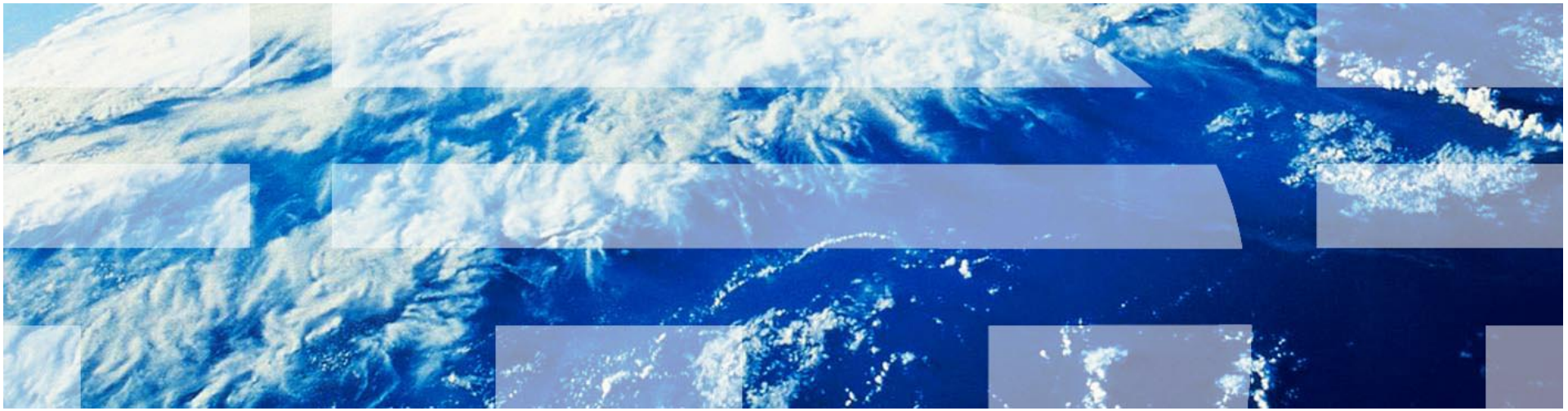
Introduction

Discovering the value of App Connect Enterprise v11

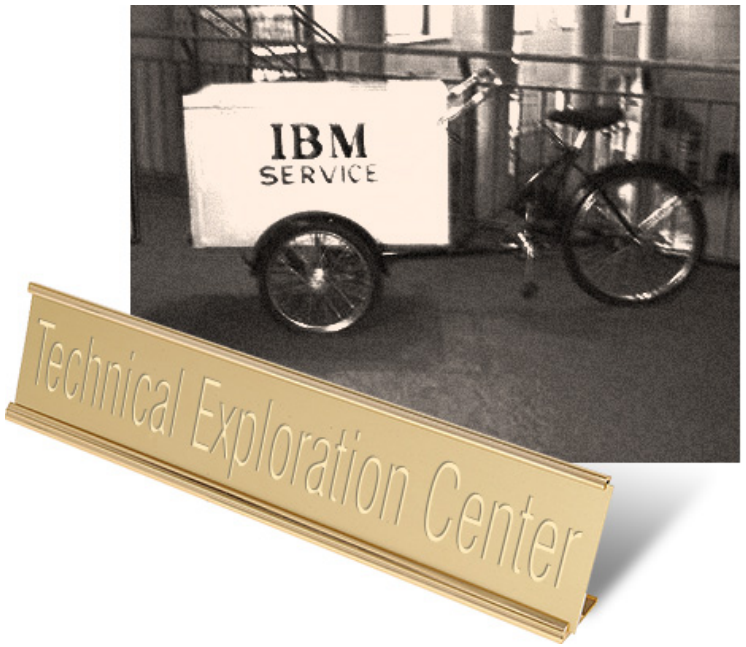


Agenda

- 09:30 - 09:45 Welcome / Intro
- 09:45 - 10:30 Introduction To App Connect Enterprise
- 10:30 - 11:10 Lab 1 - Applications and libraries, developing a simple application
- 11:10 - 11:45 Lab 2 - Message parsing, the logical message model, content-based routing
- 11:45 - 12:30 - LUNCH BREAK
- 12:30 - 13:00 Lab 3 - Patterns and file processing
- 13:00 - 13:30 Lab 4 - Message modeling with DFDL
- 13:30 - 14:00 Lab 5 - Create a REST API
- 14:00 - 14:15 COFFEE BREAK
- 14:15 - 15:00 Lab 6 - Exception handling and debugger
- 15:00 - 15:30 Administration - overview
- 15:30 - 16:00 IBM Safer Payments - detalji implementacije na ACE (Tea Milić)
- 16:00 - 16:15 Summary and wrap-up

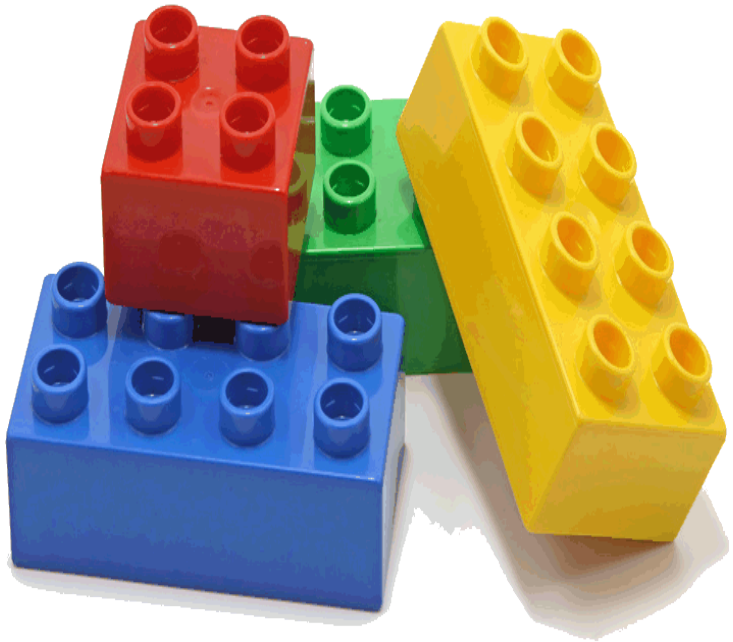
Introductions

- Please introduce yourself
- Name and organization
- Current integration technologies/tools in use

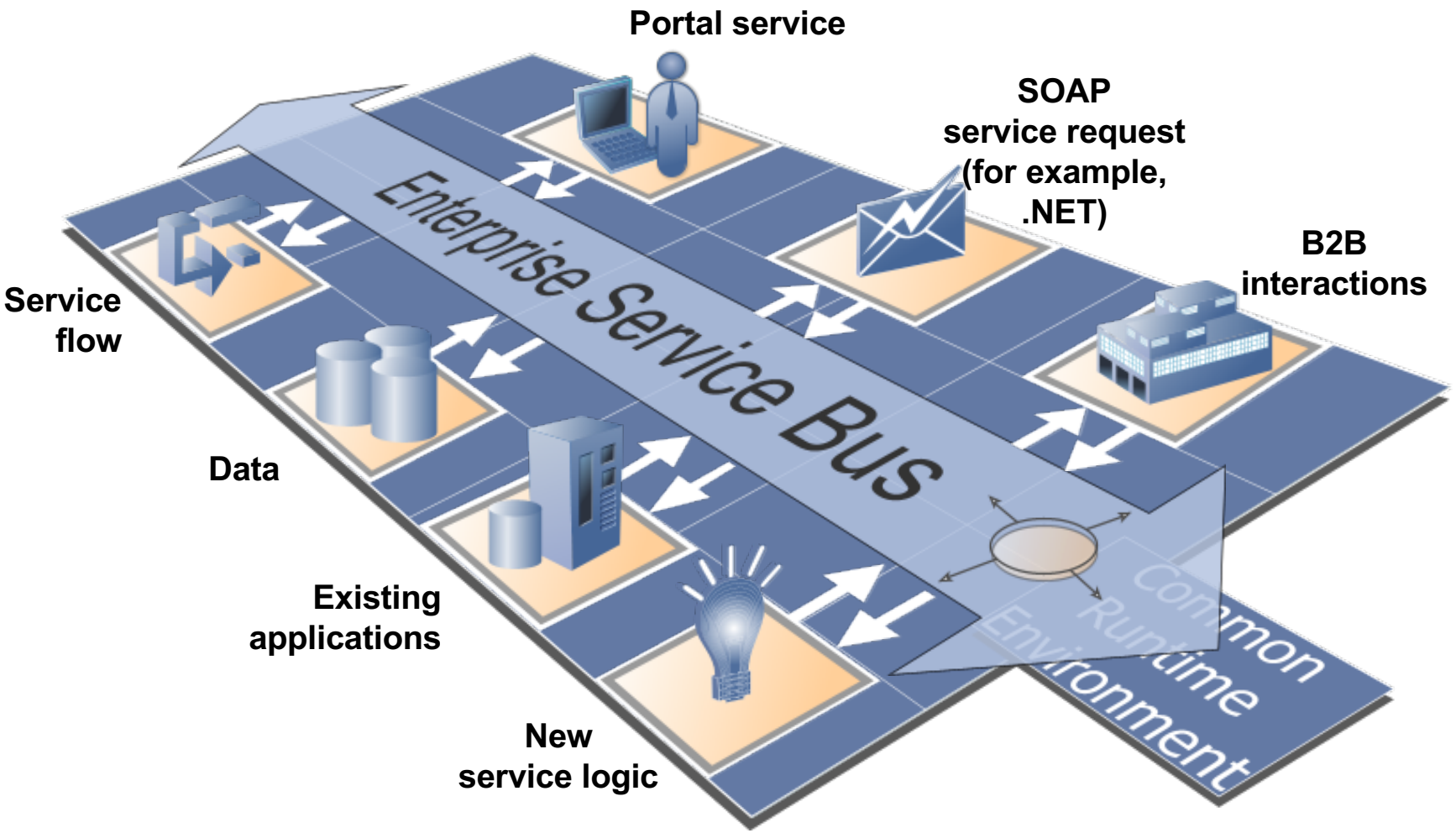


What do you want out of this Exploration session?

The Basics



Enterprise Service Bus (ESB) – architectural concept

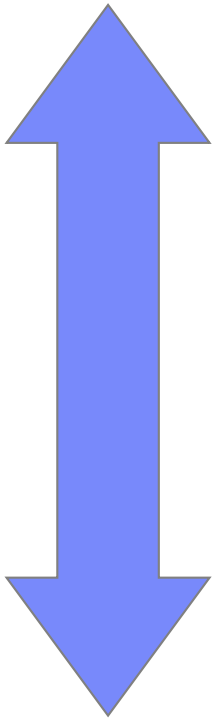


What do we mean by integration?

- Enterprise systems consist of many logical *endpoints*
  - Off-the-shelf applications, services, web apps, devices, appliances, custom-built software…
- Endpoints expose a set of inputs and outputs, which comprise:
  - Protocols – for example, IBM® WebSphere® MQ, TCP/IP, HTTP, File system, FTP, SMTP, POP3 and others
  - Message formats – for example, Binary (C/COBOL), XML, Industry (SWIFT, EDI, HL7), user-defined
- Integration is about connecting these endpoints together in meaningful ways
  - Route, transform, enrich, filter, monitor, distribute, decompose, correlate, fire and forget, request or reply, publish or subscribe, aggregation, fan-in, complex event processing…

Integration usage patterns

- **Some common integration usage patterns**
  - Add **logging** to existing service
  - **Extend reach** of existing applications
  - Distribute **database** information to where it's needed
  - **File hub** to connect batch and online
  - Integrate **packaged applications**
  - Take advantage of **.NET applications**
  - Policy enforcement point for **secure connectivity**
  - Extend enterprise to **devices and mobile**
  - **Monitor** business activity and act intelligently
  - Detect and act upon **business events and rules**
  - Connectivity and integration for **business processes**
  - Enable p**olicy-based management**
- **New usage patterns continually emerging!**



Common usage patterns

Emerging usage patterns

A broad range of supported platforms and operating environments

- **Broad range of operating system and hardware platforms supported**
  - AIX®, Windows, Linux
  - Containers
    - Openshift, k8s
  - Public or private cloud
  - PaaS

- **Virtual images for efficient utilization and simple provisioning**
  - Extensive support for virtualized environments, e.g. VMWare, AIX Hypervisor

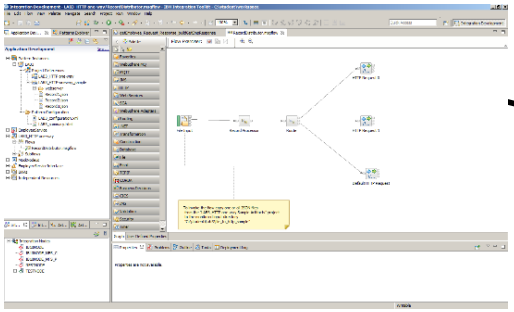
- **Includes access to full range of industry standard databases and ERP systems**
  - DB2, Oracle, Sybase, SQL Server, Informix, solidDB
  - Open Driver Manager support enables new ODBC databases to be accessed
  - JDBC Type 4 for popular databases
  - SAP, Siebel, Peoplesoft, JDEdwards at no additional cost

WebSphere. MQ

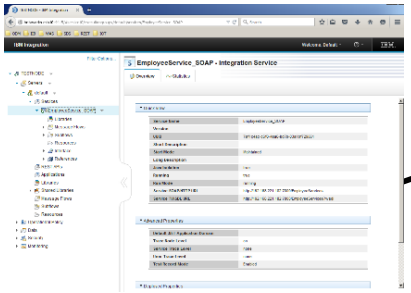
Components – What you get

	Developer workstation	Central Development, QA, Production
Integration Toolkit	X	
graphical development tool, Eclipse based		
Windows, Linux, Intel		
Integration Server	X	X
runtime engine		
various platforms		

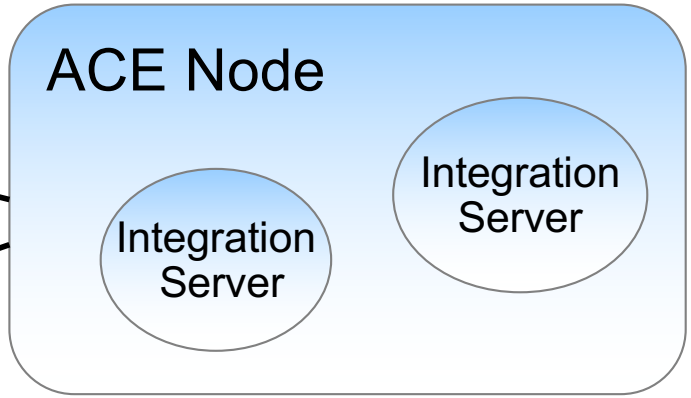
Architected for high performance and scalability



App Connect Enterprise Toolkit



Web Admin



- App Connect Enterprise Toolkit
  - Development and test environment
- Web Administration tool

- Integration node
  - Standalone runtime environment that runs message flows
  - Multiple Integration Servers allow isolation and scalability
  - Many different platforms

App Connect Enterprise Toolkit – Making Programming Easy

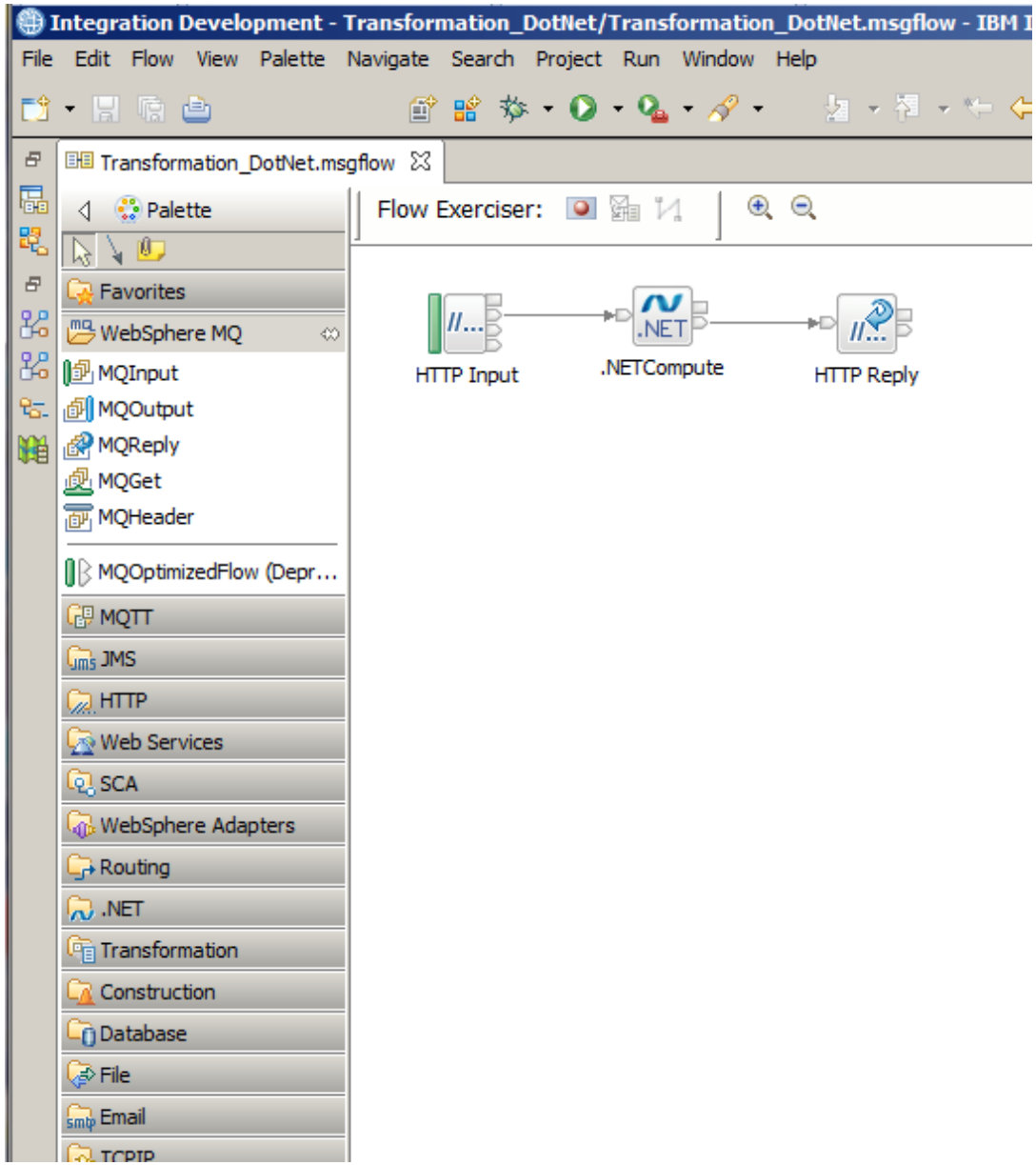
App Connect Enterprise is easy to learn

Simple basic concepts supporting powerful capabilities

Great samples for common scenarios
- Learn by practical example
- Quickly customize a working flow rather than starting from scratch

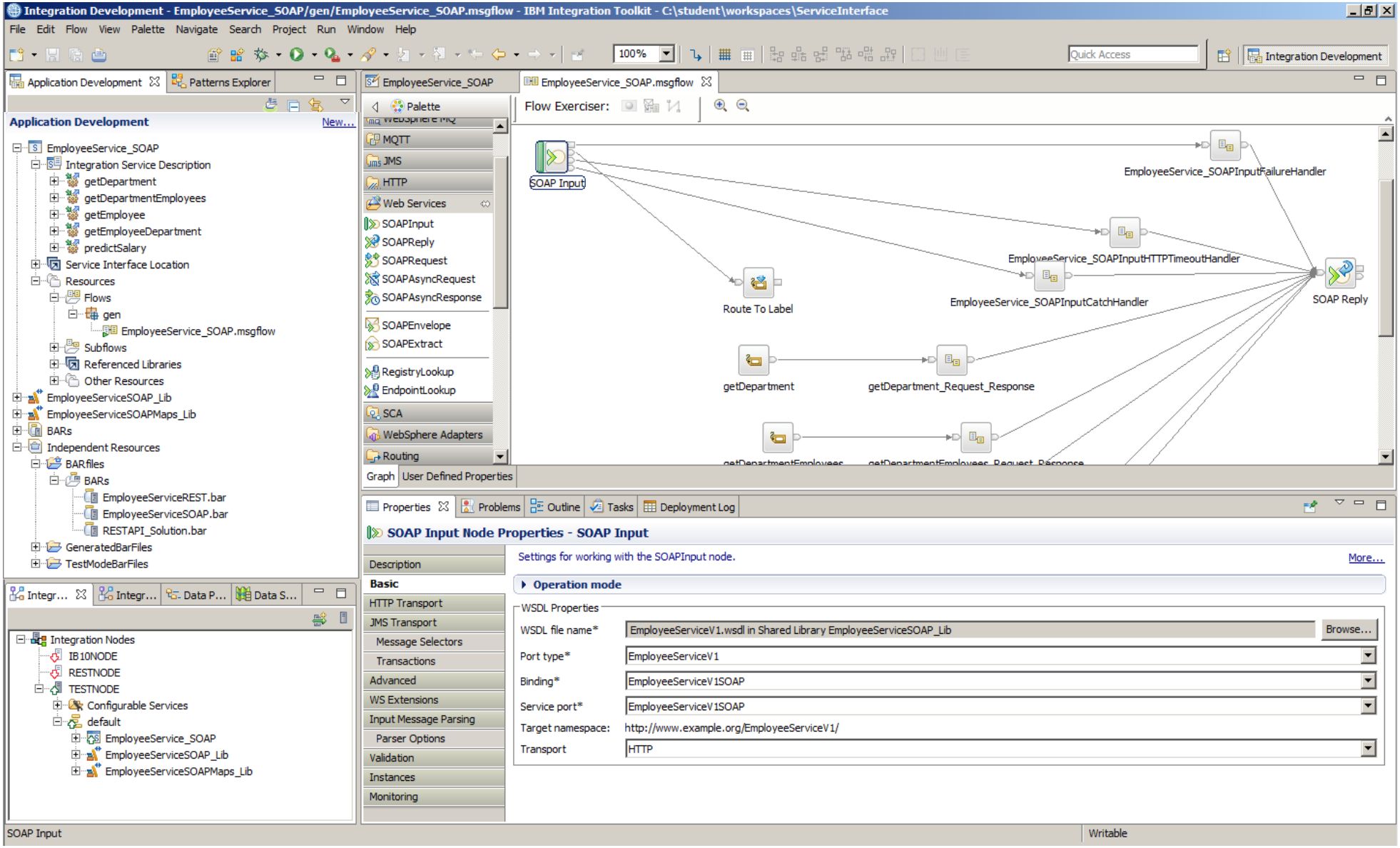
Graphical tools help first time users become quickly productive

App Connect Enterprise has two key building blocks
- **Nodes** connect to sources of data and perform route or transform operations
- **Flows** connect nodes together into processes that achieve desired business results

