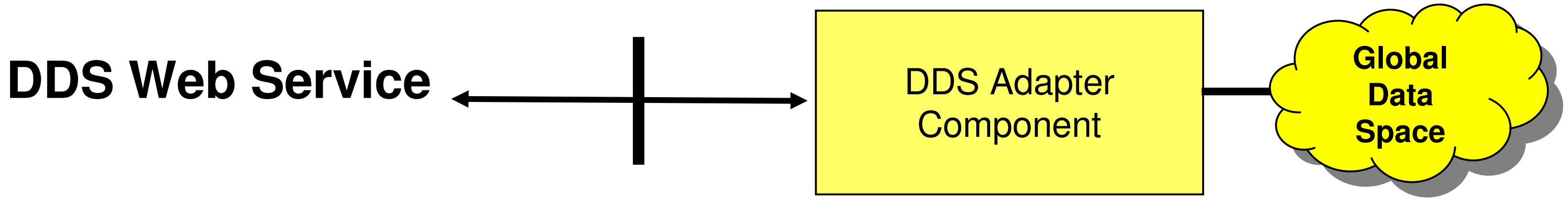
Using DDS as a Web Service Transport



Using DDS-SQL Integration as a Web Service Transport



DDS as a Web Service: Expose the Data Model



- Service Data Model
  - ParticipantInfo
  - DataTypeInfo
  - TopicInfo
  - PublisherInfo
  - DataWriterInfo
  - SubsriberInfo
  - DataReaderInfo
  - DataReaderListenerInfo
  - :

```
                    XML Schema Example
<complexType name=“ParticipantInfo”>
    <element name =“domain_id” type=“xsd:int”>
</complexType>
:
```

- Service Contract
  - CreateParticipant
  - RegisterType
  - CreateTopic
  - CreatePublisher
  - CreateDataWriter
  - CreateSubscriber
  - CreateDataReader
  - SetDataReaderListener
  - :

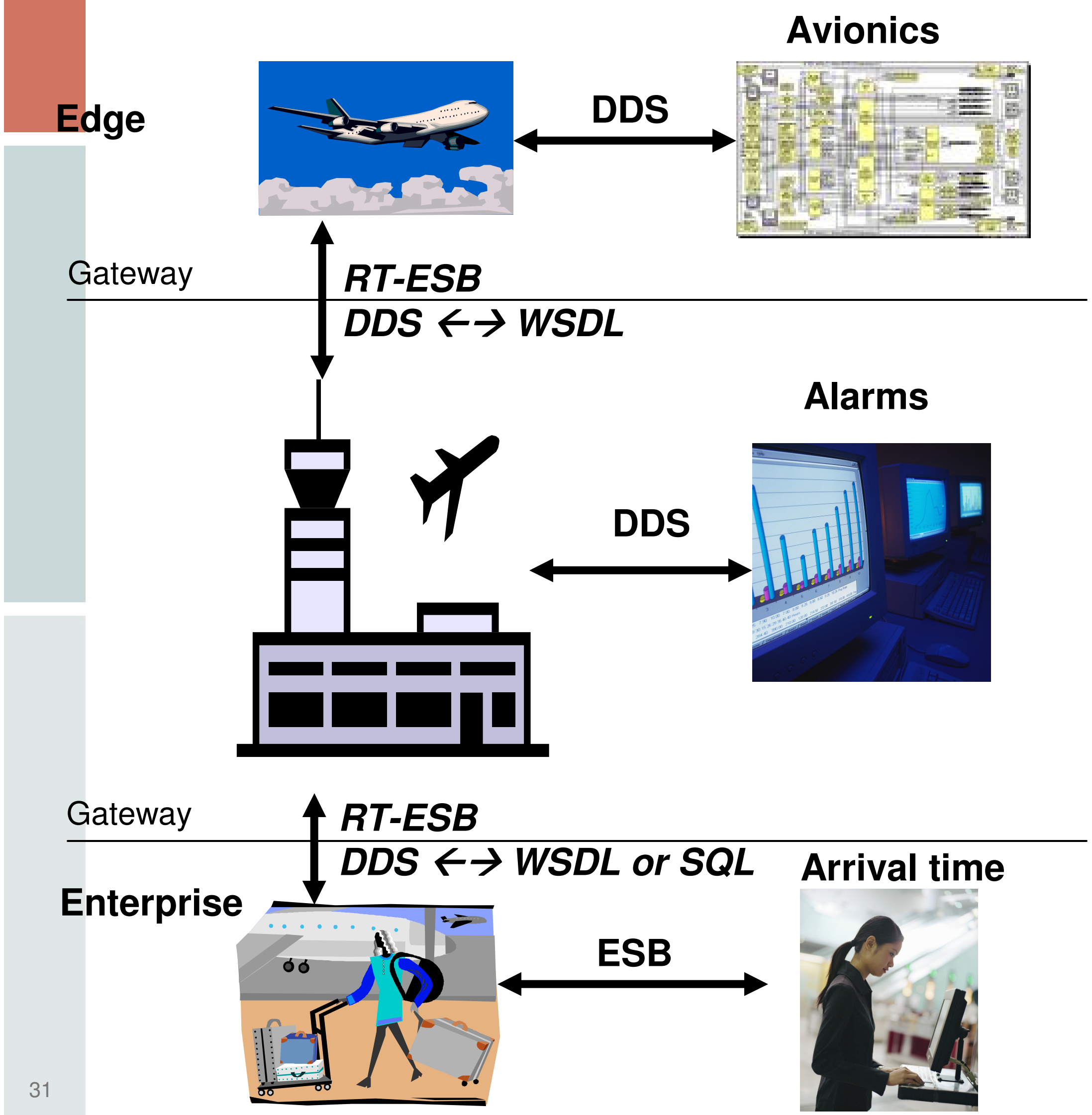
```
                          WSDL Example
<portType name=“DDSPort”>
  <operation name=“CreateParticipant”>
    <input message=“ParticipantInfoMessage” name=“participantInfo”>
    <output message=“URIMessage” name=“participantURI”>
  </operation>
   :
</portType>

<message name=“ParticipantInfoMessage”>
  <part element=“ParticipantInfo” name=“participantInfo”/>
</message>
```