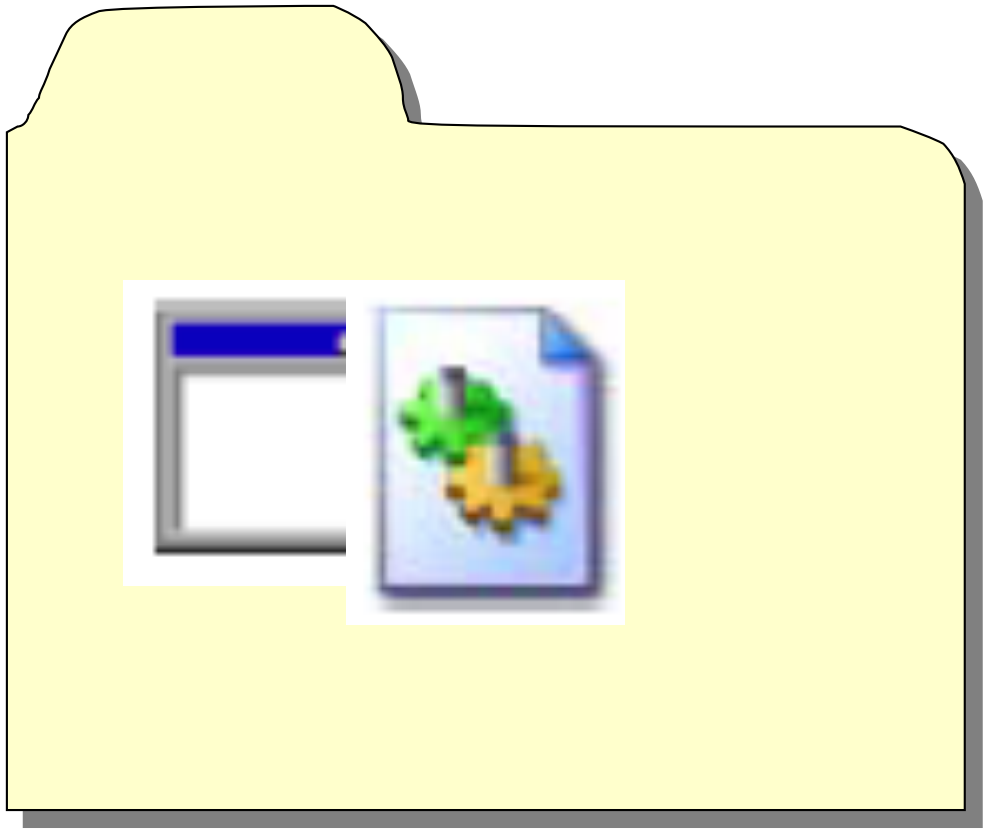


* ACE Toolkit runs on Windows and Linux

Easy to Develop - App Connect Enterprise Toolkit is visual and graphical

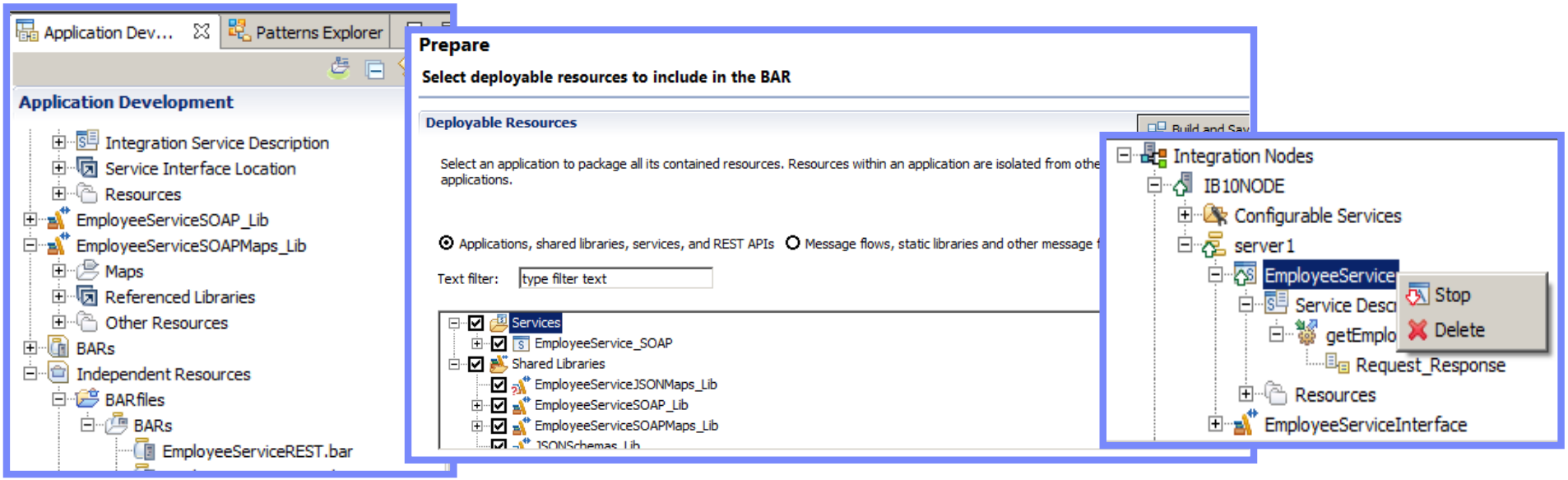


App Connect Enterprise development concepts



Applications, Libraries and Services

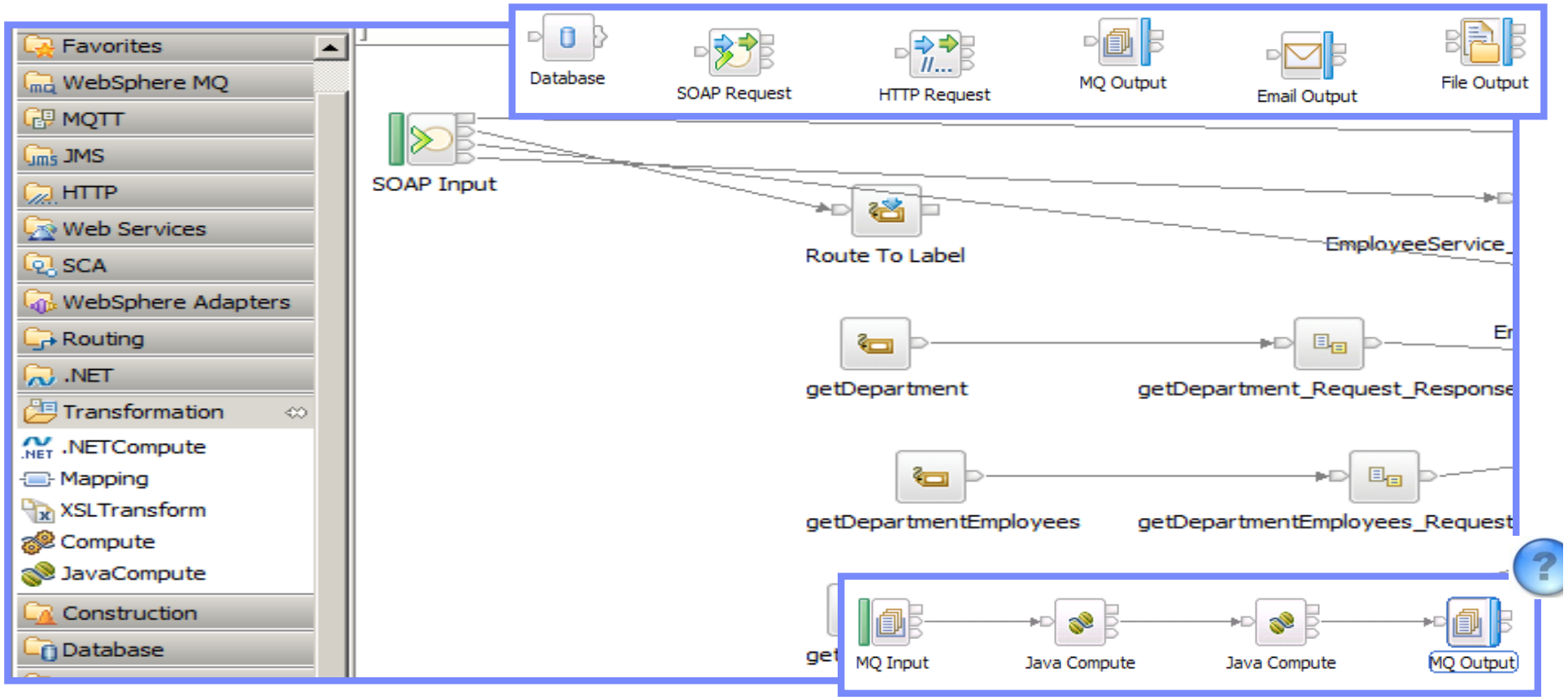
- *Applications* package end-to-end connectivity *solutions*
  - The concept of an application is shared between the toolkit and runtime
  - Applications are deployed and managed as a single unit of isolation
- *Services* are *Applications* with a well defined interface (*contract*)
  - The service interface is expressed with a WSDL and a port type
  - Service creation includes tooling to build the interface *and* structure
- *Libraries* package resources for *reuse* (flows, scripts, models)
- Resources in Application and Services are not visible to anything else
- Use this to manage your solutions inside an Integration Server



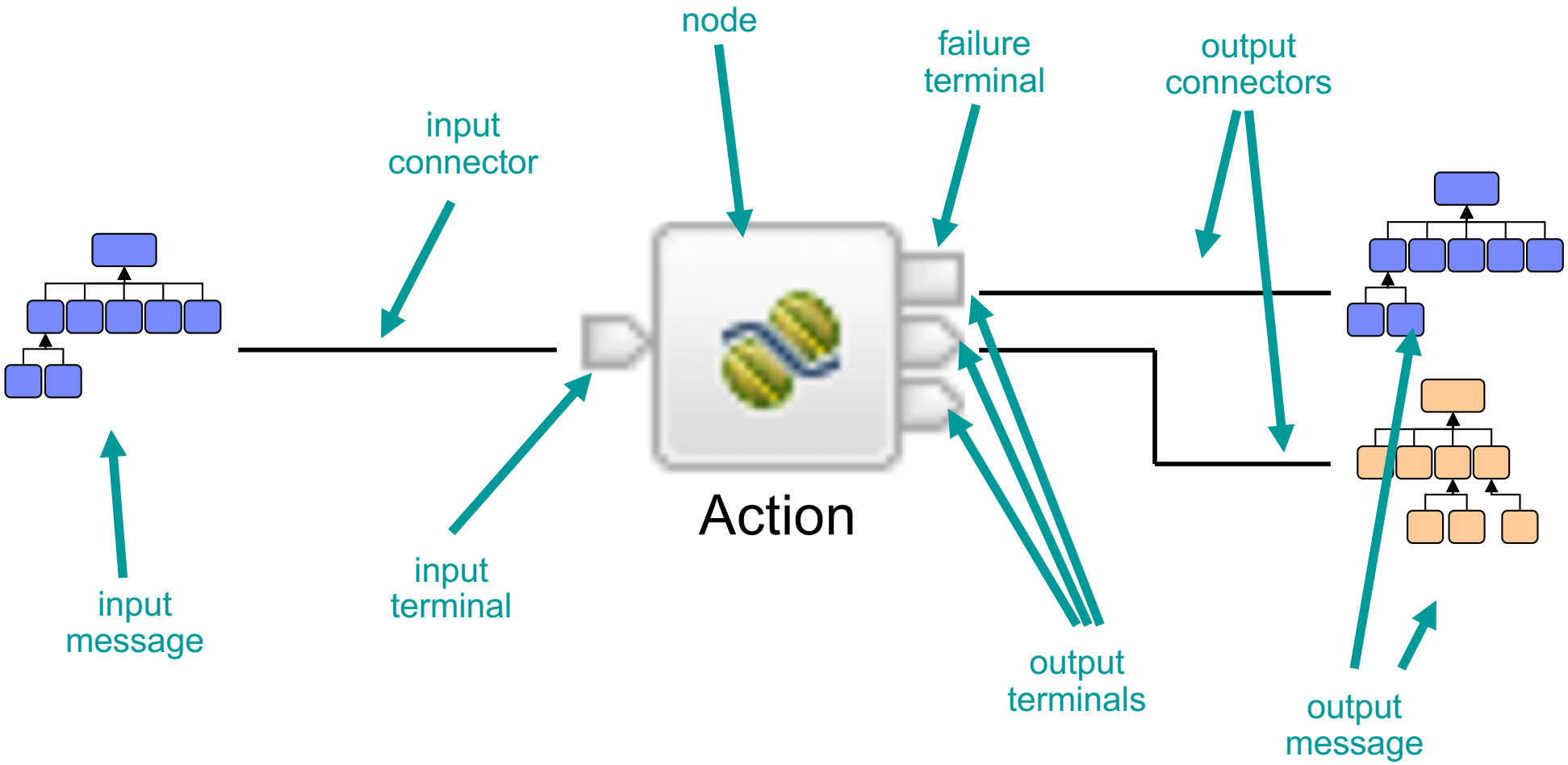
App Connect Enterprise uses a Node-Based Programming Model

Makes it Easy to Create Connectivity Solutions!

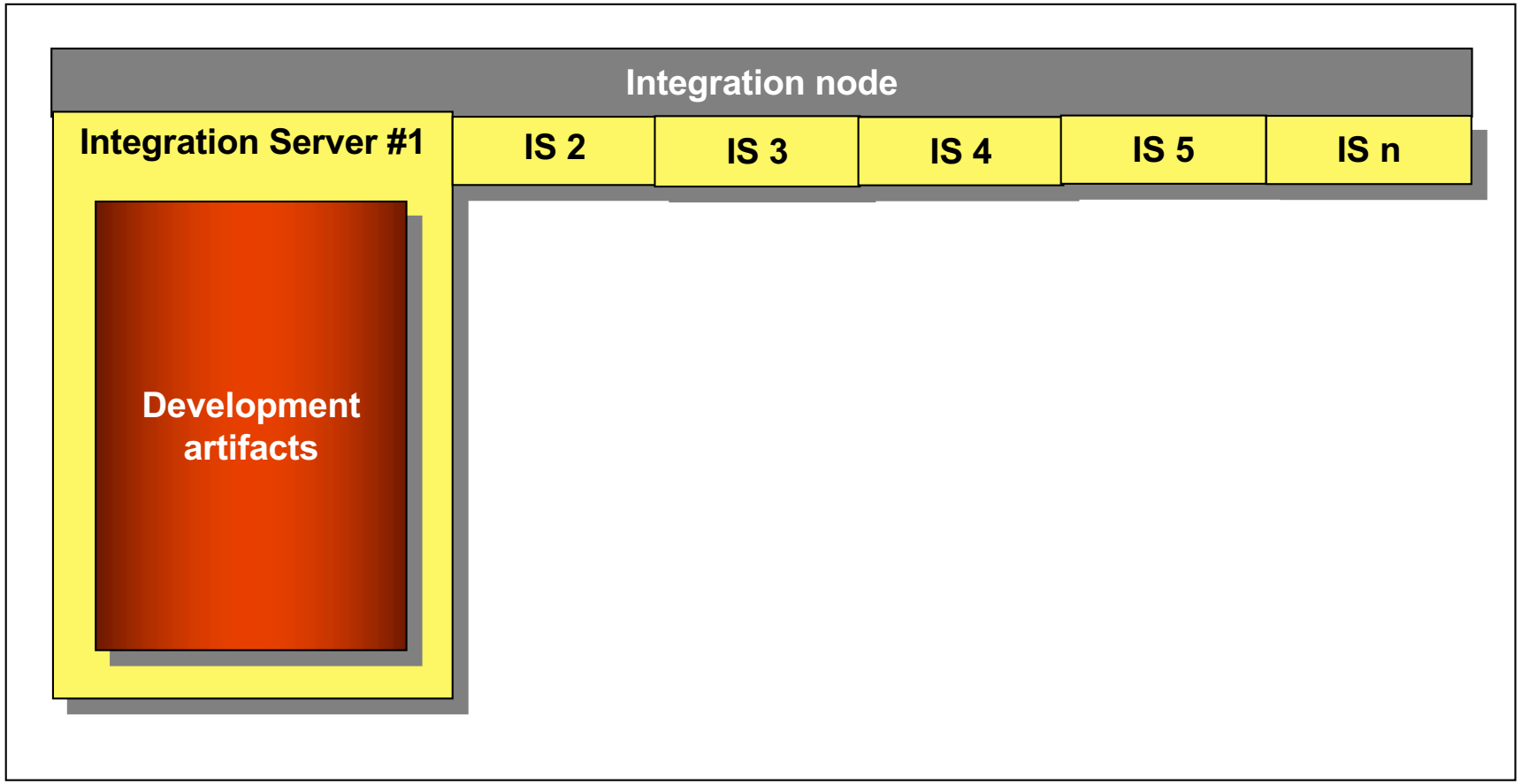
- Built-in nodes encapsulate transports, technologies and applications
  - Nearly 100 nodes available out-of-the-box!
  - Intent is to make the common tasks easy, and the rest possible!
  - Use the built-in nodes to reduce the amount of custom code required
  - Doing so makes best use of the built-in facilities like activity trace and resource statistics



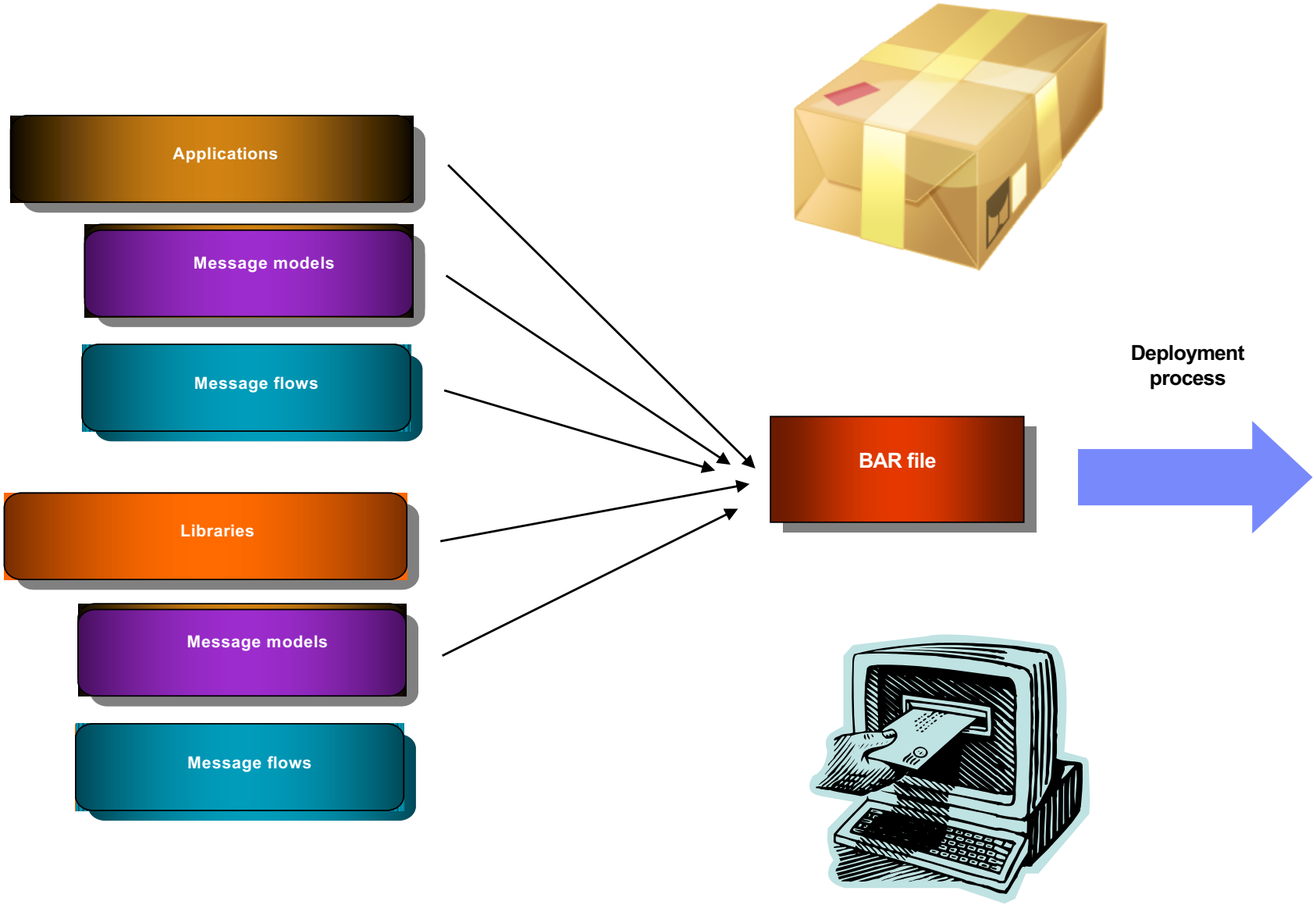
Nodes are reusable and consistent



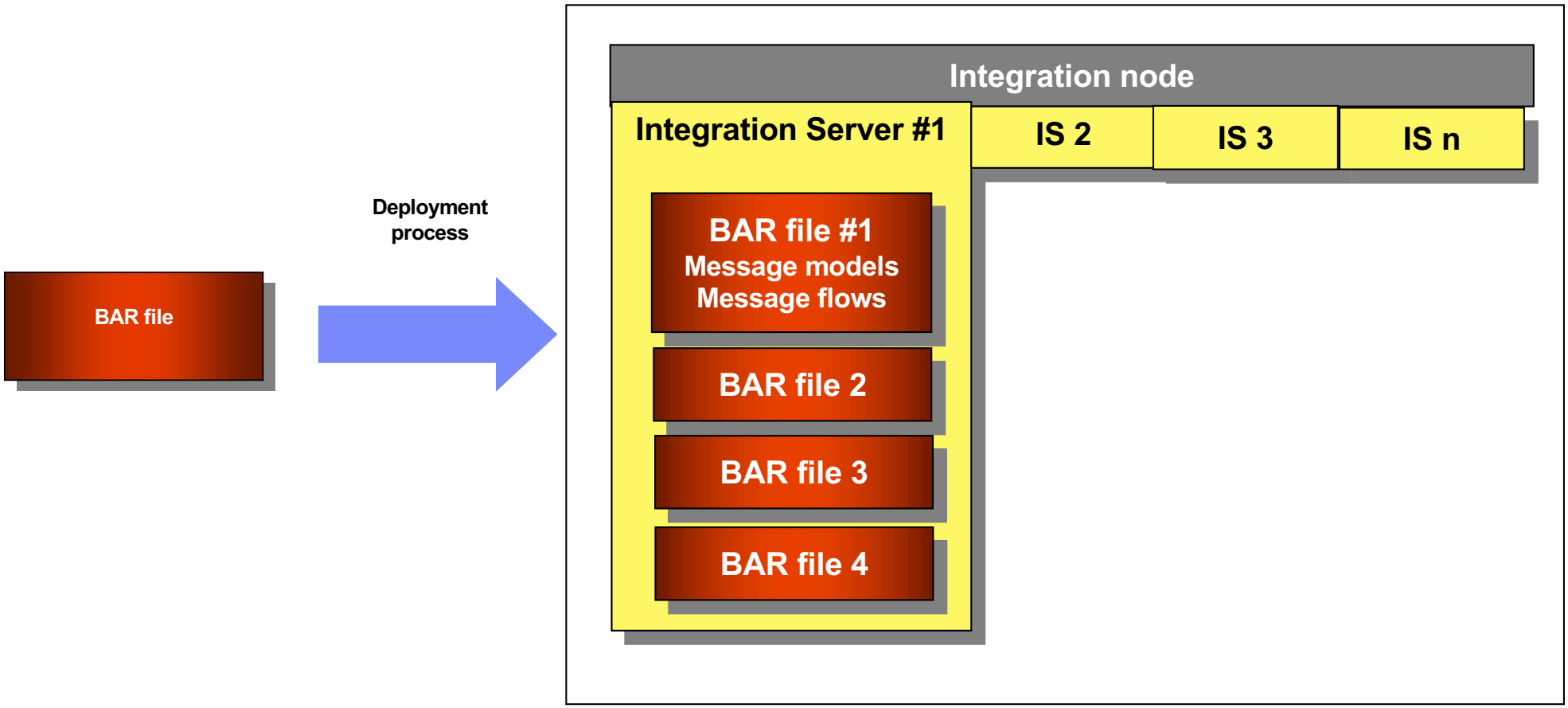
Integration Servers run your work



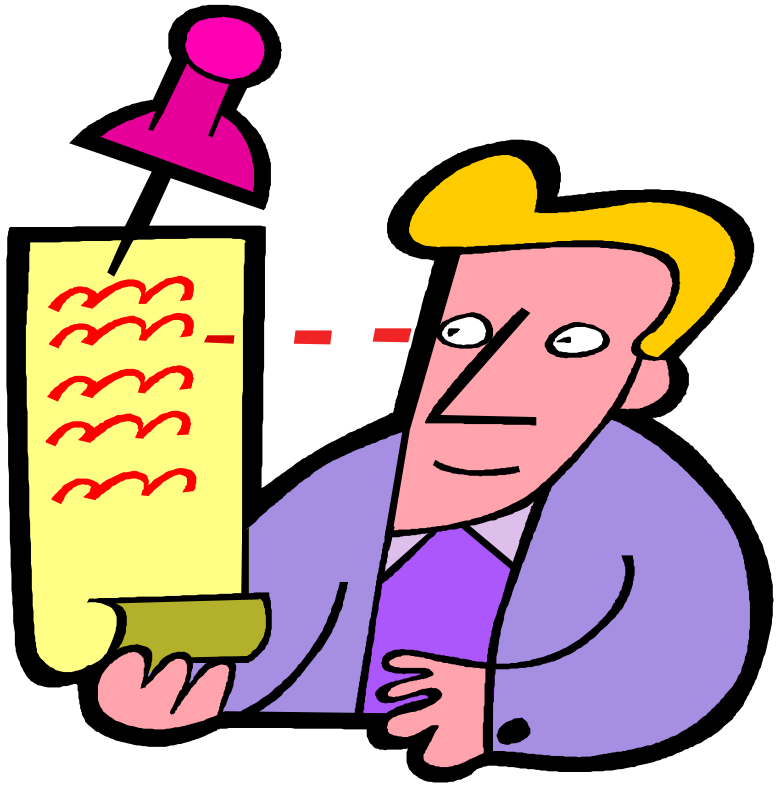
Development – BAR files and deployment



Development – BAR files and deployment



Understanding Data Modeling, Parsing and Validation

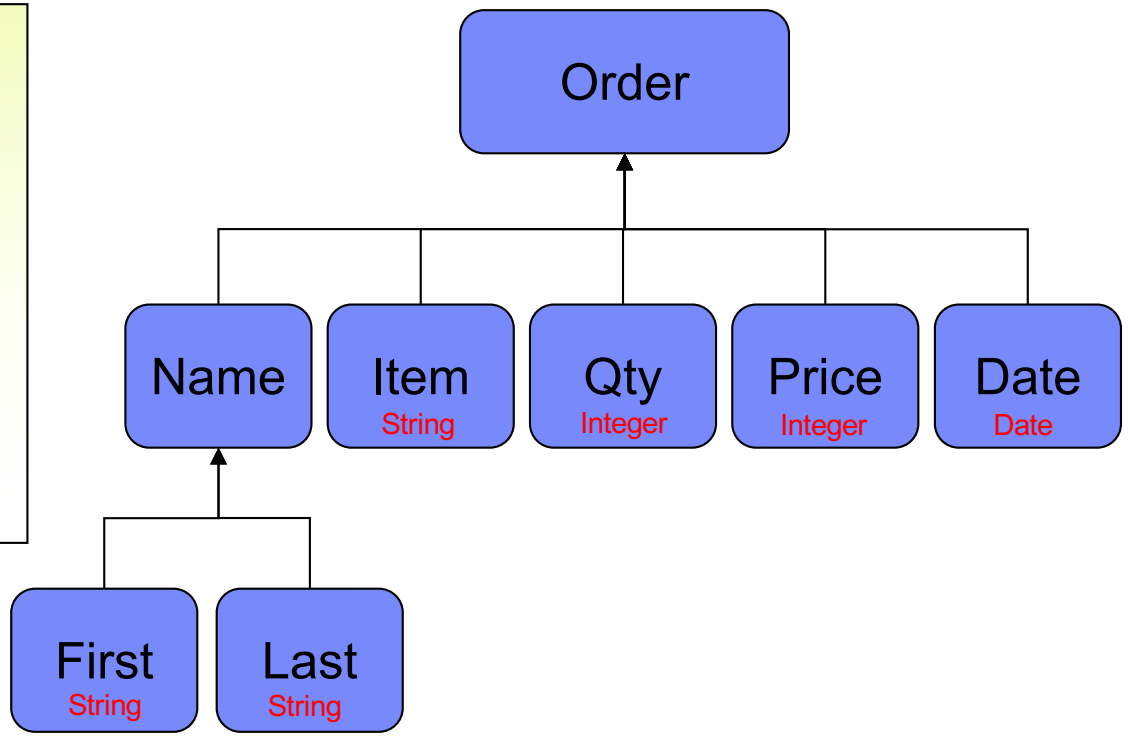


Logical message model

Easily model any type of data!

Physical ⟷ Logical

```
<order>
  <name>
    <first>John</first>
    <last>Smith</last>
  </name>
  <item>Graphics Card</item>
  <quantity>32</quantity>
  <price>200</price>
  <date>07/11/08</date>
</order>
```




```
John,Smith,Graphics Card,
32,200,07/11/08
```

```
John Smith.............
Graphics Card.........
3220020071108.........
```

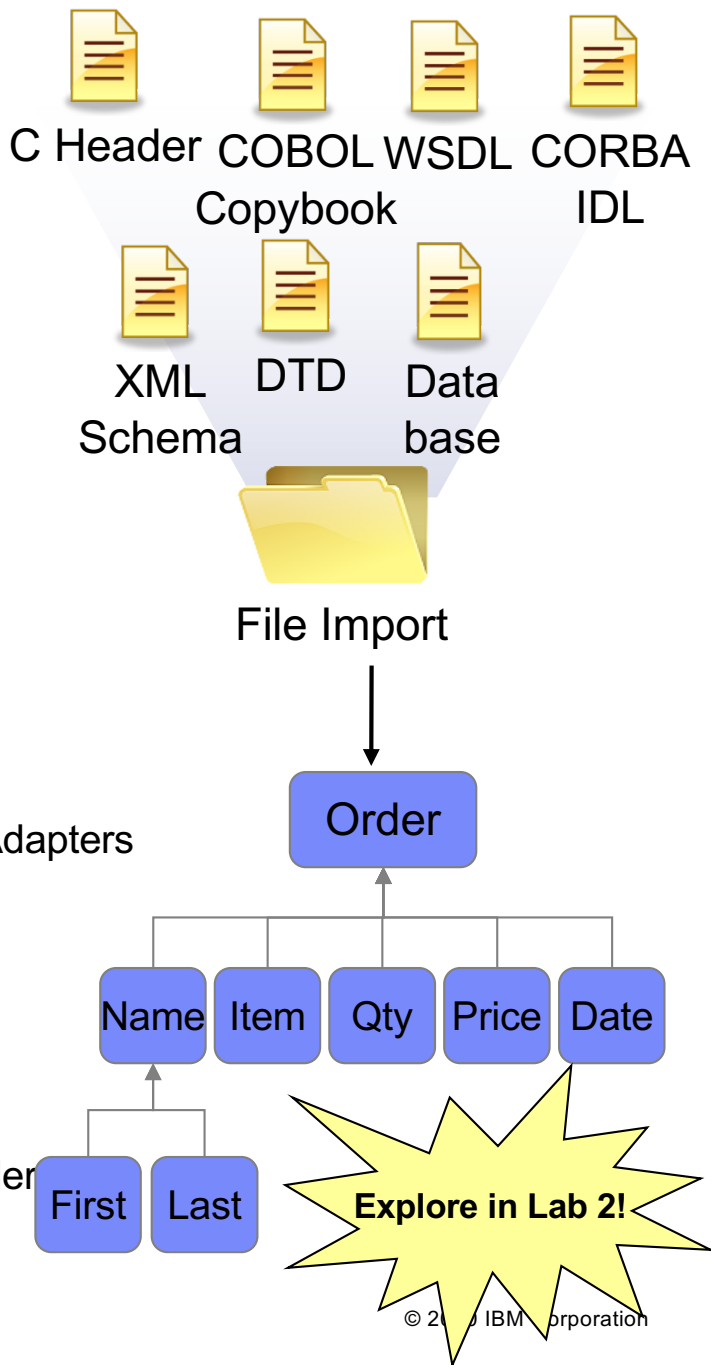
Navigate the Message Tree using XPATH:

$Body/Order/Name/Last or $Root/XMLNSC/Order/Name/Last

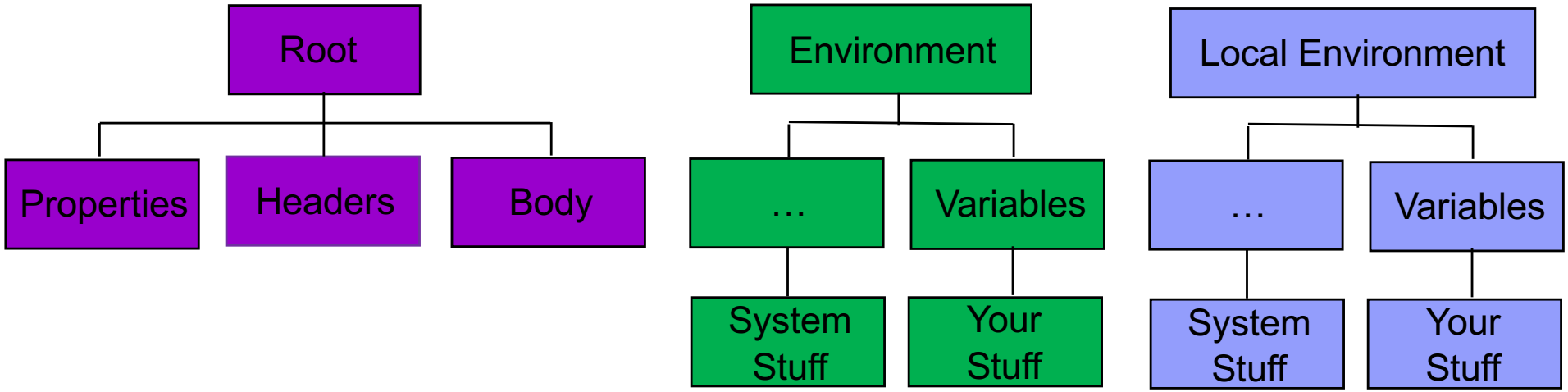
Parsers do the work

Make message handling easy!

- Transform raw data into logical view and vice versa
- Data models
  - Created on-the-fly if data is self-defining (XML, JSON)
  - Defined using importers or definitions in ACE Toolkit
- Payload parsers:
  - Created based on input/response message parsing tabs
  - Or can be overridden by header information such as MQRFH2
  - Can also be created using Java, ESQL, etc language calls
    - **BLOB** - No structure, just a sequence of bits
    - **XML** – XMLNSC, XMLNS
    - **Standards** - SOAP, MIME, JSON
    - **Model** – DFDL, MRM
    - **DataObject** - For interfacing to EIS systems via WebSphere Adapters
    - **JMS** – JMSMap, JMSStream
    - **Other** - WTX, User plugin parsers
- Transport header parsers:
  - Created by the transport nodes
    - **WMQ** - MQMD, MQRFH2, MQWIH, MQCICS, MQRMH ….
    - **HTTP** - HttpInputHeader, HttpRequestHeader, HttpReplyHeader
    - **JMS** - JMSTransport …



The Message Assembly – What is it?

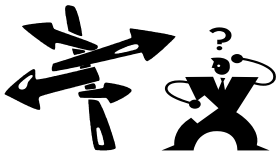


The Message Assembly is the internal (ACE) representation of a message. It consists of four "trees"

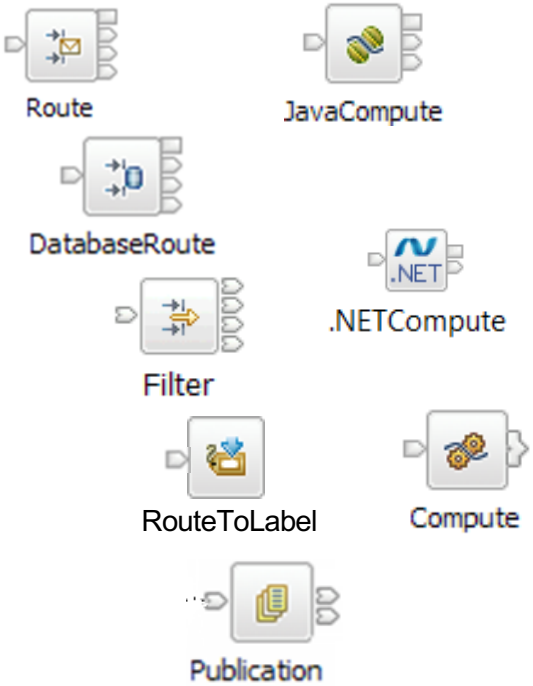
- The ***Message Tree***, which includes all the headers that are present in the message, in addition to the message body
  - The *Message Tree* is always present. The root of the *Message Tree* is called *Root*
  - It is passed from node to node in a single instance of a message flow
- The ***Environment Tree***
  - The *Environment* tree can be used to store information while the message passes through the message flow.
  - The *Environment* tree differs from the *LocalEnvironment* tree in that a single instance of it is maintained throughout the message flow.
- The ***LocalEnvironment Tree***
  - The *LocalEnvironment* tree is a structure in which you can also store information while the processes the message
  - The *Localenvironment* tree differs from the *Environment* tree in that an *Environment* variable's scope is defined for the whole flow, while a *LocalEnvironment* variable's scope is defined at the node level (although it can optionally be propagated to the next node)
- The ***ExceptionList Tree***
  - The *ExceptionList* tree is a part of the Message Assembly in which the message flow writes information about exceptions that occur when a message is processed.

Routing and Orchestration

Routing – Delivering data to the right places

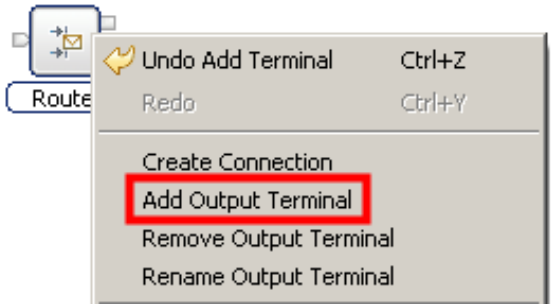


- Core integration function
  - Can be simple to complex
  - Can be based on content or rules
  - Decisions can be static or dynamic
- Many ways to do routing:
  - Programmatically
    - JavaCompute node (Java)
    - .NETCompute (Any CLR-compliant language)
    - Compute node (eSQL)
  - Configuration-based
    - Route-To-Label
    - Filter node
    - DatabaseRoute node
    - Publication node
    - **Route** node

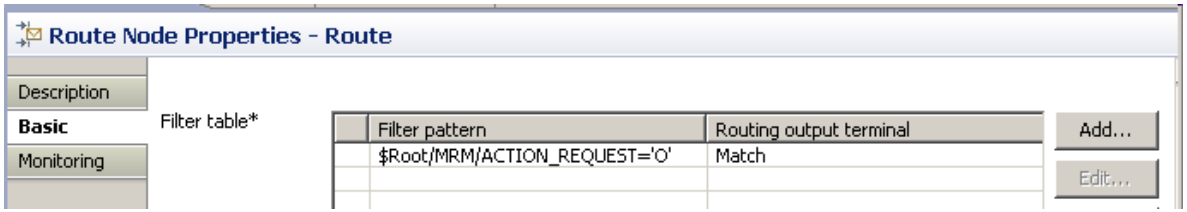


Routing is easy using the Route node!

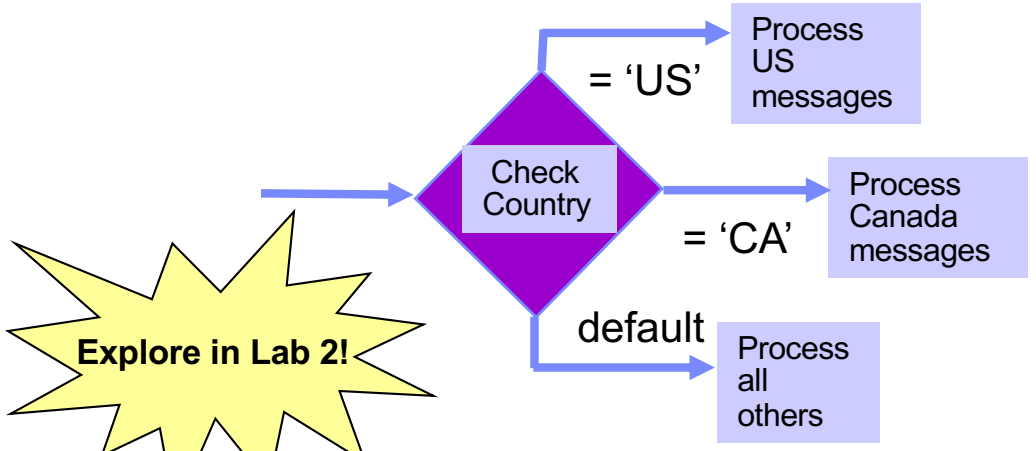
- Configuration-driven
  - No programming needed
- As many output paths as needed
  - Three default output terminals (Match, Default, Failure)
  - Add additional output terminals as needed



- Filter table controls routing to output terminals
  - Made up of XPath expressions
  - Default terminal if none true
  - Match on *First* or *All*



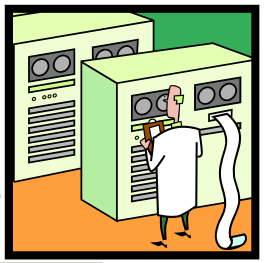
- Lab 2 will demonstrate use
  - Configure a ***Route*** node
  - Define two additional output terminals
  - Label them ***US*** and ***Canada***
  - ***default*** path for no-match situation
  - Use the XPath Expression Builder to define the rules



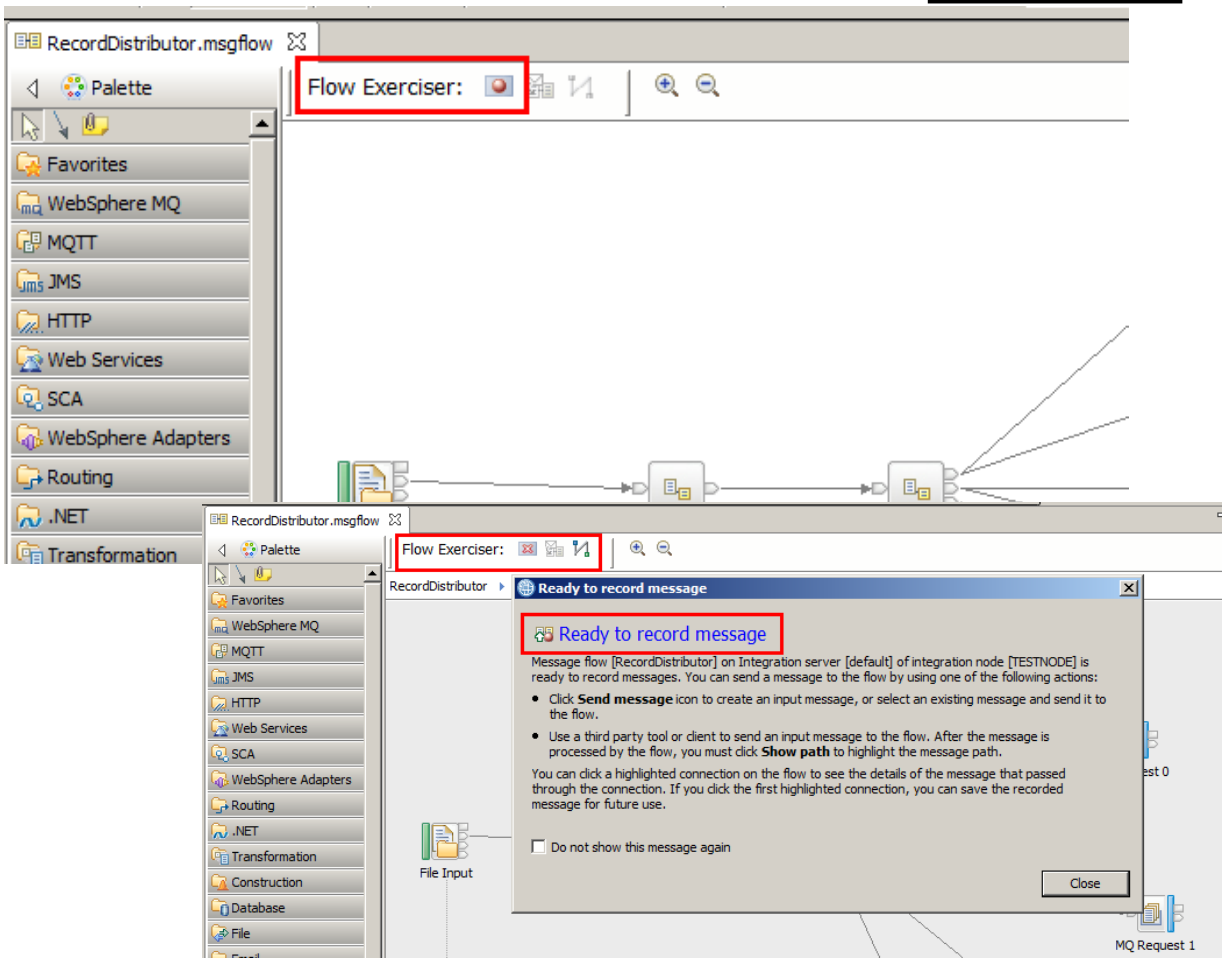
Tools

Message Flow Exerciser

Makes message flows easy to test!



- **The App Connect Enterprise Toolkit includes a Message Flow Exerciser**
- **Can be used to test message flows containing the following types of input**
  - WebSphere MQ, JMS, SOAP, HTTP and SCA
- **Monitors the output nodes in a message flow**
  - Provides information about the path that a test message takes through a message flow
  - This includes information about errors that are generated by the message flow
- **Test scripts can be saved**
  - Enables repeatable test suites to be created

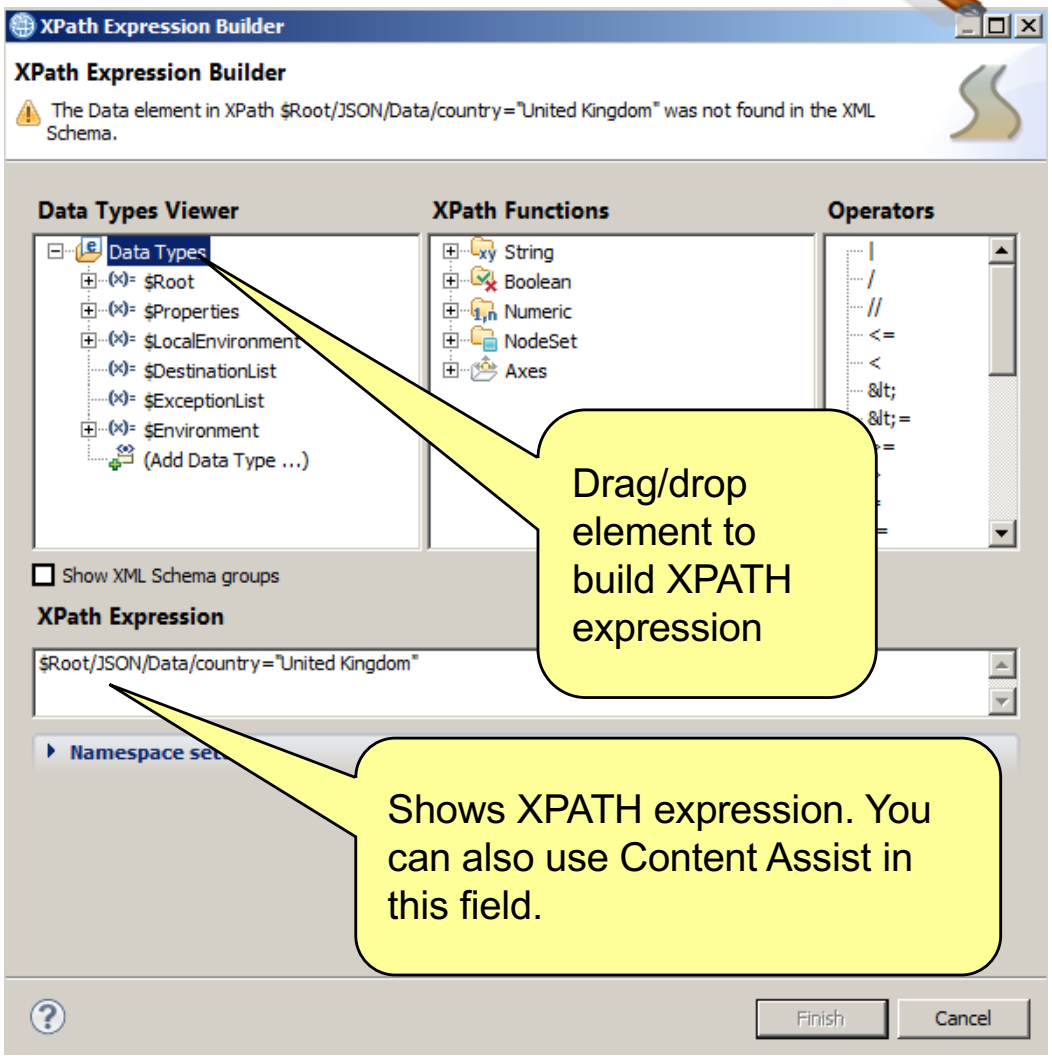


Explore in Lab 1!

XPath Expression Builder

Makes message formats easy to navigate!

- **The App Connect Enterprise Toolkit includes an XPath Expression Bu**
- Easy-to-use tool
  - Build XPath visually
  - Or enter directly
- ***Data Types Viewer***
  - Use message tree
- ***XPath Functions*** palette
  - Supplied XPath functions
- ***Operators*** palette
  - XPath operators
- ***XPath Expression*** field
  - Displays expression as it is generated
  - Content assist available

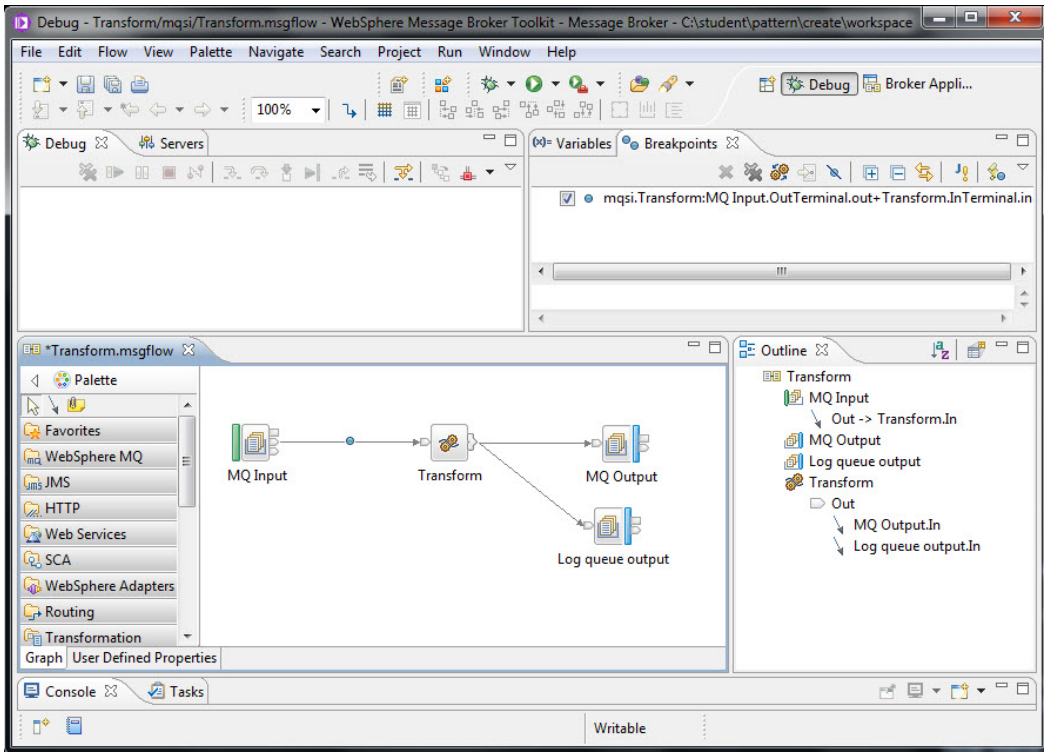


Explore in Lab 2!

Integrated Flow Debugger

Makes finding problems easy!