New Possibilities using the Real-Time ESB

- Bridge the gap between the Edge and the Enterprise
  - Resolve the impedance mismatch
    - Real-Time Database Cache
  - Exchange data between the edge and the enterprise
  - Access enterprise data from embedded devices
- Integrate with disparate technologies
  - Leverage integration of other technologies with Web Services
  - Realize the promise of an “ESB” from the edge to the enterprise
- Real-Time SOA
  - Enable high performance or real-time Web Services
  - Using DDS as a transport for a Web Service
- Remote monitoring and control of DDS applications
  - Using DDS as a Web Service

Bridging the Gap: Air traffic control example redux



Edge

Avionics

DDS

Gateway

RT-ESB

DDS ←→ WSDL

Alarms

DDS

Gateway

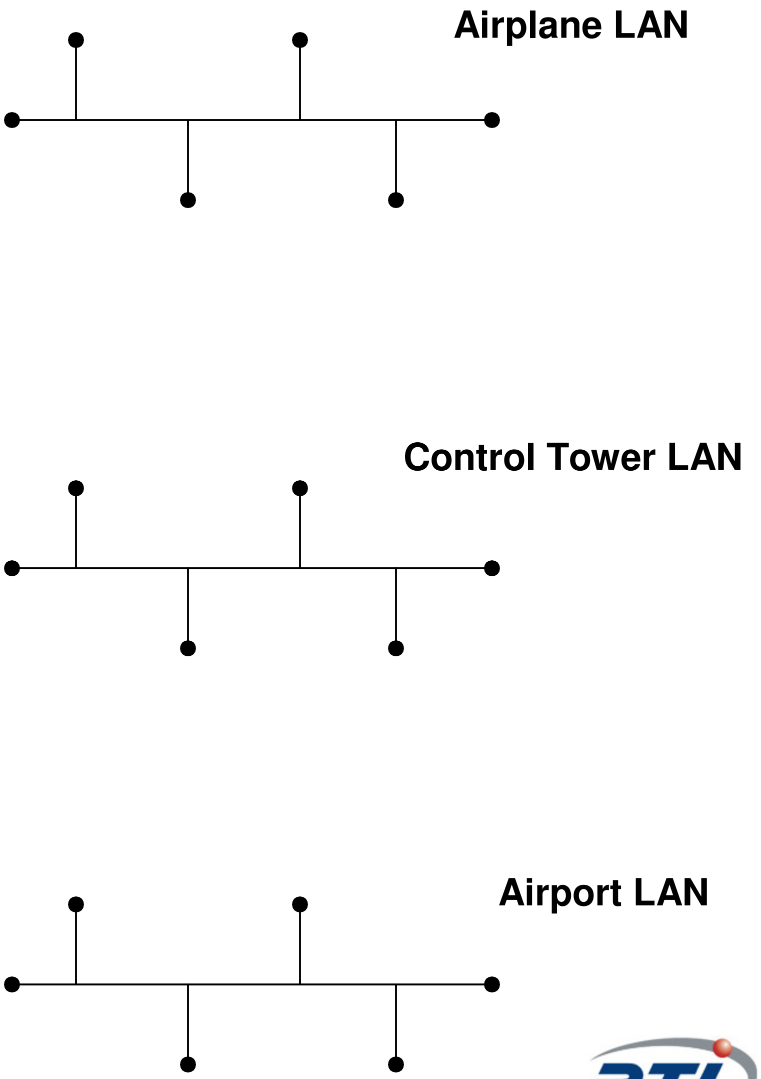
RT-ESB

DDS ←→ WSDL or SQL

Enterprise

Arrival time

ESB

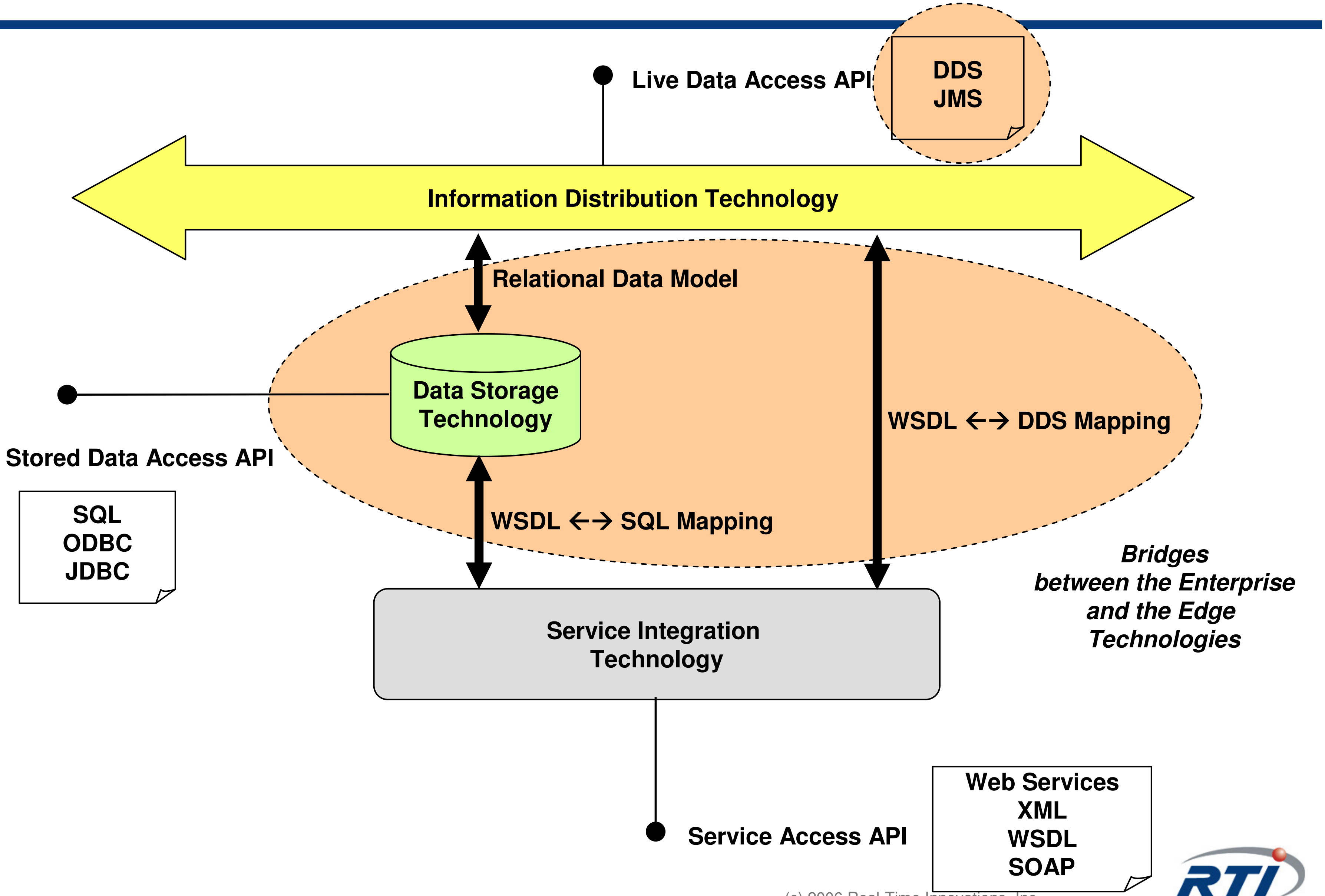


Airplane LAN