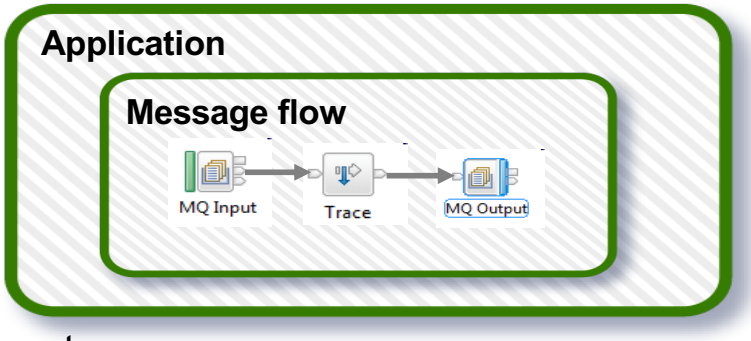
- **The App Connect Enterprise Toolkit includes an integrated Flow Debugger**
  - Convenient, easy to use graphical interface for flow testing
  - Operates in a ***Debug*** perspective
  - Can be initiated from Test Client or Debug Perspective
- Breakpoints between nodes or within node logic
  - ***Stop*** execution at any point
    - Current content of message trees examined and/or modified
  - ***Step Over*** executes next node then pauses
  - ***Resume Execution*** (breakpoints intact)
  - ***Run To Completion*** (breakpoints disabled)
- Step into node and debug at the source level
  - Subflows
  - Compute (eSQL)
  - JavaCompute (Java)
  - .NETCompute



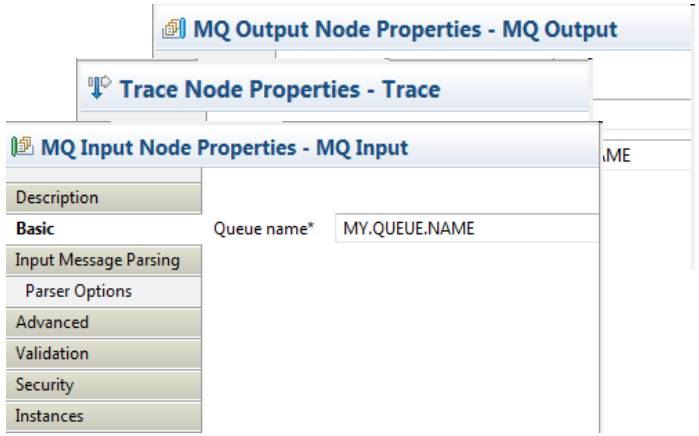
Explore in Lab 7!

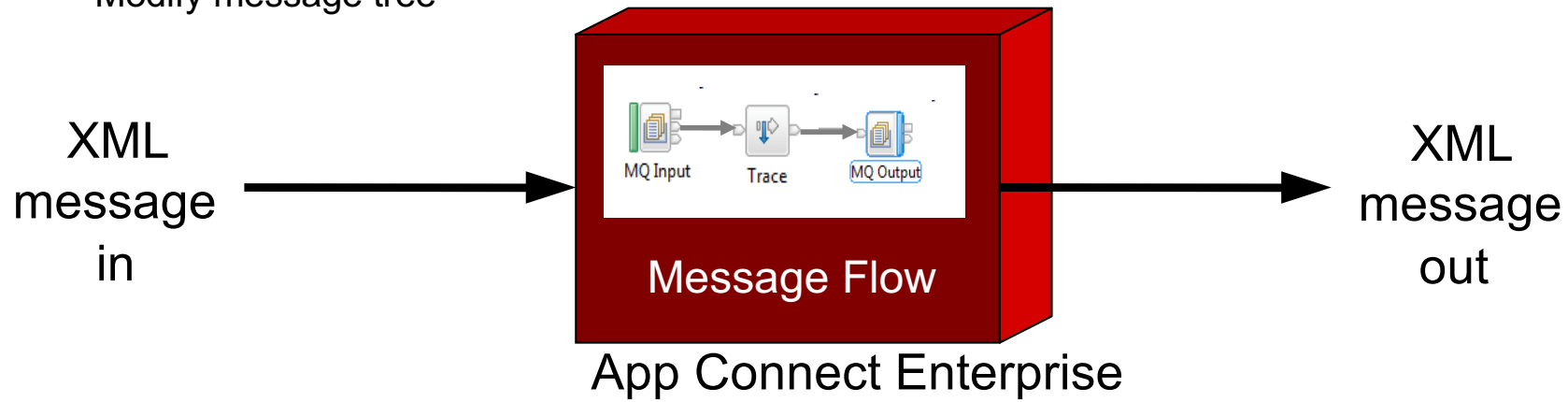
Questions

Time to explore



- ***Lab 1*** - Create a new **application**
  - Construct a simple **Message Flow** in the application
    - **HTTPInput** node gets messages from a queue
    - **HTTPOutput** node puts messages to a queue
    - **Trace** node renders message structure into a readable format
  - Test your application
    - Use the **Message Flow Exerciser** to deploy and test
      - XML Message In/Out, No Transformation
- ***Lab 2*** - Extend your flow to do message parsing and validation
  - Modify your message flow to perform content-based routing
  - Use the interactive debugger to test your flow
    - Set breakpoints within the flow
    - Examine message tree
    - Modify message tree





It's time For Labs 1-2!

- **Introductory labs (approx. 45 – 60 mins)**
  - **Building and executing a simple message flow**
  - **Extending the message flow for parsing and validation**
  - **Extending the message flow for content-based routing**
- **Tools uou will use:**
  - **Message Flow Exerciser**
  - **XPath Expression Builder**
  - **Integrated Flow Debugger**

Advanced integration features and functions

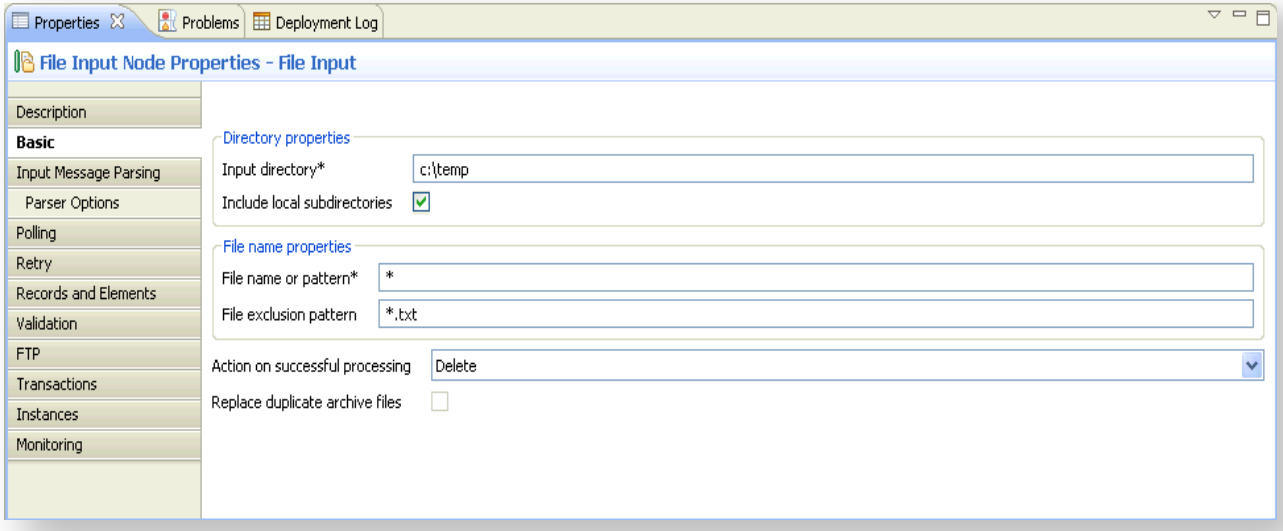
File support overview

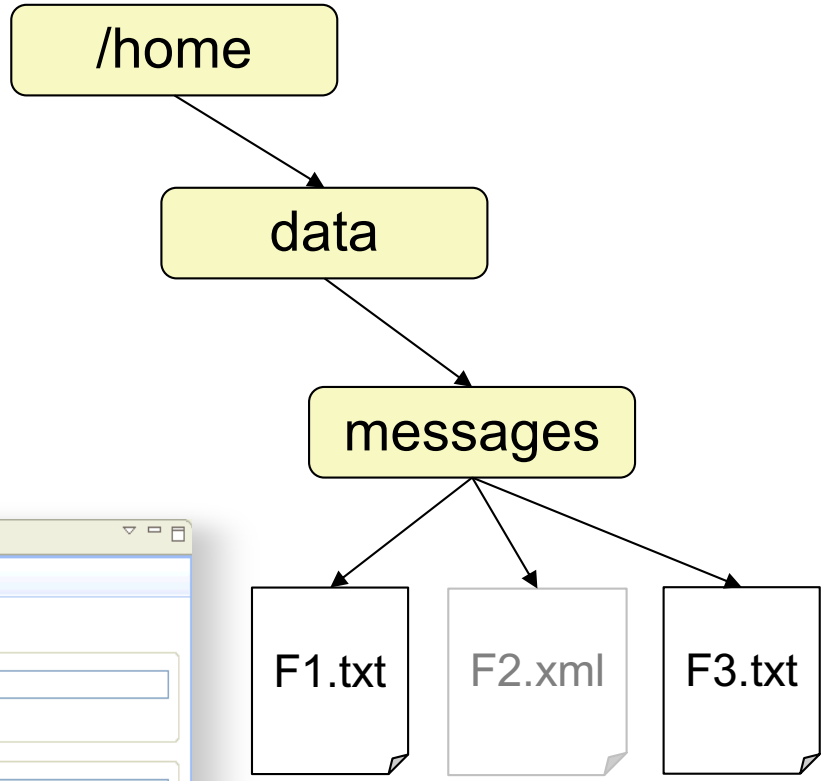
Making file processing simple

- **Native file processing support**
  - Local files, as well as FTP/SFTP/FTPS
- **Node-based implementation easy to use**
  - FileInput, FileOutput and FileRead nodes
  - Combine with other ACE nodes
    - (e.g.) File to MQ, File to database, File record filtering
- **Provides efficient handling of large files**
  - Allows very large (gigabyte) files to be processed without using excessive memory
  - Appropriate ACE parsers have been enhanced to request data “on demand”
- **Comprehensive support for record detection**
  - Simple: LF, EOL, CRLF, fixed-length, whole-file, user-defined
  - Parser: Use an existing message definition to identify record boundaries
- **Flexible operational model**
  - Archive and back-out directory, checkpoint/restart recovery model
  - Hot deployment, no restart necessary
  - Configurable services allow operational change management
- **File processing examples**
  - Split batch of records to set of messages
  - Combine messages to produce batch of records in file
  - Transforming data formats
  - Routing based on file content
  - Connecting and sending to different protocols (MQ, JMS, HTTP, TCPIP, SAP, Siebel, Web Services, IMS, CICS, DB…

FileInput node – Basic algorithm

- **Pattern-driven approach**
  - Scans directory for files that match pattern
  - File exclusion pattern to specify files to ignore
- **Include local subdirectories option:**
  - Scan specified directory and all subdirectories, recursively
    - Skips directories that cannot be accessed
    - Internal directories are made at each level
    - Symbolic links are supported
    - Infinite recursion detected
- **Locked files ignored**



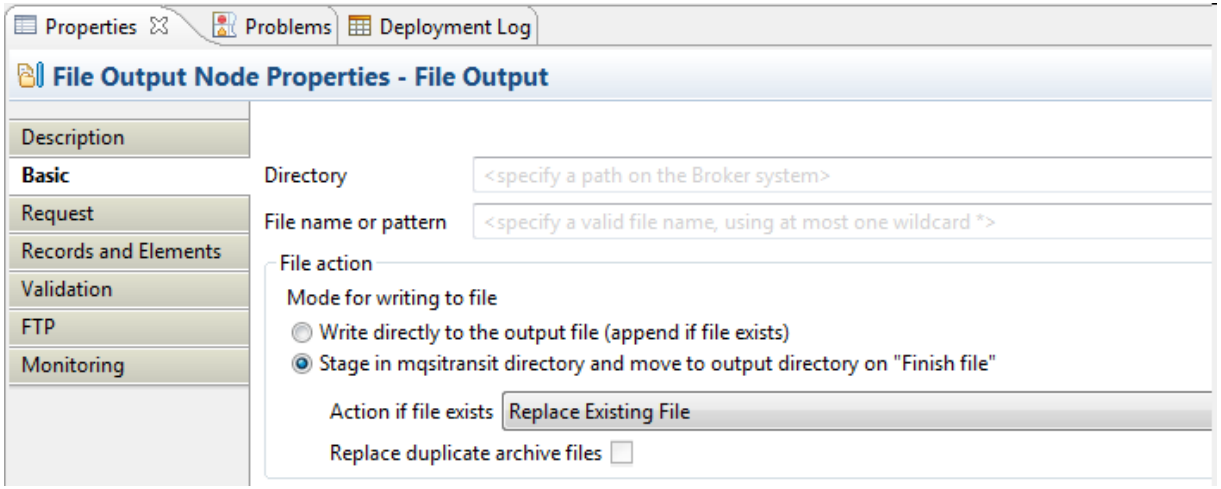


For example:

Input directory:
/home/data/messages
File name or pattern: *.txt

FileOutput node – Basic algorithm

- Message body written to specified file
- Destination directory and file name can be set dynamically in flow
- Options if output file already exists include:
  - Replace existing file, fail, archive previous file, append record
- FTP option as well
  - Transfer is synchronous
  - Use additional instances if throughput rate is issue



Integrate flows with managed file transfer products

Consistent with FileInput & FileOutput nodes but exploit MFT product features


FTE Input


FTE Output

- **WebSphere MQ Managed File Transfer (MFT)**
  - File transfer over MQ backbone – move data reliably!
  - Basic FTP; many MFT products do not truly move data reliably
  - Enable end-to-end transactional file processing
    - Receive / send FTE transfers
    - Consistent with FileInput/FileOutput nodes but make full use of the power of MQ MFT
      - MFT Metadata provided in `LocalEnvironment`, allowing intelligent processing of transfers
    - On output, `LocalEnvironment` allows transfer overrides and customizable metadata
    - Typical scenarios are reliable file-to-queue, database-to-file, file-to-file
    - Agents automatically installed, configured and managed

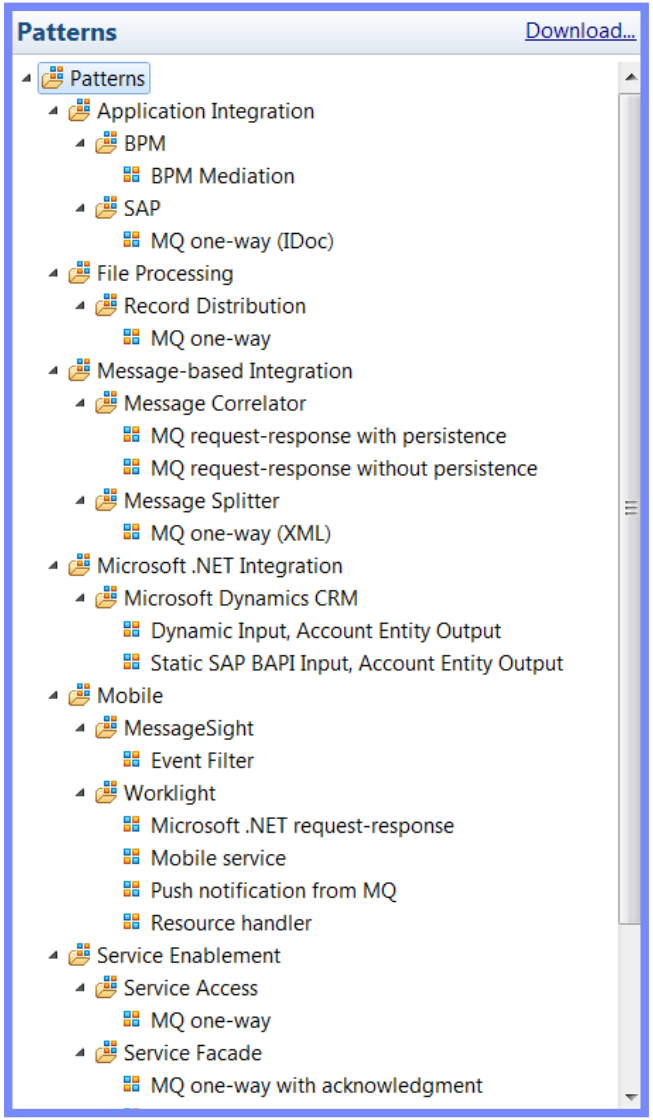
CD Input

CD Output

- **IBM Sterling Connect:Direct**
  - Connect:Direct nodes extend file processing capabilities
    - Receive / send Connect:Direct transfers - Uses standard CD client API
    - Consistent with FileInput/FileOutput nodes but make full use of the power of Connect:Direct
    - Simplifies using Connect:Direct (no need to understand process scripts)
    - Input node monitors Connect:Direct servers stats for completed transfers
    - Processes files immediately - Connect:Direct Metadata available to flow
    - Can leave file unchanged after processing (just delete notification message)
    - Output node identifies destination Connect:Direct server, directory, and more
      - Can be overridden using local environment - Including any Connect:Direct options
    - Wild card file names, sequential and partitioned datasets supported
    - Connect:Direct Metadata in `LocalEnvironment`

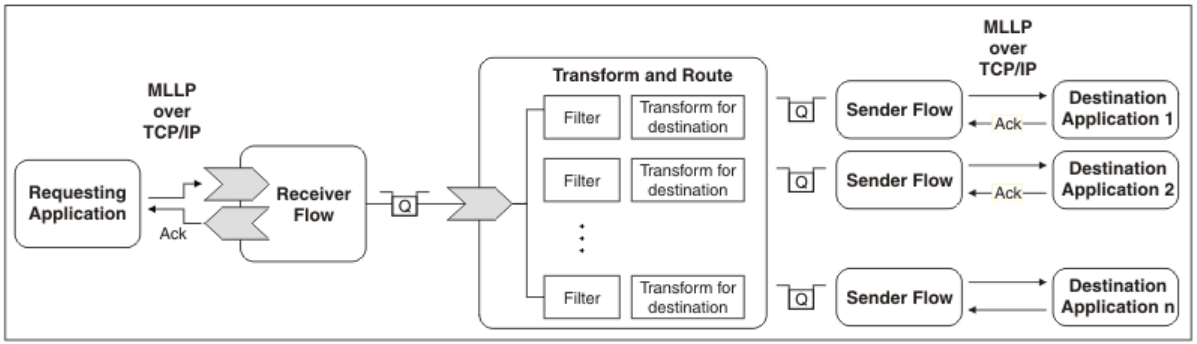
Patterns for simplified development

- Creates top-down, parameterized connectivity solutions
- Reduces common problems in flow development
- Establishes best practices for integration solutions
- Reduces time-to-value for solution development
- Patterns are easily extended with regular ACE functionality



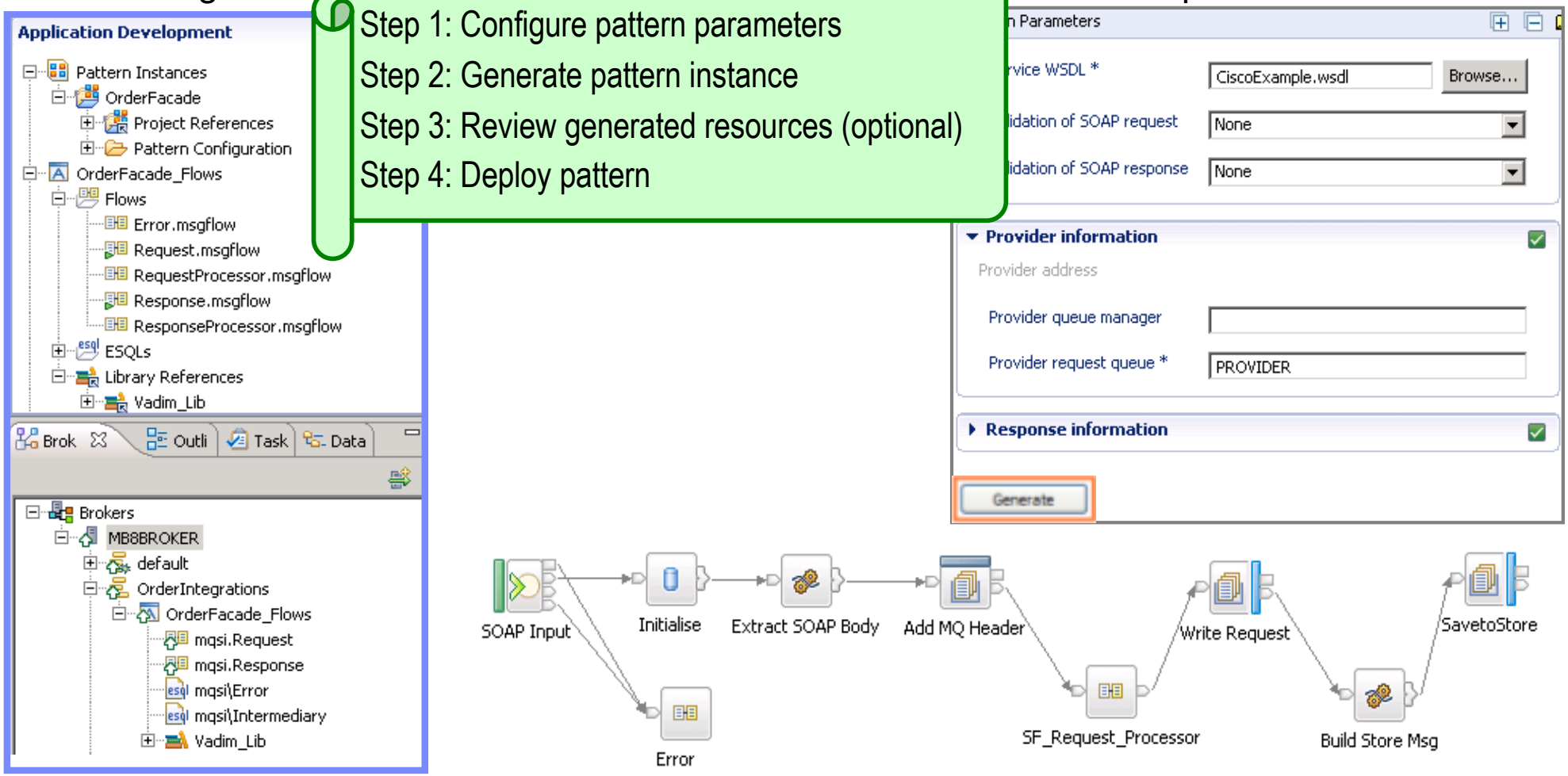
Healthcare: HL7 to HL7 pattern

The Healthcare: HL7 to HL7 pattern integrates an application that can send Health Level Seven International (HL7) v2 messages with one or more applications that can receive HL7 messages. The applications must be capable of sending and receiving HL7 messages by using Minimal Lower Layer Protocol (MLLP) over TCP/IP.



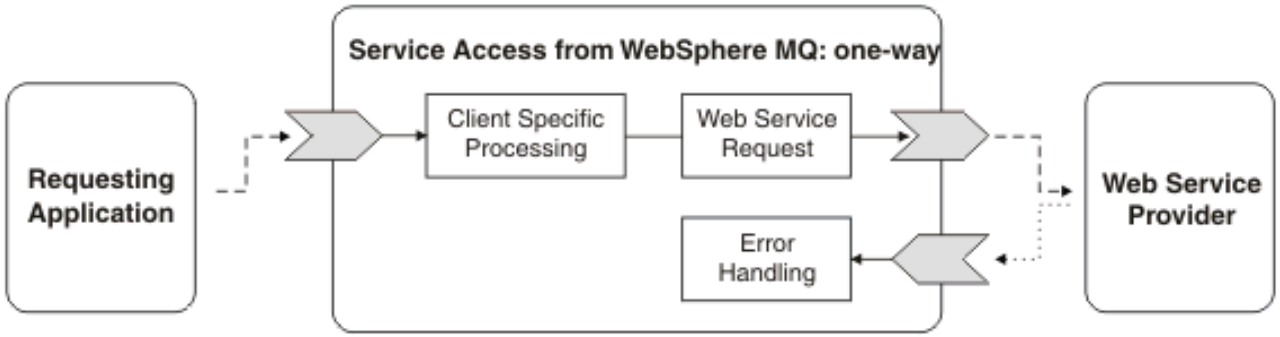
Pattern generation

- Pattern generation creates the production-ready App Connect Enterprise projects
  - Generated projects reflect the configuration choices of the pattern user
  - Configuration is saved so that the pattern can be re-generated if required

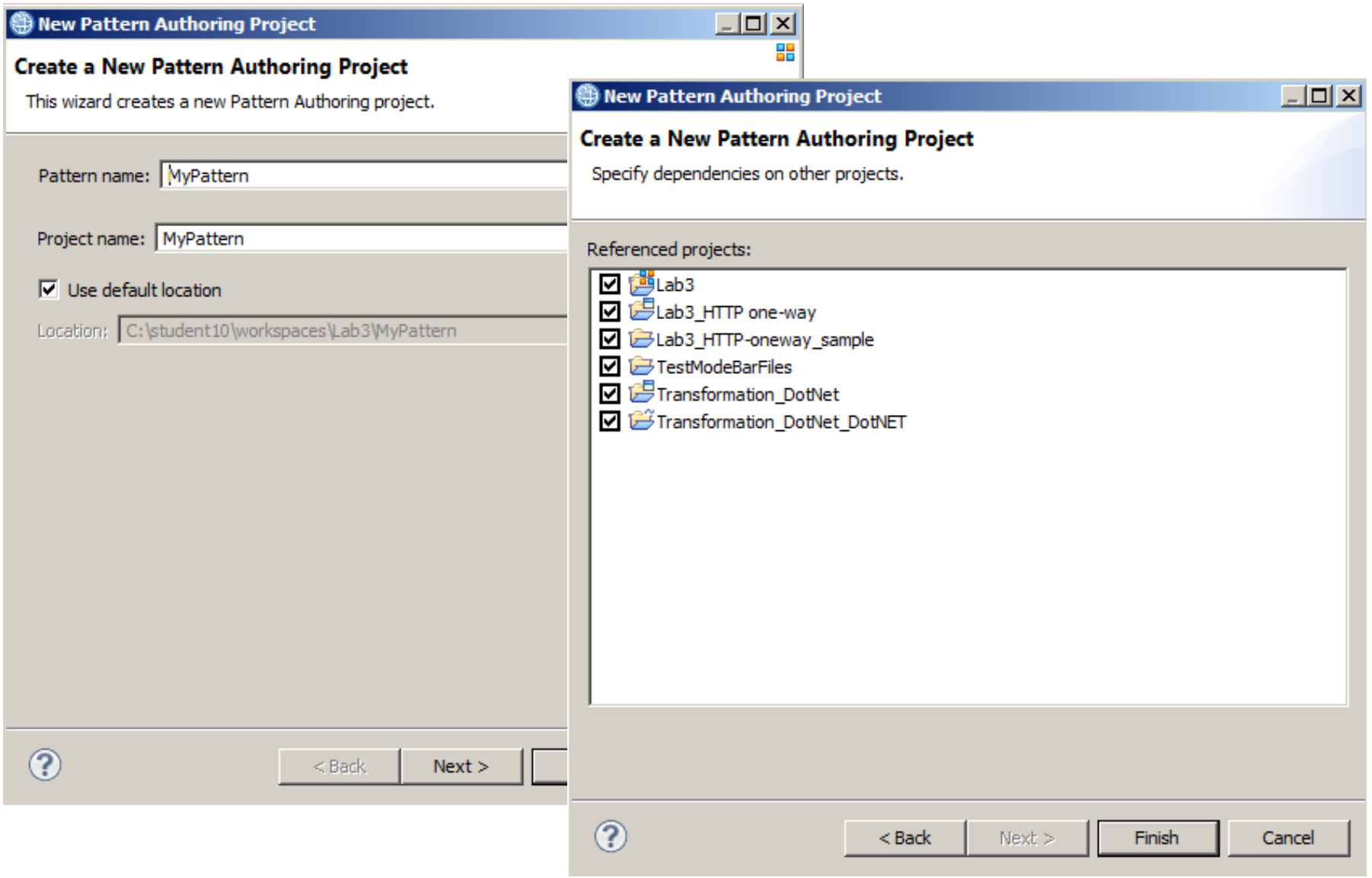


Built-in patterns

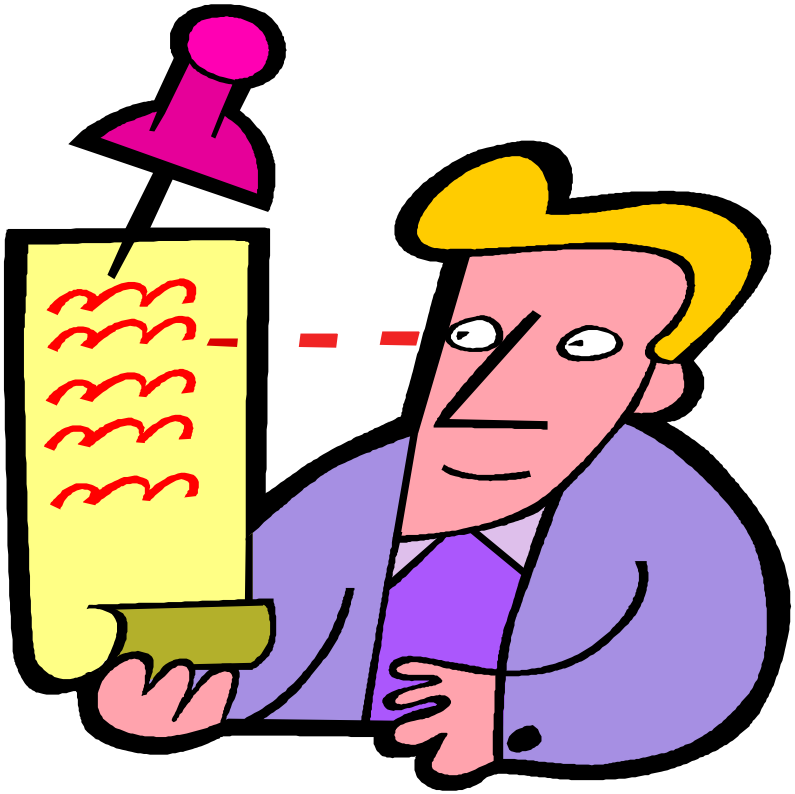
- App Connect Enterprise provides a core set of built-in patterns
- These implement a variety of common scenarios:
  - Web service front end to a MQ based application
  - Processing data stored in a file and routing to one or more queues
  - Adding a proxy in front of a web service provider
  - Processing data from an SAP system and routing to MQ
  - Shredding messages and routing to one or more queues
- Patterns are selected based on client feedback and field experience
- This core set of patterns continues to grow with each release



Have a better idea? Create your own patterns!



Message Modelling with DFDL

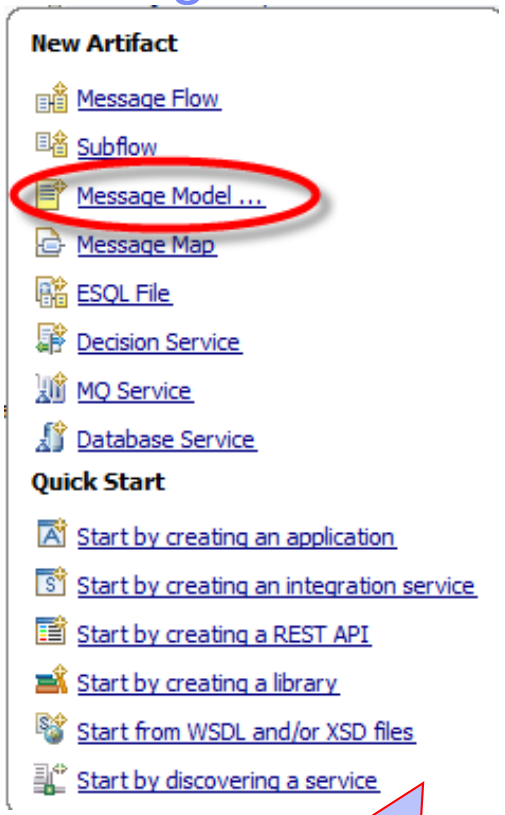


Data format description language (DFDL)

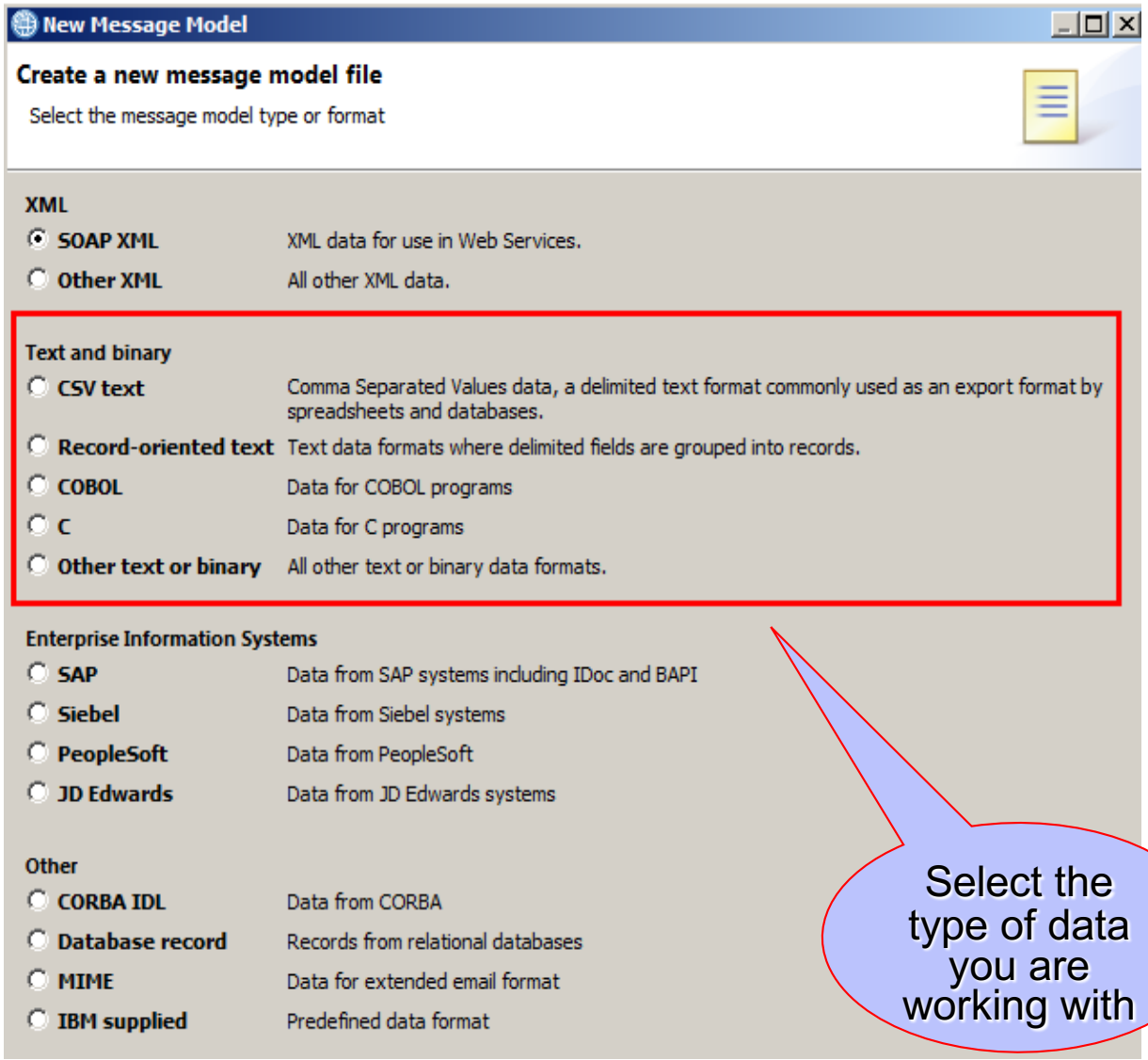
- A new open standard
  - Open Grid Forum (OGF)
  - http://www.ogf.org/
  - Version 1.0
  - ‘Proposed Recommendation’ status
- A way of *describing* data…
  - NOT a data format itself!
- Describe any data format …
  - Textual and binary
  - Commercial record-oriented
  - Scientific and numeric
  - Modern and legacy
  - Industry standards
- High performance …
  - Right data format for the job

- Leverage XML technology and concepts
  - W3C XML Schema subset & type system
  - Annotations within the XSD
    - Physical representation of data
  - XPath to reference fields within data
- Round-tripping
  - Read / write data in described format
    - From same description
- Keep simple cases simple
  - Simple descriptions human readable
- Generality
  - Think “Type Tree + MRM” and more

Creating a DFDL model



Launcher for creating App Connect Enterprise project artifacts



Select the type of data you are working with

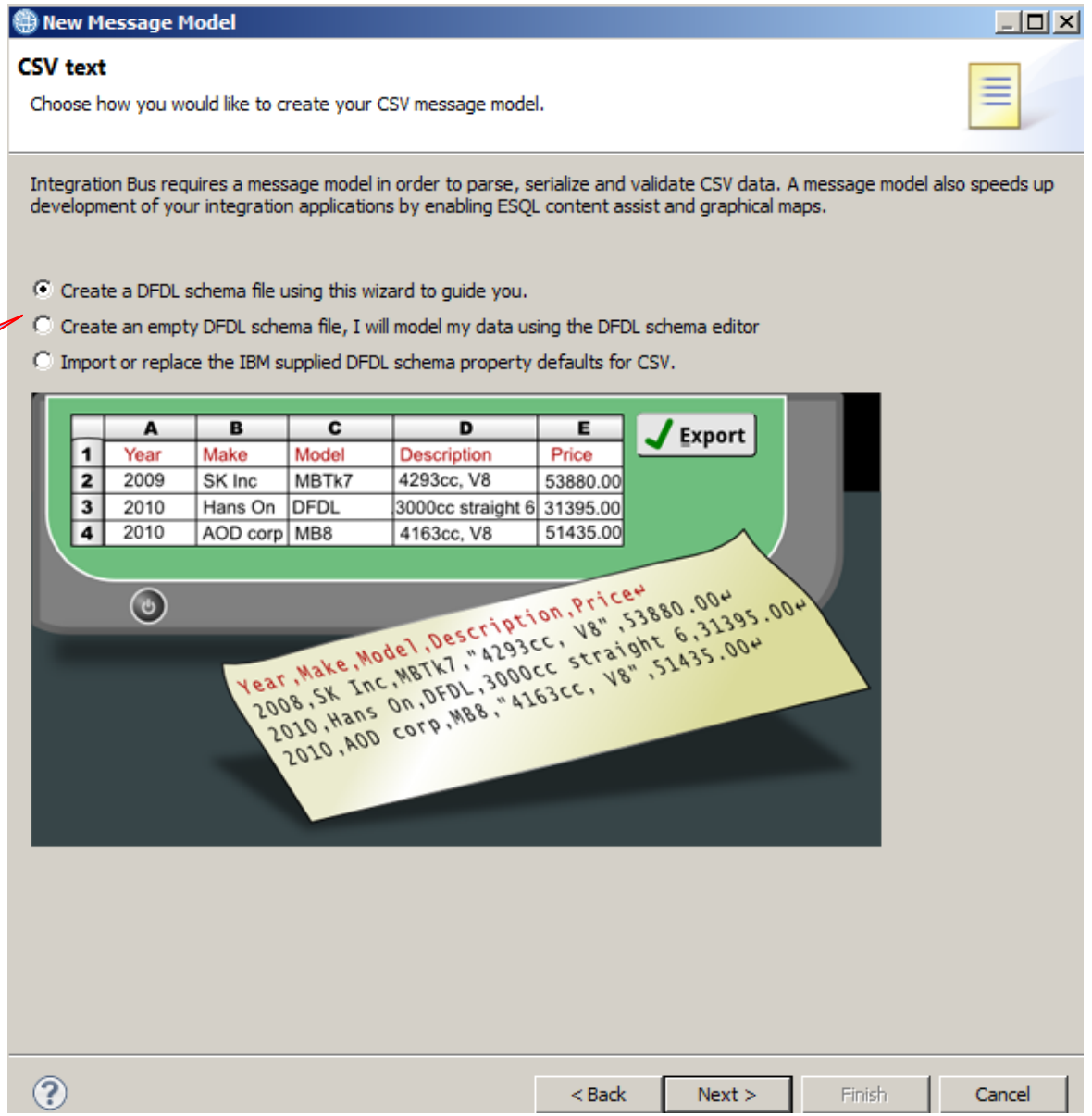
- Or drop existing DFDL schema into App Connect Enterprise library

Wizard options for creating a DFDL model

- Guided authoring
- Using the DFDL editor
- Importing from other metadata
- Already have DFDL schema

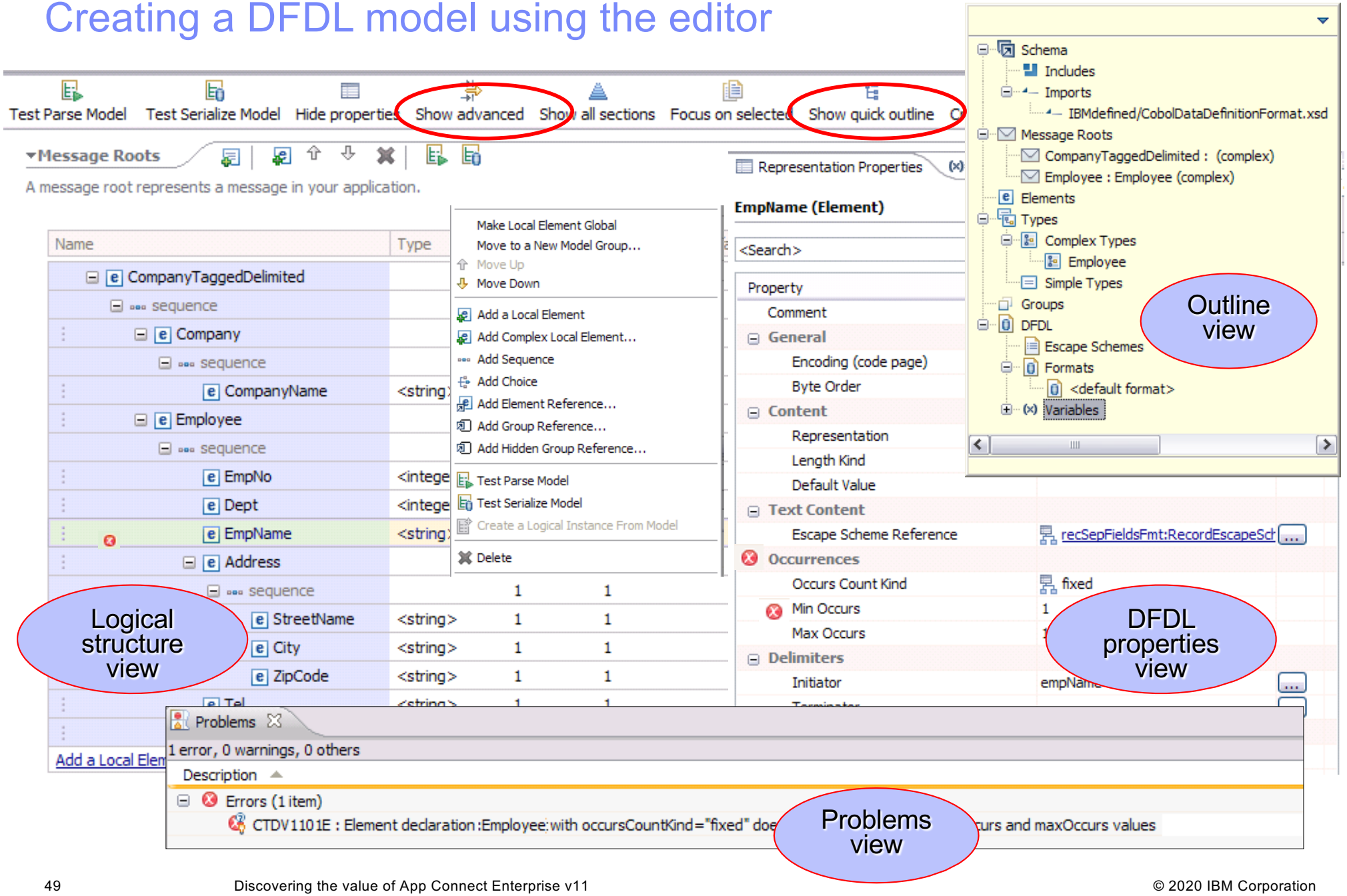
Select how you want to create the DFDL model

Explore in Lab 8!

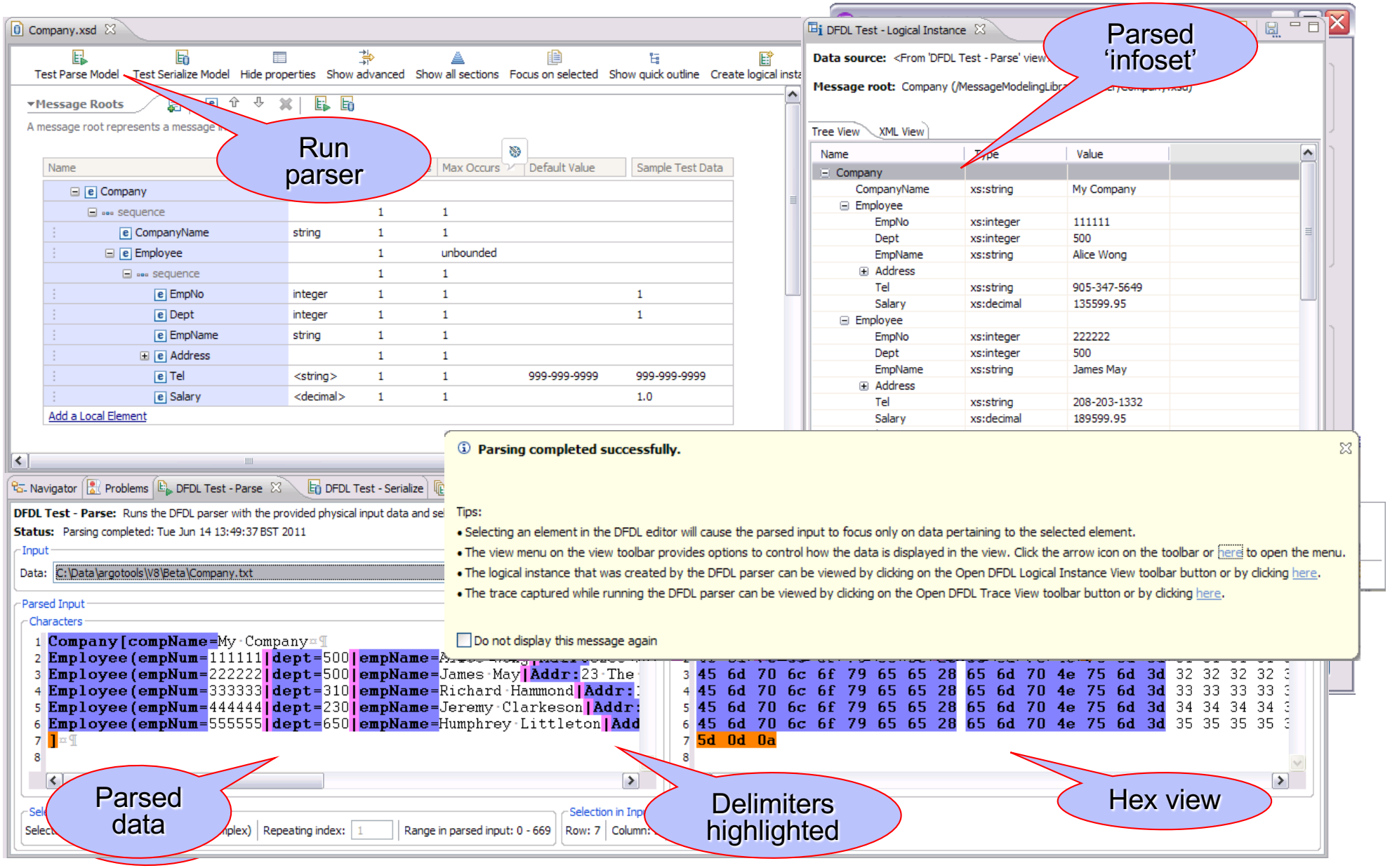


	A	B	C	D	E
1	Year	Make	Model	Description	Price
2	2009	SK Inc	MBTk7	4293cc, V8	53880.00
3	2010	Hans On	DFDL	3000cc straight 6	31395.00
4	2010	AOD corp	MB8	4163cc, V8	51435.00

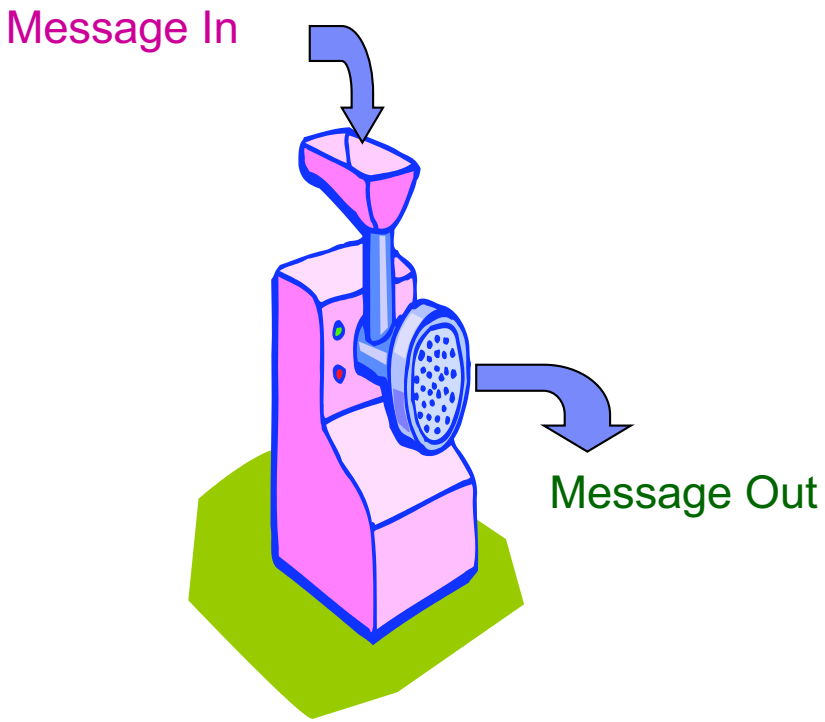
Creating a DFDL model using the editor



Testing a DFDL model within the editor



Transformation – graphical data mapping



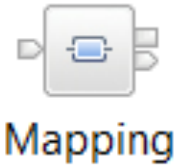
Powerful transformation options

- App Connect Enterprise has a number of transformation options:
  - Graphical Mapping
  - XSLT
  - ESQL
  - Java
  - .NET

- Reflects the importance of transformation in connectivity solutions
  - User-defined nodes supported for Java and C/C++
- Every transformation option has strengths and weaknesses
  - Performance and scalability
  - Backend integration
  - Skill sets and learning curve
  - Developer usability
  - Portability and maintenance
- Use a transformation technology appropriate to the problem at hand!