Control Tower LAN

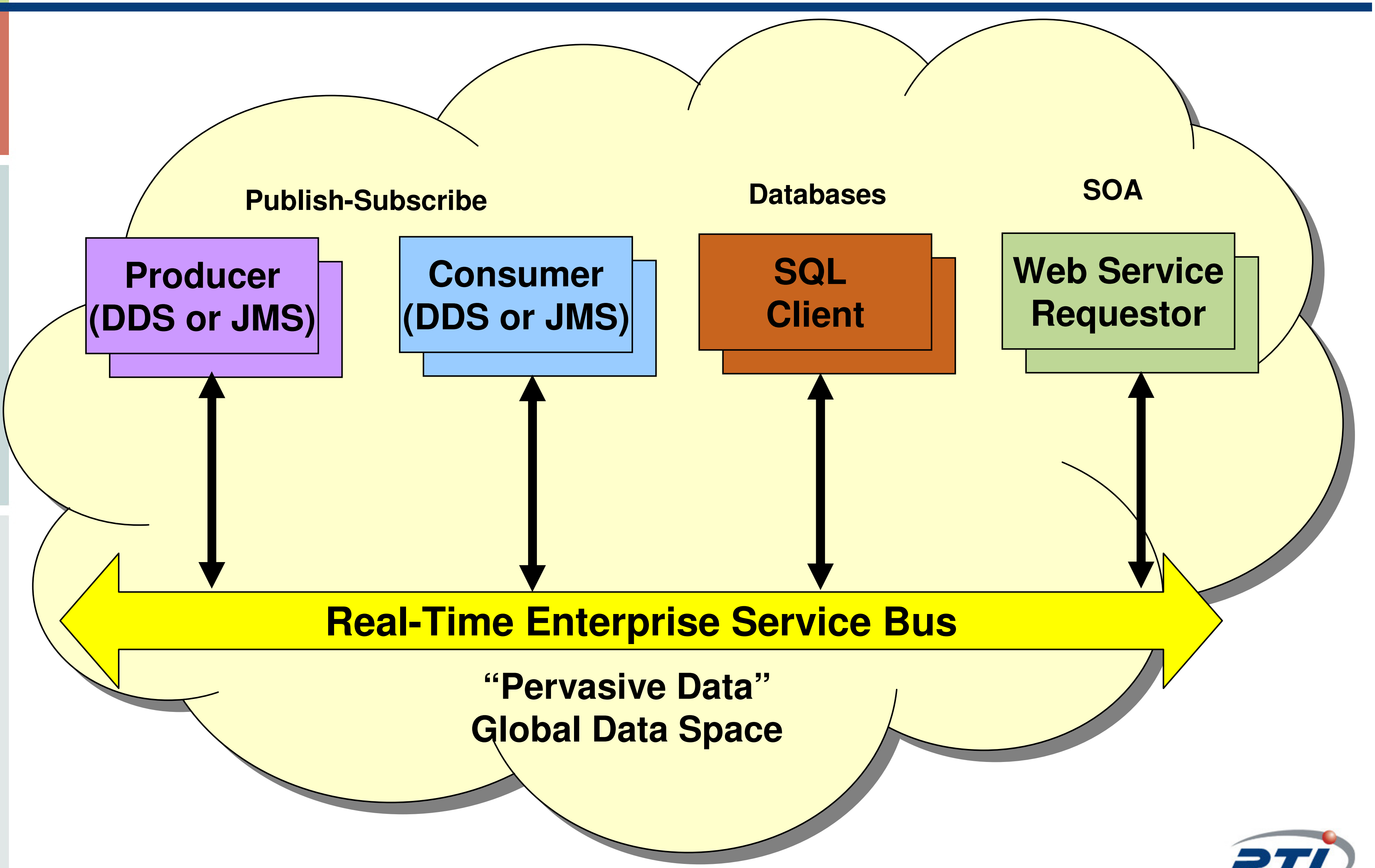
Airport LAN

Bridging the Gap: Integrating DDS with JMS, SQL, Web Services



Bridges between the Enterprise and the Edge Technologies

The Real-Time Enterprise Service Bus



Thank You

Q & A

Rajive.Joshi@rti.com

Real-Time Innovations, Inc

For more Information:

- www.rti.com
  - **RTI Whitepaper: A Comparison and Mapping of Data Distribution Service (DDS) and Java Message Service (JMS)**