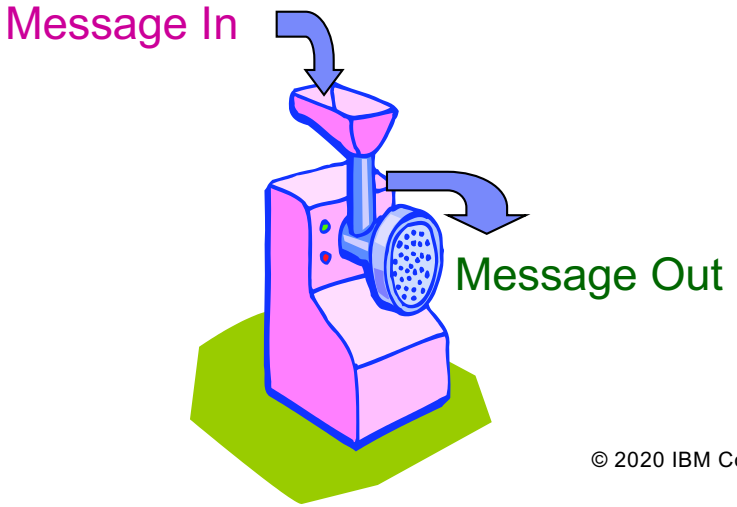
IBM graphical mapper

Makes transformation easy!

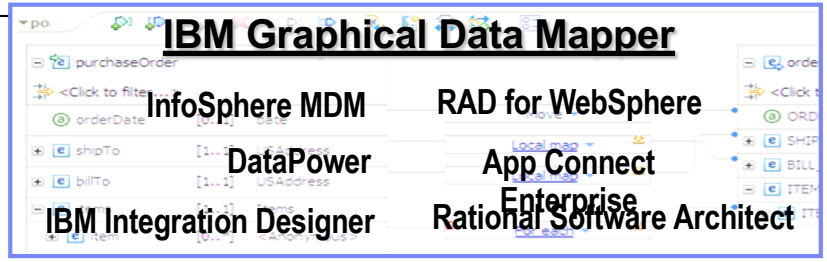


- Graphical mapping tool simplifies transformation, improves usability
  - Simple learning curve
  - Simple transformations are easy to create
  - Complex transformations build on concepts previously mastered
- Improved capability over older mapping technologies
  - Builds upon 'best of breed' mapping technologies in IBM products
  - Full XPath 2.0 scripting support
    - Standards-based language designed for hierarchical data structures
  - Databases can be a map source/target as well
- Rich set of features make GDM a good default transformation choice
- Excellent performance
  - Dedicated runtime engine

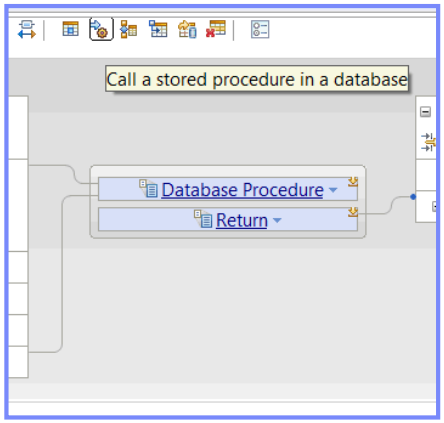
Explore in Lab 4!



Graphical transformations

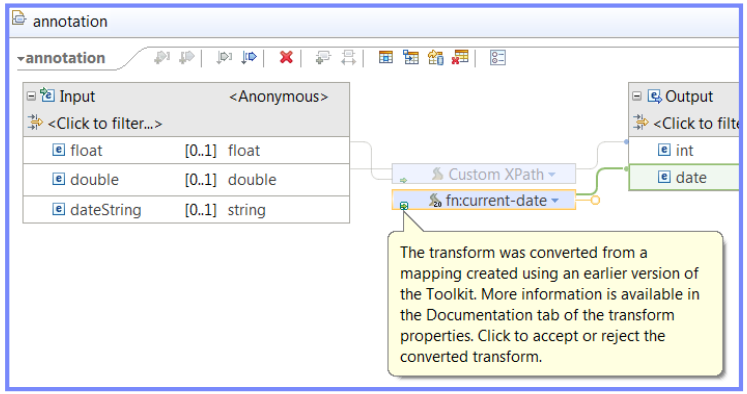


- IBM GDM designed for whole IBM product set, e.g.
  - App Connect Enterprise V9 – v11, WebSphere Message Broker v8, DataPower®
  - InfoSphere® Master Data Management v11, Integration Designer v7.5/v8
  - Rational Application Developer for WebSphere Software v8.5
  - Rational Software Architect v8.5, RSA for WebSphere Software v8.5
  - Other products yet to announce

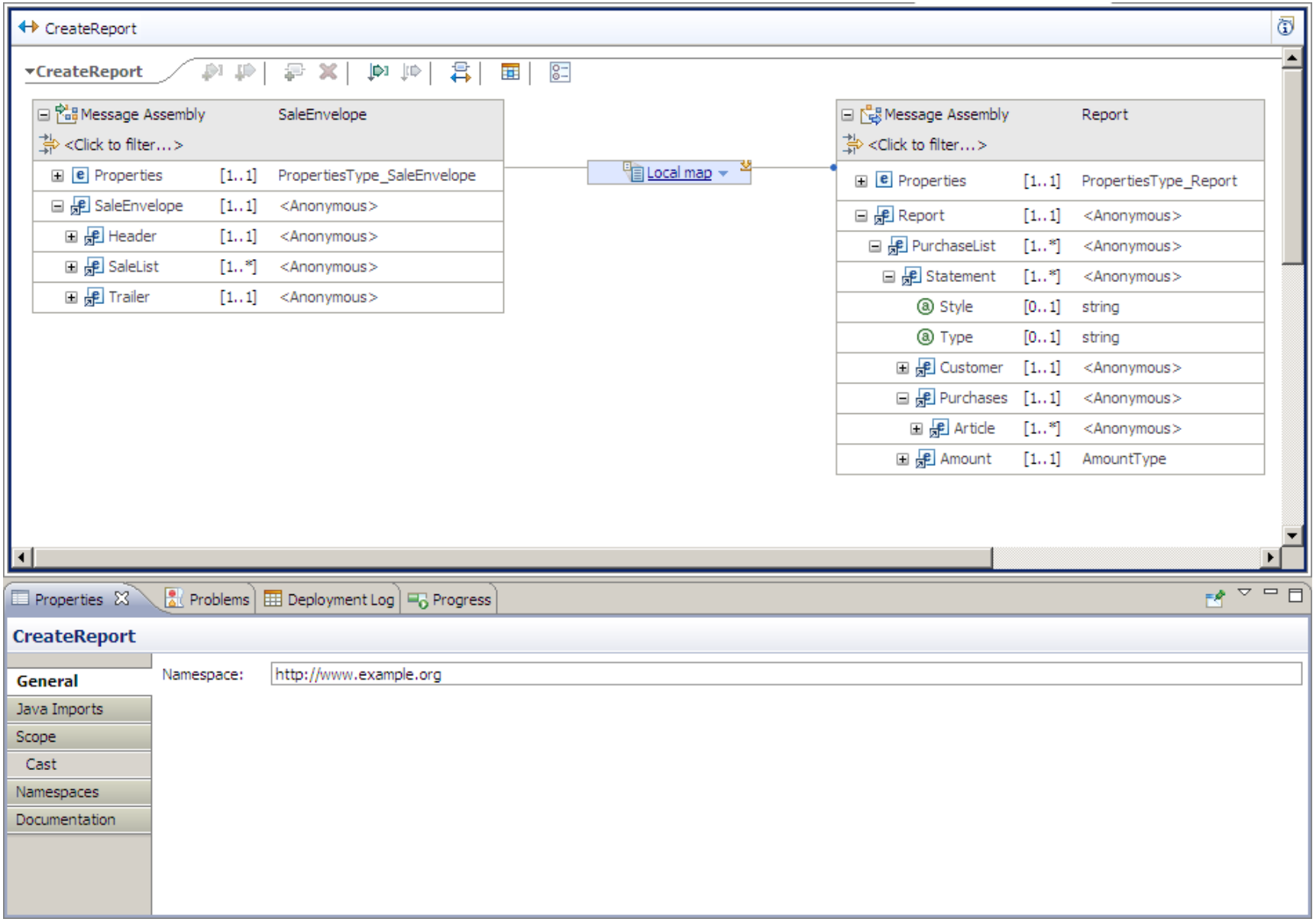


- Directly access stored procedures from within a map
  - Complements existing database select, insert, update, delete
  - Incorporate user-defined database functions into your graphical transforms
  - All standard broker databases supported, e.g. Oracle, DB2, SQLServer…

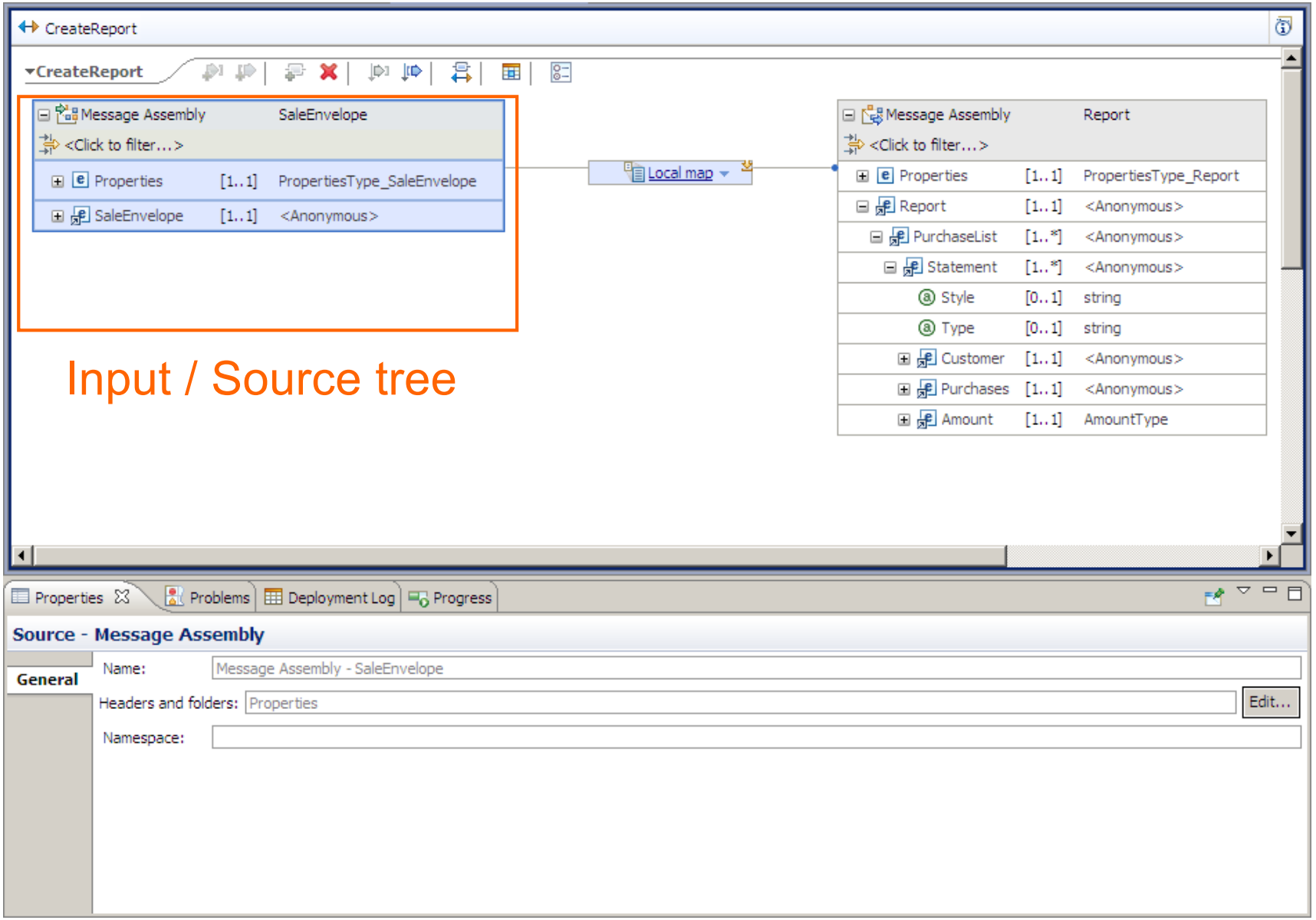
- Maps available to user patterns
  - Graphical creation of flows which require transformation logic
    - For example: new input or output messages
  - Invocation of mapper when pattern instances are generated
  - User guidance through HTML pattern help and task list
  - Patterns to demonstrate include CRM account mapping



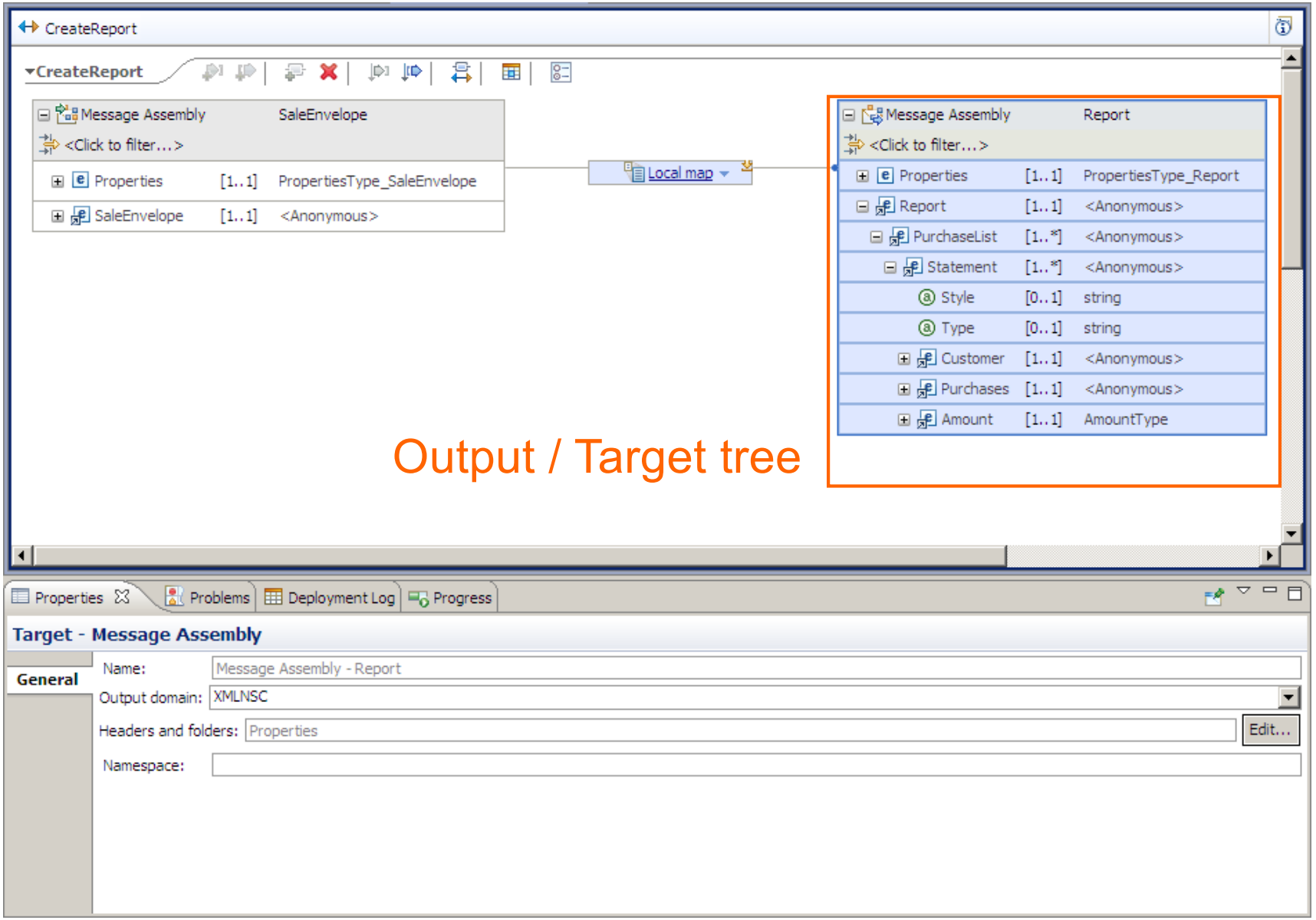
The mapping editor



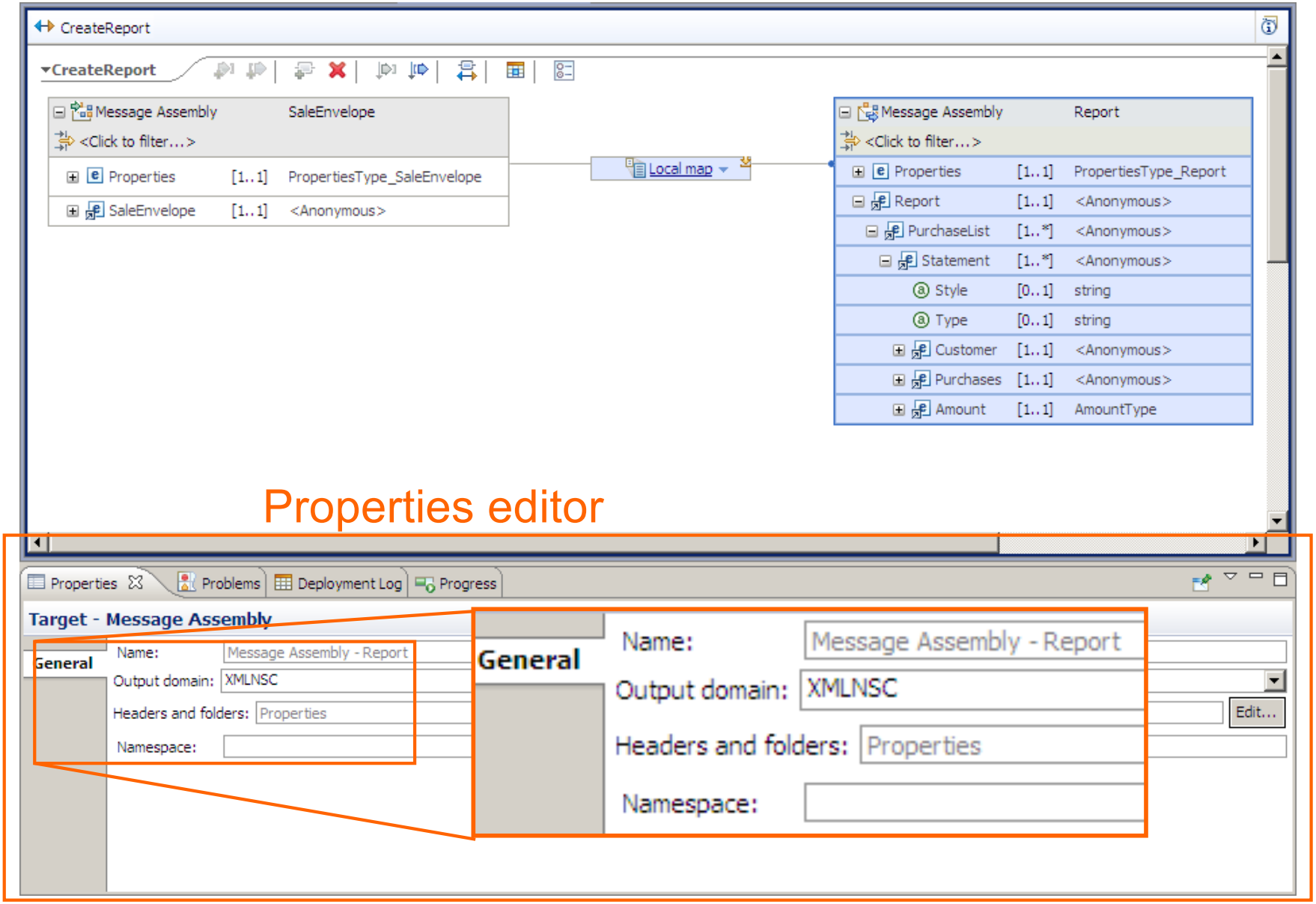
The mapping editor



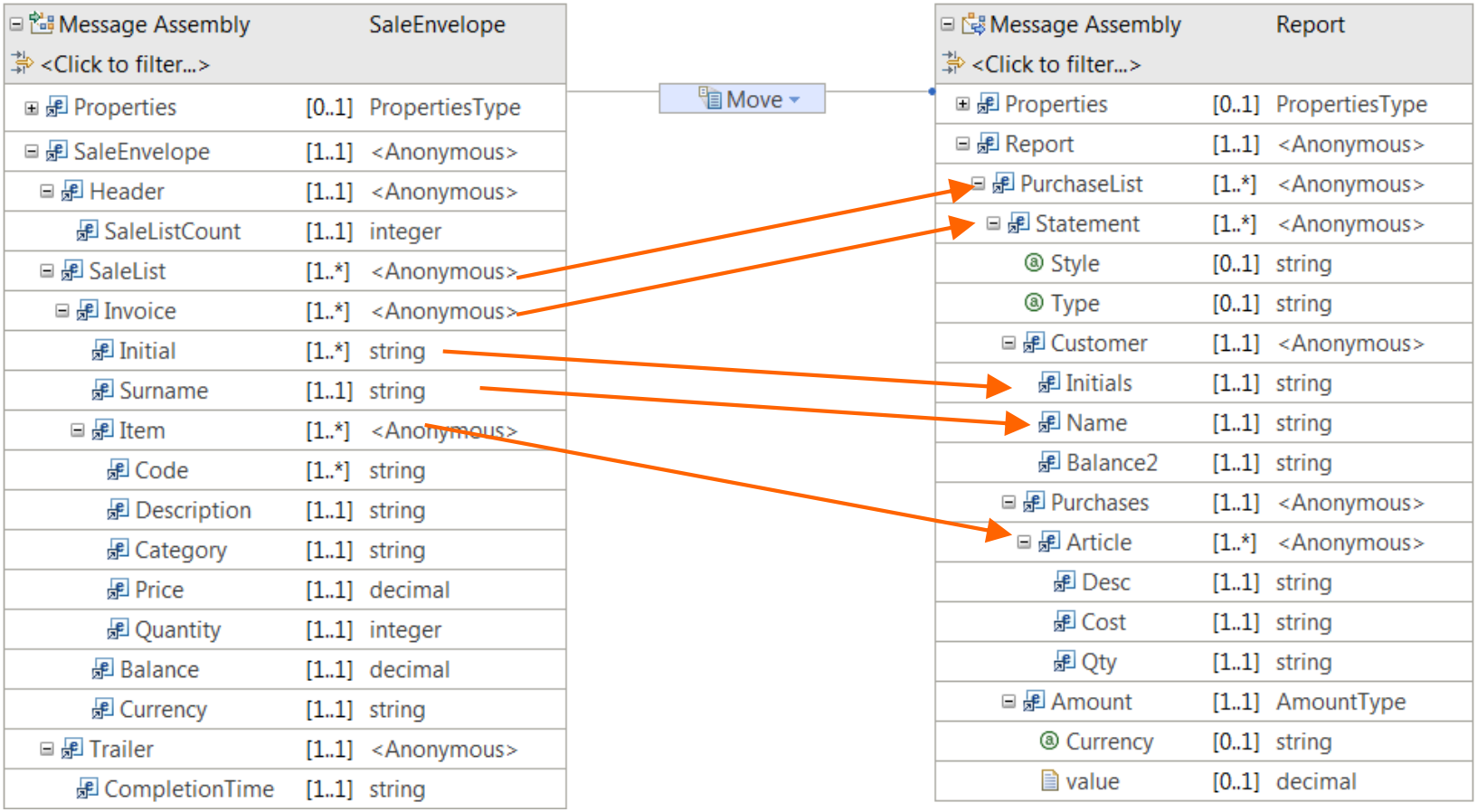
The mapping editor



The mapping editor

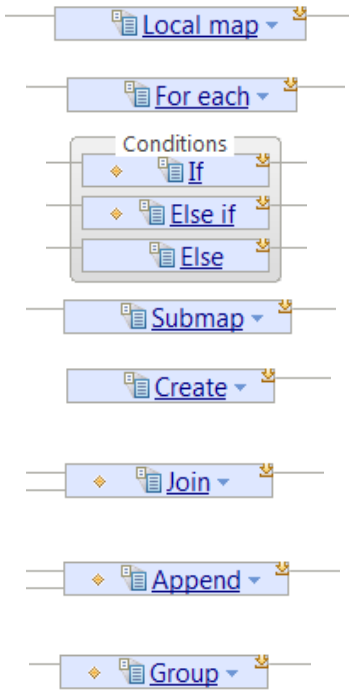


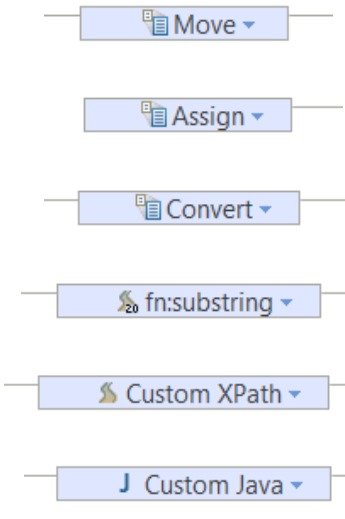
Structured mappings



Transform types

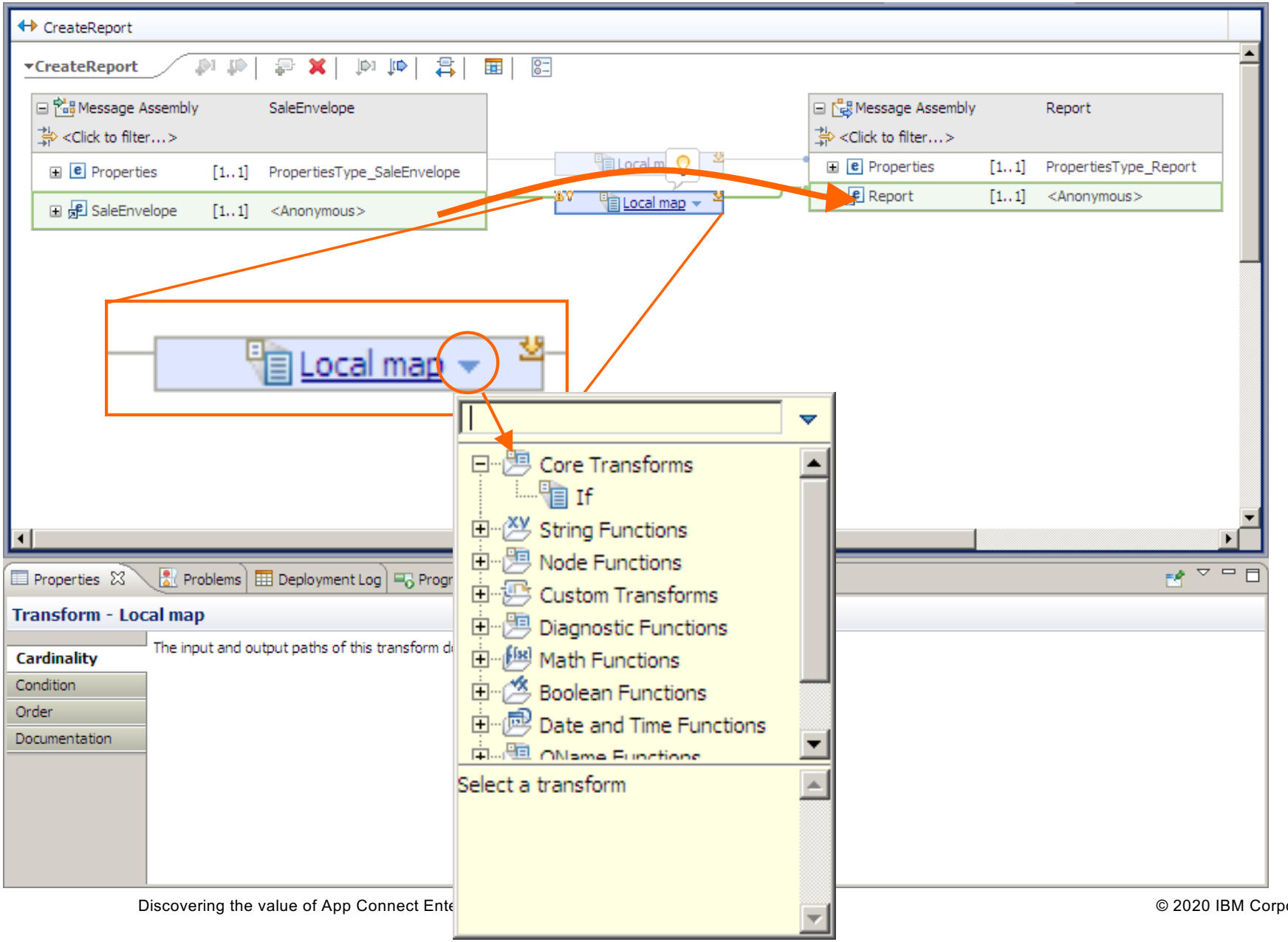
- Local
- For each
- If / Else
- Submap
- Create
- Join
- Append
- Group





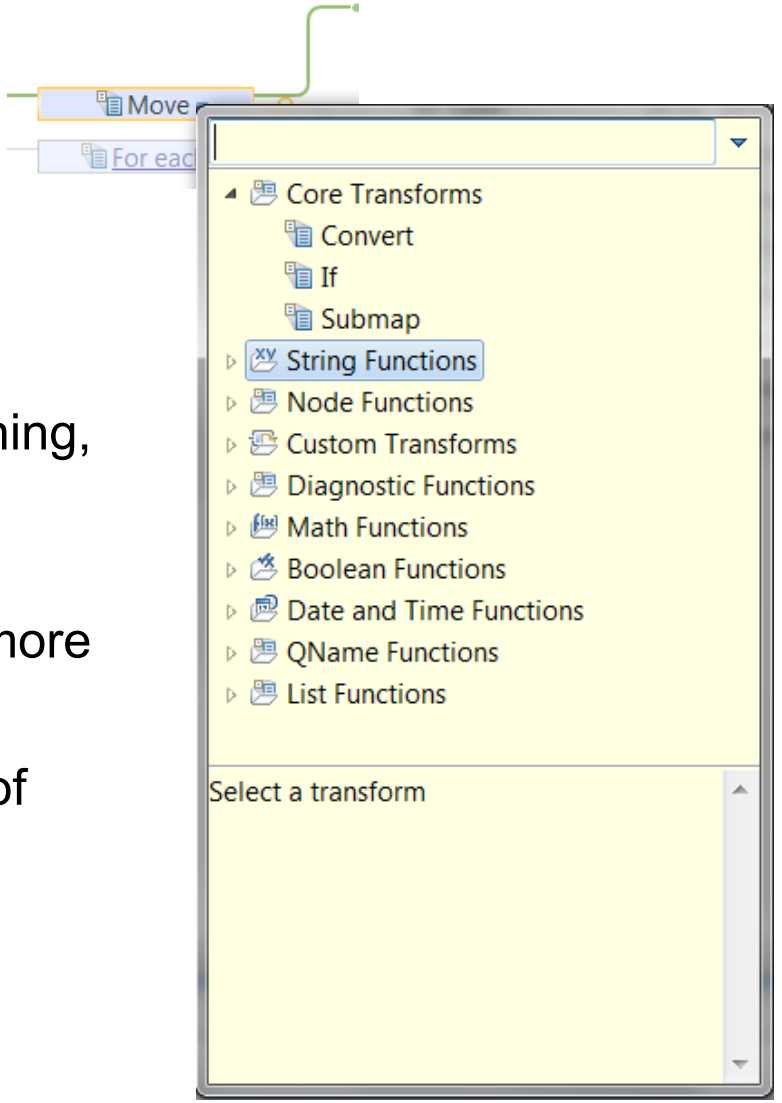
- Move
- Assign
- Convert
- XPath function – categorised
- Custom XPath expression
- Java – user defined
- ESQL – user defined

Creating mappings



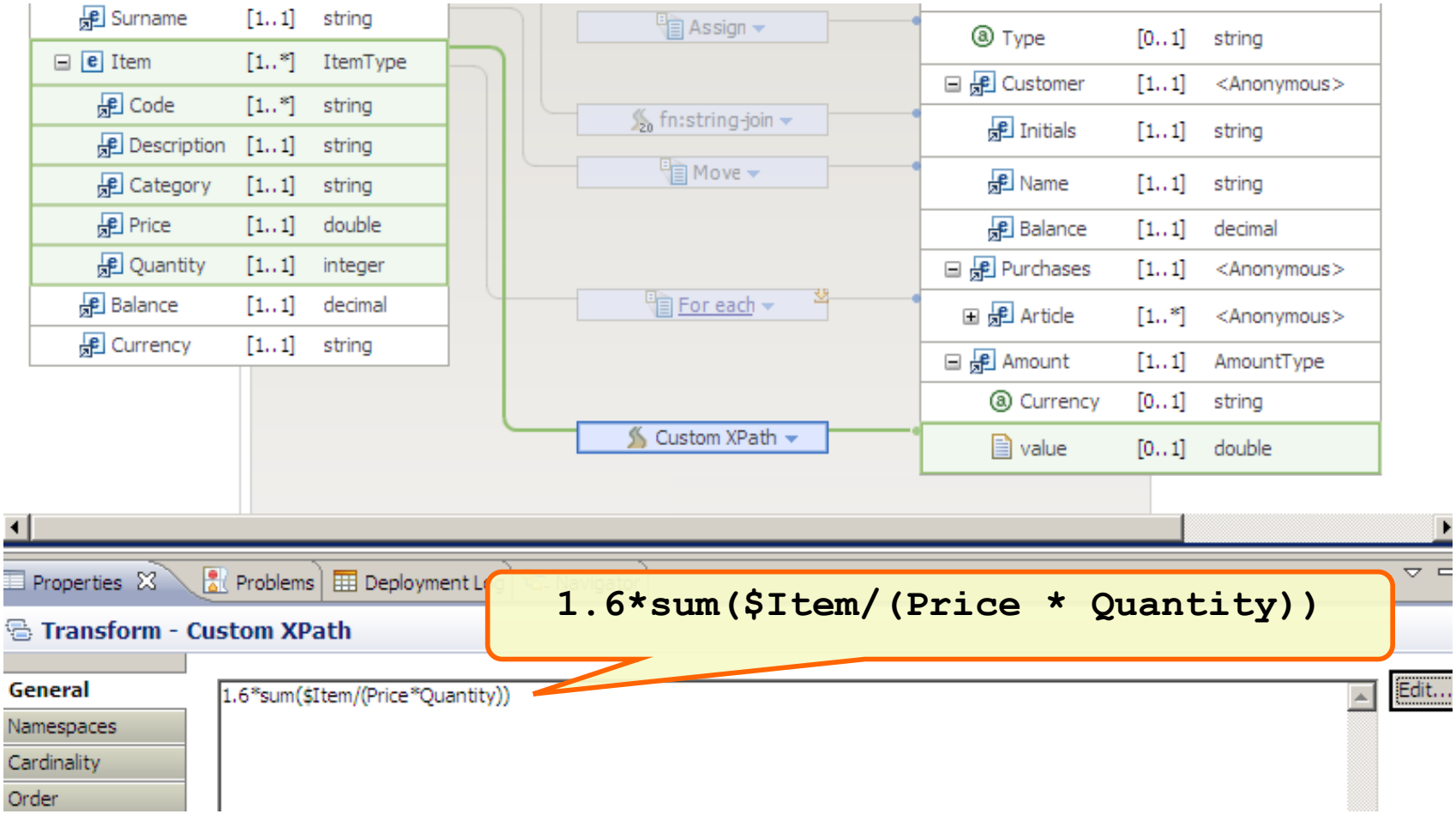
Function transforms

- The target value can be computed by applying a function to one or more inputs
- Large function library from XPath 2.0
- String manipulation
  - For example: concatenation, sub-string, matching, find/replace, regex
- Numeric calculation
  - Counting, summing, rounding, min/max, and more
- Date/time processing
  - Creating timestamps, extracting components of date/time, duration processing



Custom XPath transform

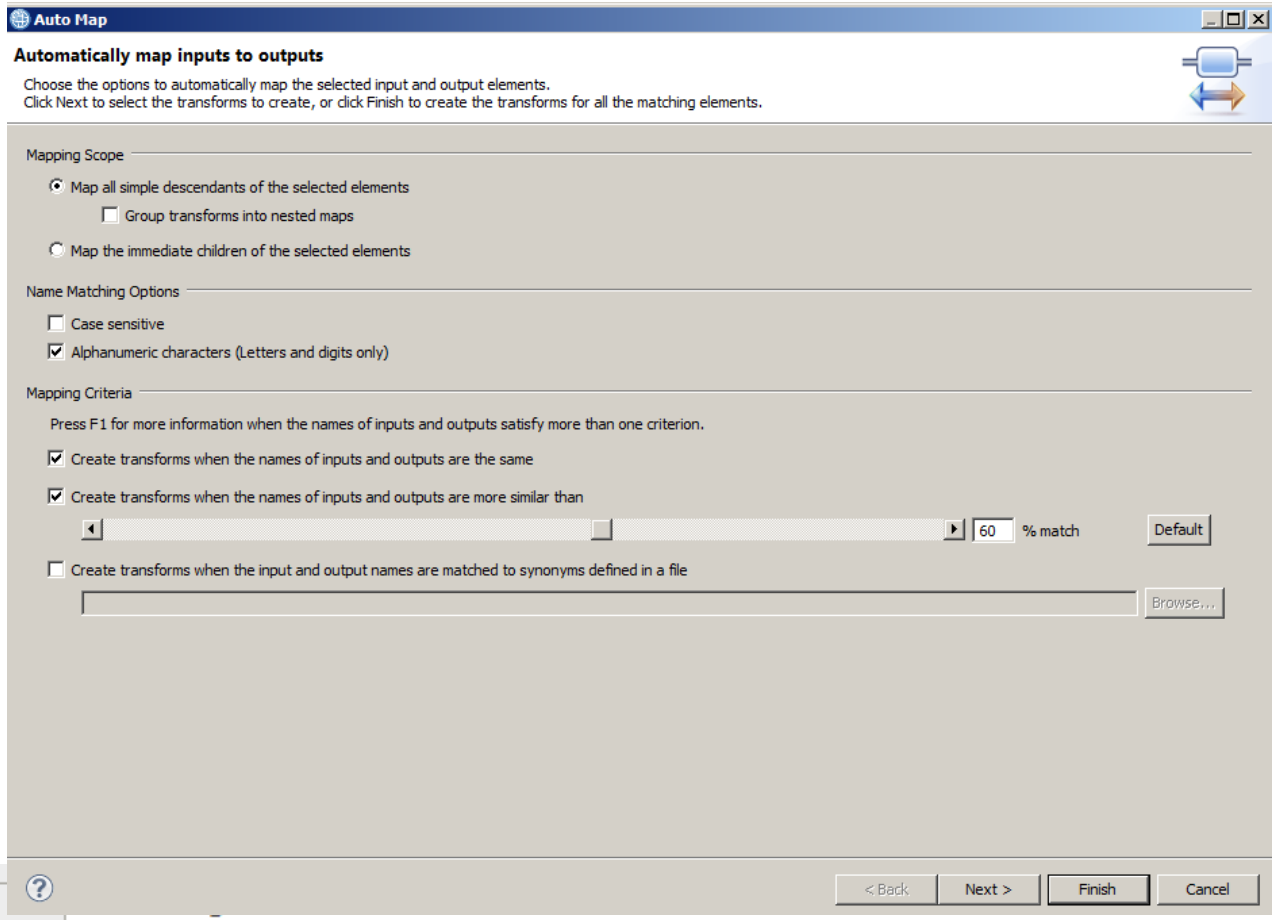
- Transform the input to the output using any legal XPath 2.0 expression

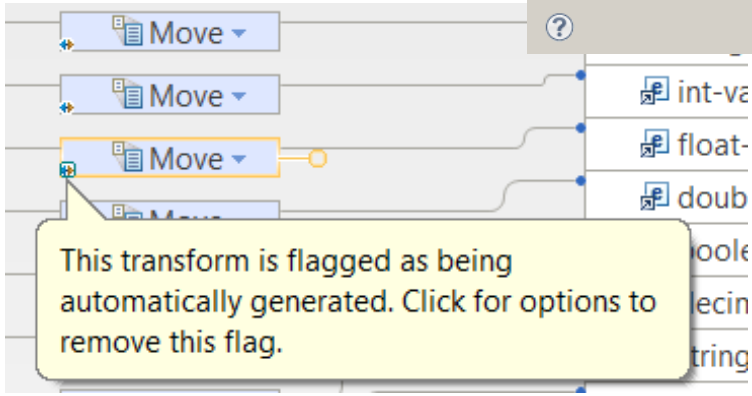


Automap

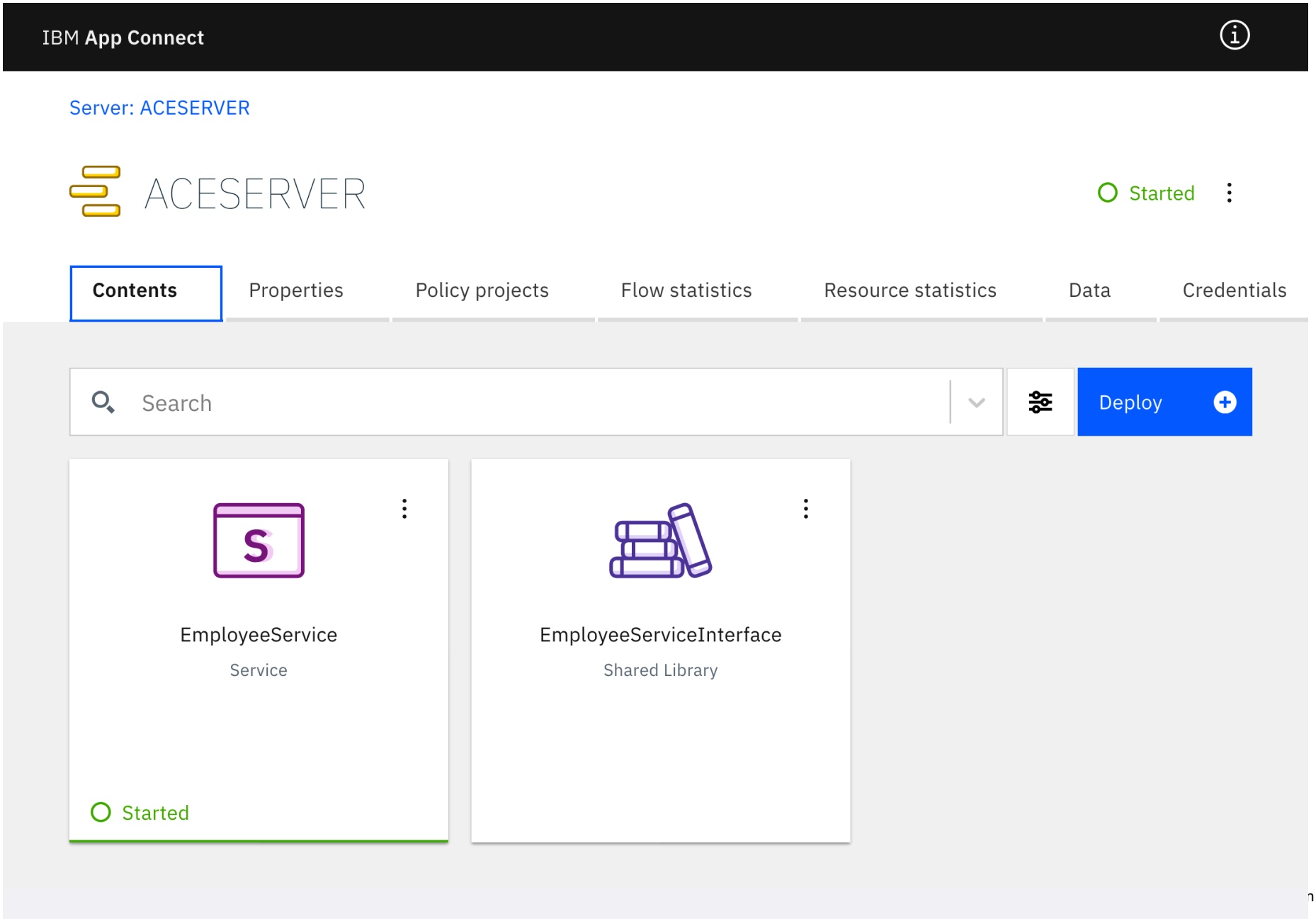
- Automates the task of producing transformations between source and target elements of the same (or similar) names
- Useful for working with large schemas

Explore in Lab 4!





Web administration



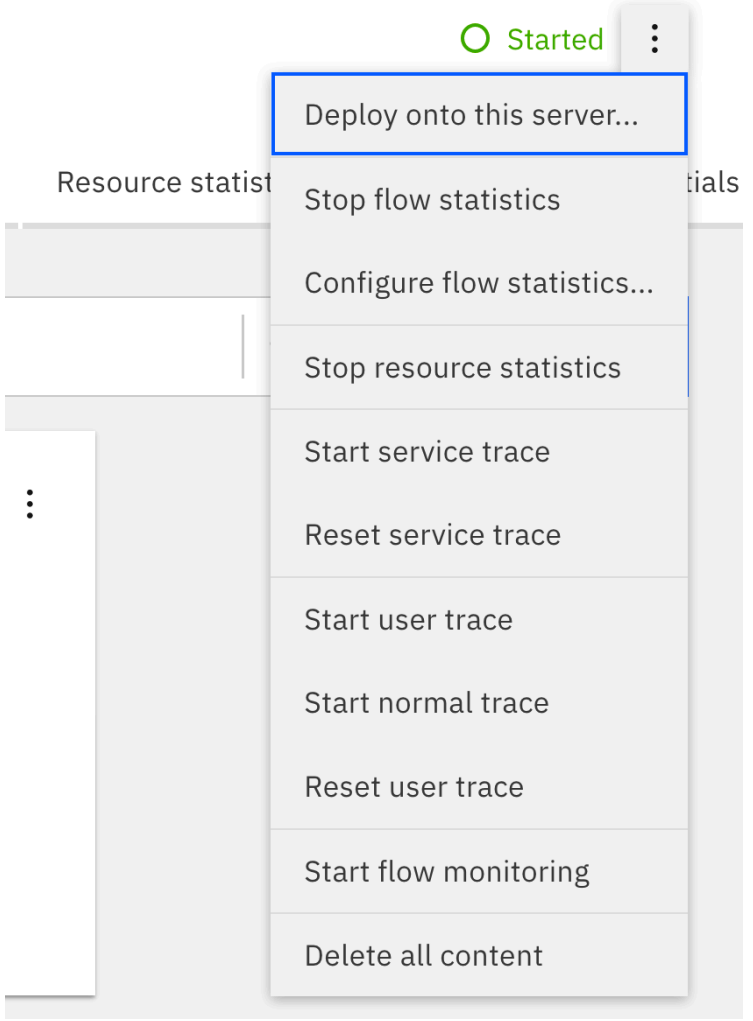
Web administration console

- Provides comprehensive web management interface
  - Focus on non-administrators to understand nodes and resources
- Supports all major browsers: Firefox, IE, Opera, Safari, Chrome
- Easy to configure
  - No extra “moving parts” - uses internal HTTP server to serve data
  - Just start a port for web admin, and begin using
- Access to modify resources

- View resource details with click or button
- Includes full suite of resources
  - Apps, libraries, flows, configurable services and others

Manage resources

- Manage resources deployed to integration servers in the Web UI
  - Deploy BAR files
  - Start/Stop All Application Types
  - Start/Stop All Message Flows
  - Delete All Content
  - Delete specific resources



Resource statistics

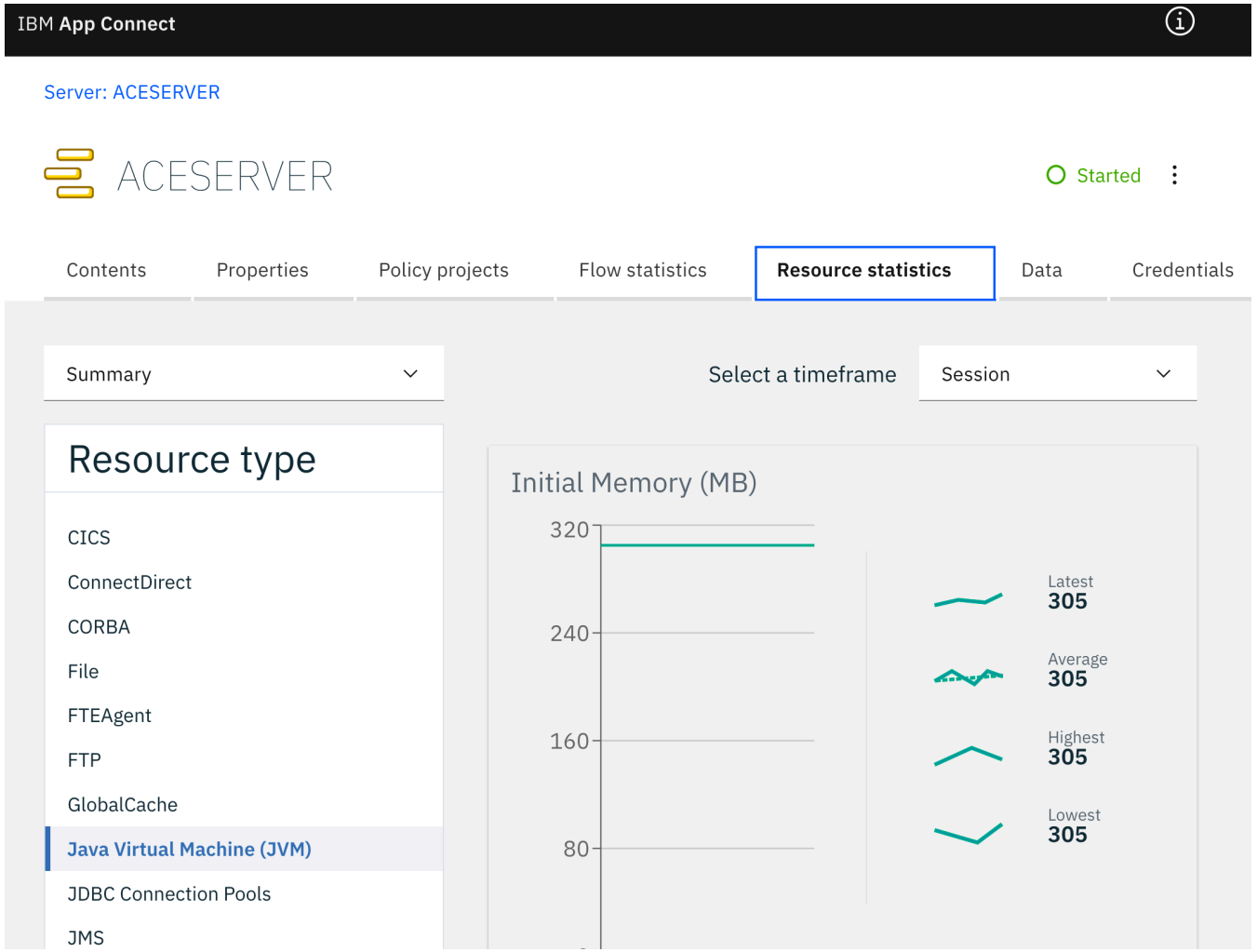
- Resource statistics are collected by an integration server to record performance and operating details of resources:
  - Used to ensure that your systems are using available resources efficiently
  - Can help to pre-empt situations where system resources are overburdened

- If you detect that system resources are under pressure
  1. Analyse the resource statistics data
     - Analysis may require specialist skills and knowledge of each resource type
  2. Assess whether the cause of the concern is the use of those resources by ACE

- Resource statistics collection is not active by default
  - Collection can be activated on individual integration servers using the Web UI
    - Collection may result in a minor performance degradation of the integration node

Viewing resource statistics data

- To view resource statistics data:
  1. Open the Web UI for your integration server
  2. Click the **Resource Statistics** tab



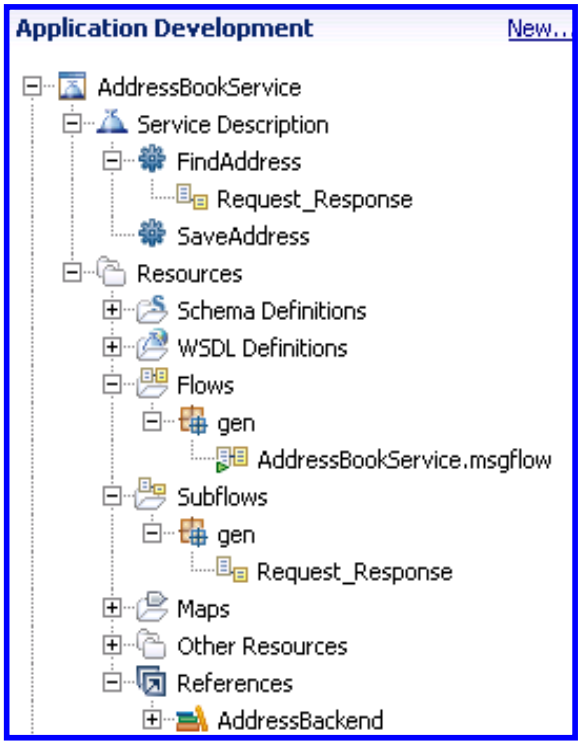
Support for Services

Integration Services

- Integration Services are means of developing, deploying and managing your service oriented integration solutions
- Designed to make development and management of service-oriented solutions easier

- Integration Service
  - An application with a well-defined interface
    - Service interface is expressed via WSDL with a port type
    - Service interface gives *service* application structure
  - Promotes encapsulation and isolation for service oriented integration solutions
  - Can reference one or more libraries
  - Binding must be specified for successful deployment
    - Default binding is created out of the box

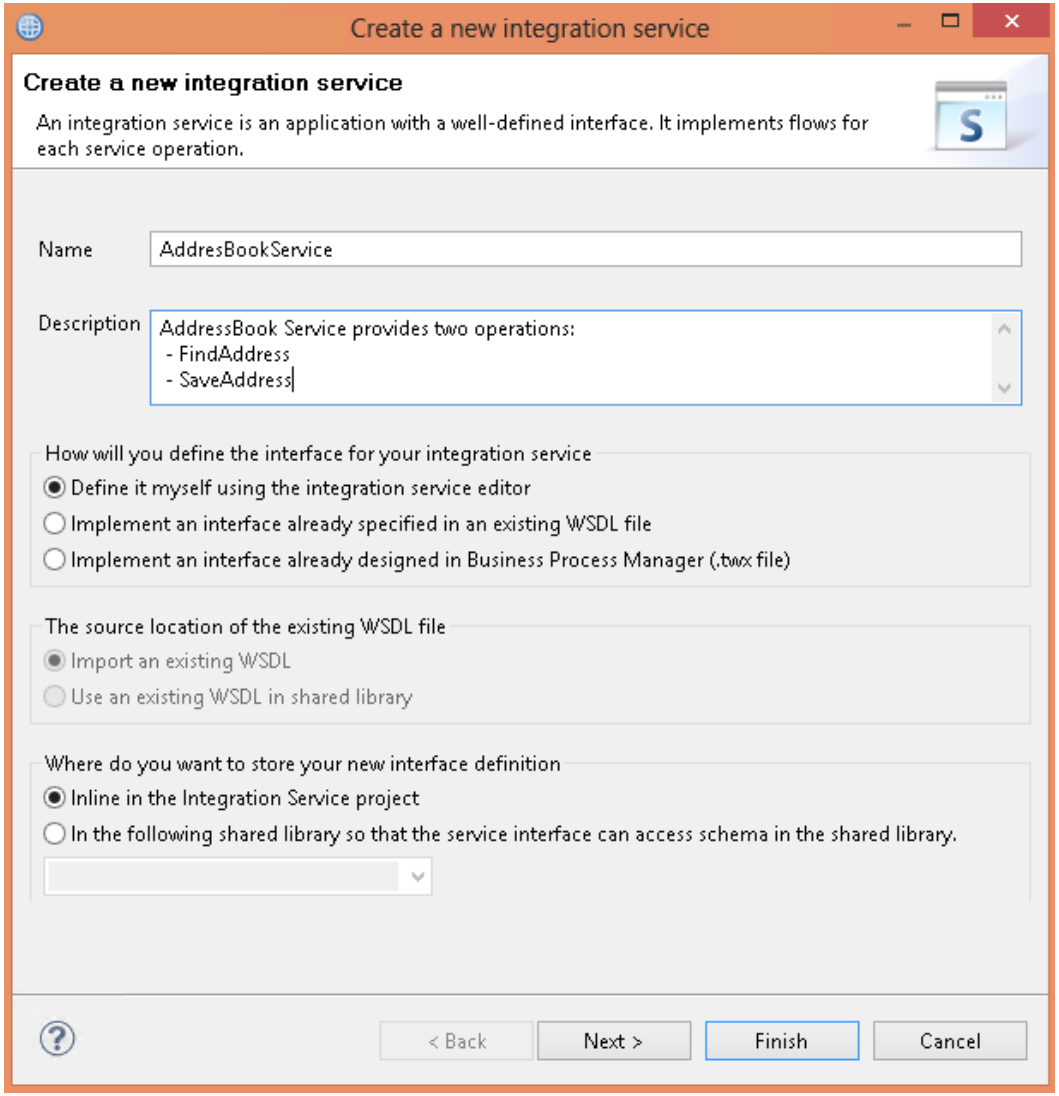
Integration Service anatomy



- WSDL (port type) defines service interface
- Service interface defines one or more operations
- *Service Descriptor* (XML) ties service interface with the service implementation
- Each operation is implemented as one (or more) .subflow and supporting resources (e.g. Maps, ESQL, XSDs)
- Main entry point is implemented as .msgflow, but users don’t need to concern themselves with it.
- Supporting resources may also reside in referenced libraries

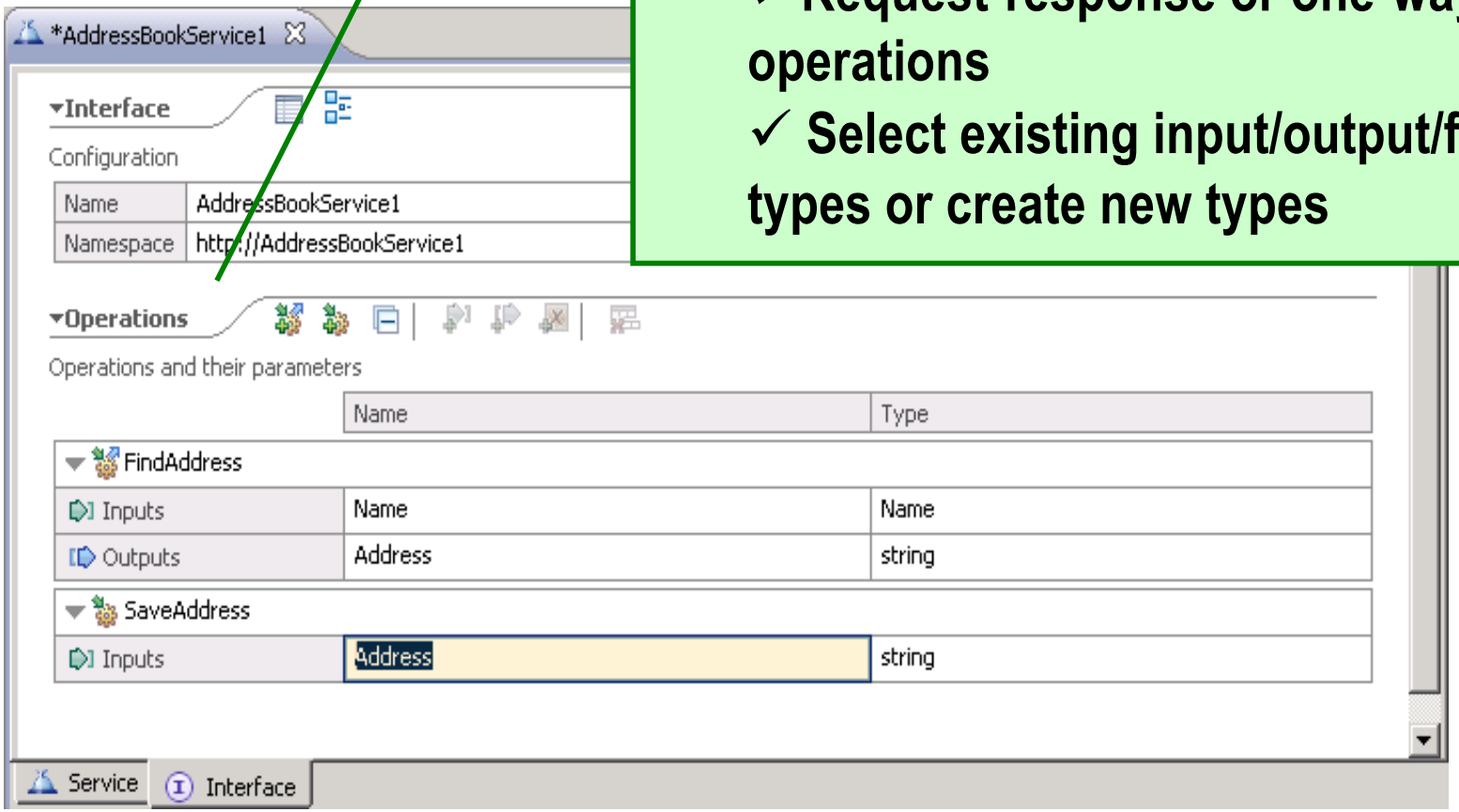
Creating a new Integration Service

- Define it yourself
- Import existing WSDL



Option one – Define it yourself

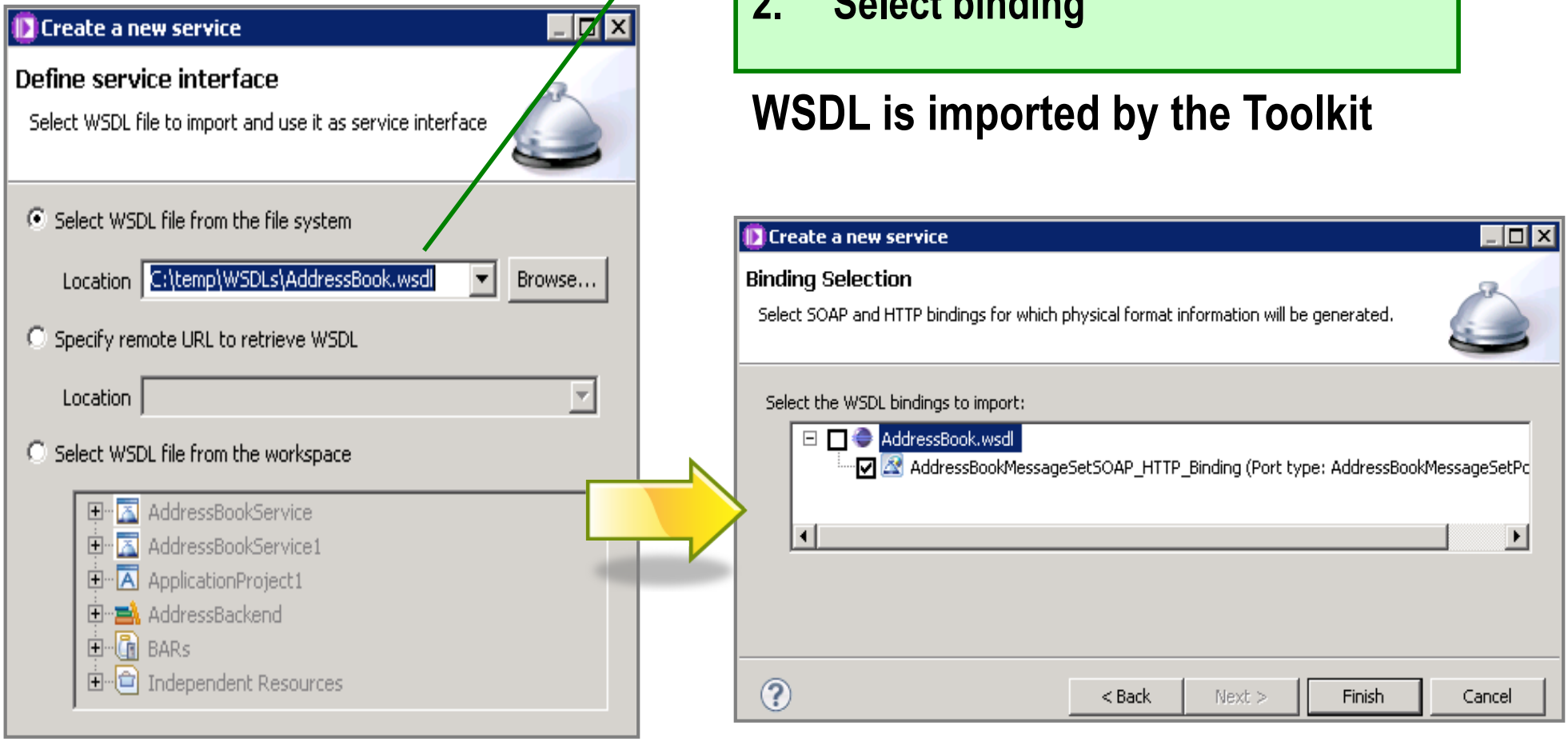
- Integration Service editor comes up with Interface tab getting focus.
- User defines one or more operations, their inputs, outputs and faults.
  - ✓ Request-response or one-way operations
  - ✓ Select existing input/output/fault types or create new types



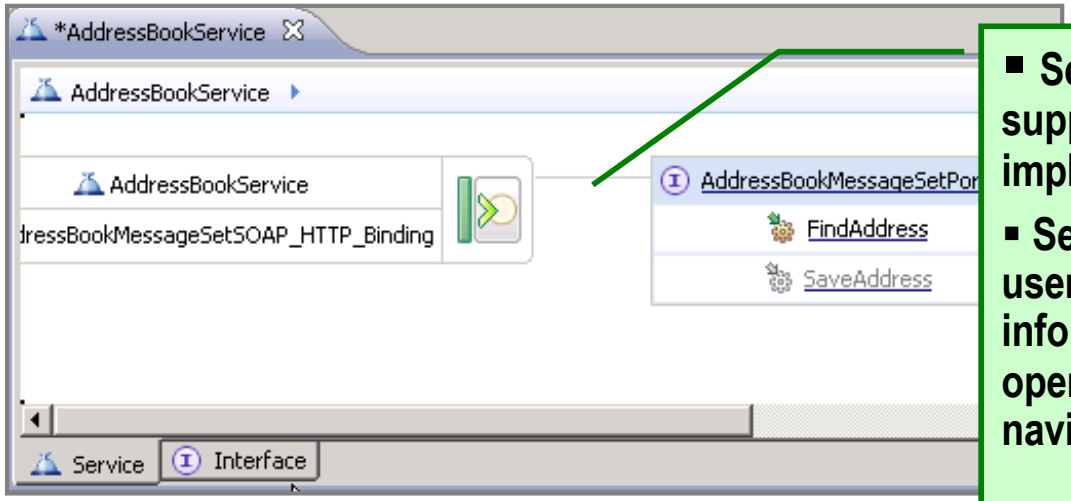
Option two – Use existing WSDL

1. Select existing WSDL file
2. Select binding

WSDL is imported by the Toolkit

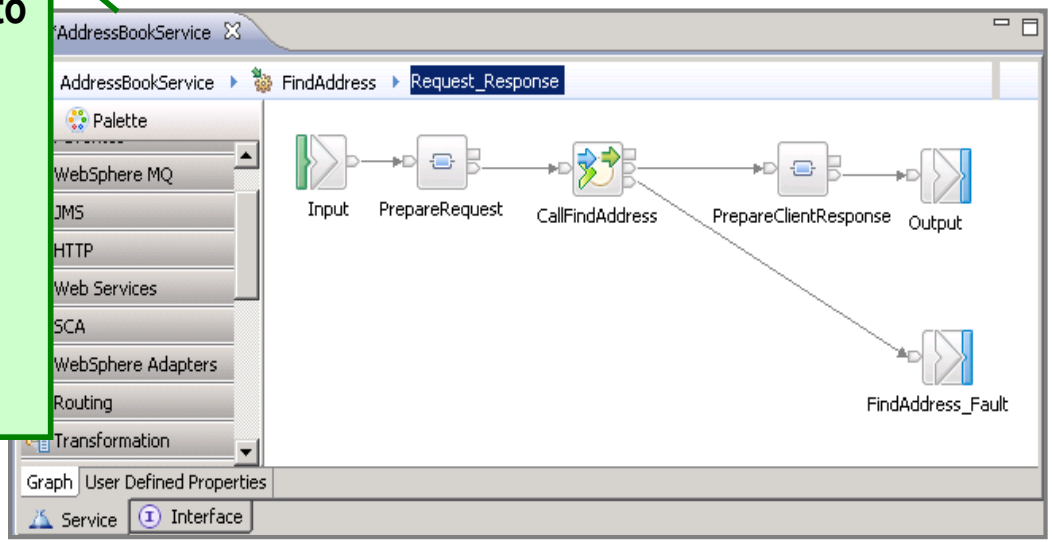


Implementing integration services – Integration Service editor



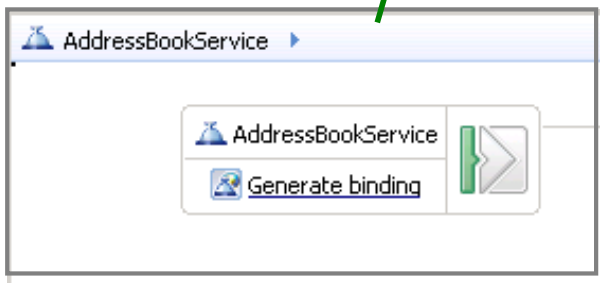
- Service tab in the service editor is for supplying and navigating integration service implementation
- Service Overview is an entry point that gives user an overview of their service (binding information, *function selector*, quick view of operations: implemented/not implemented) and navigational capabilities

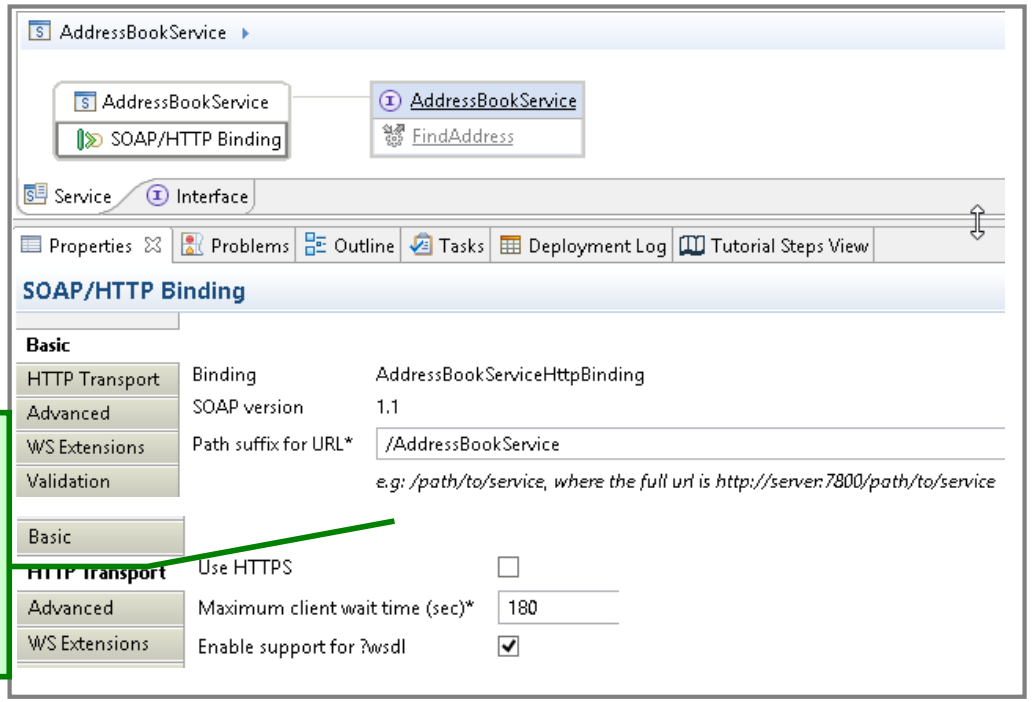
- Selecting particular operation opens a flow editor to provide implementation for service subflow
- One .subflow per operation:
  - Request-Response subflow for request-response operations
  - Request subflow for one-way operations



Integration Service binding

- If Integration Service binding has not been defined, user selects 'Generate Binding' link to complete simple *Binding configuration* wizard.
  - SOAP over HTTP binding
  - REST API



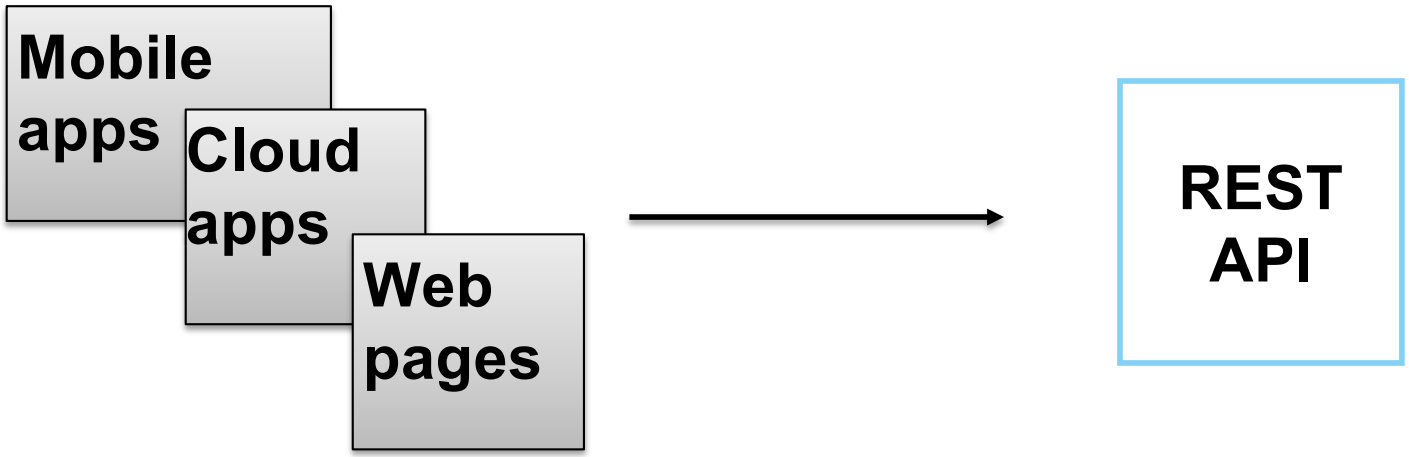


- User can update service binding configuration via Properties view

Note that service interface (WSDL file) cannot be updated via Properties view

REST APIs

- A REST API is a lightweight web service API based on HTTP, and is a much simpler alternative to SOAP based web services.
- A REST API describes a set of resources and a set of operations that can be called on those resources.
- Those operations can be called from any HTTP client—there are HTTP clients available for most programming languages.
- Operations in a REST API can easily be called from JavaScript code running in a web browser, or application code running on a mobile device.



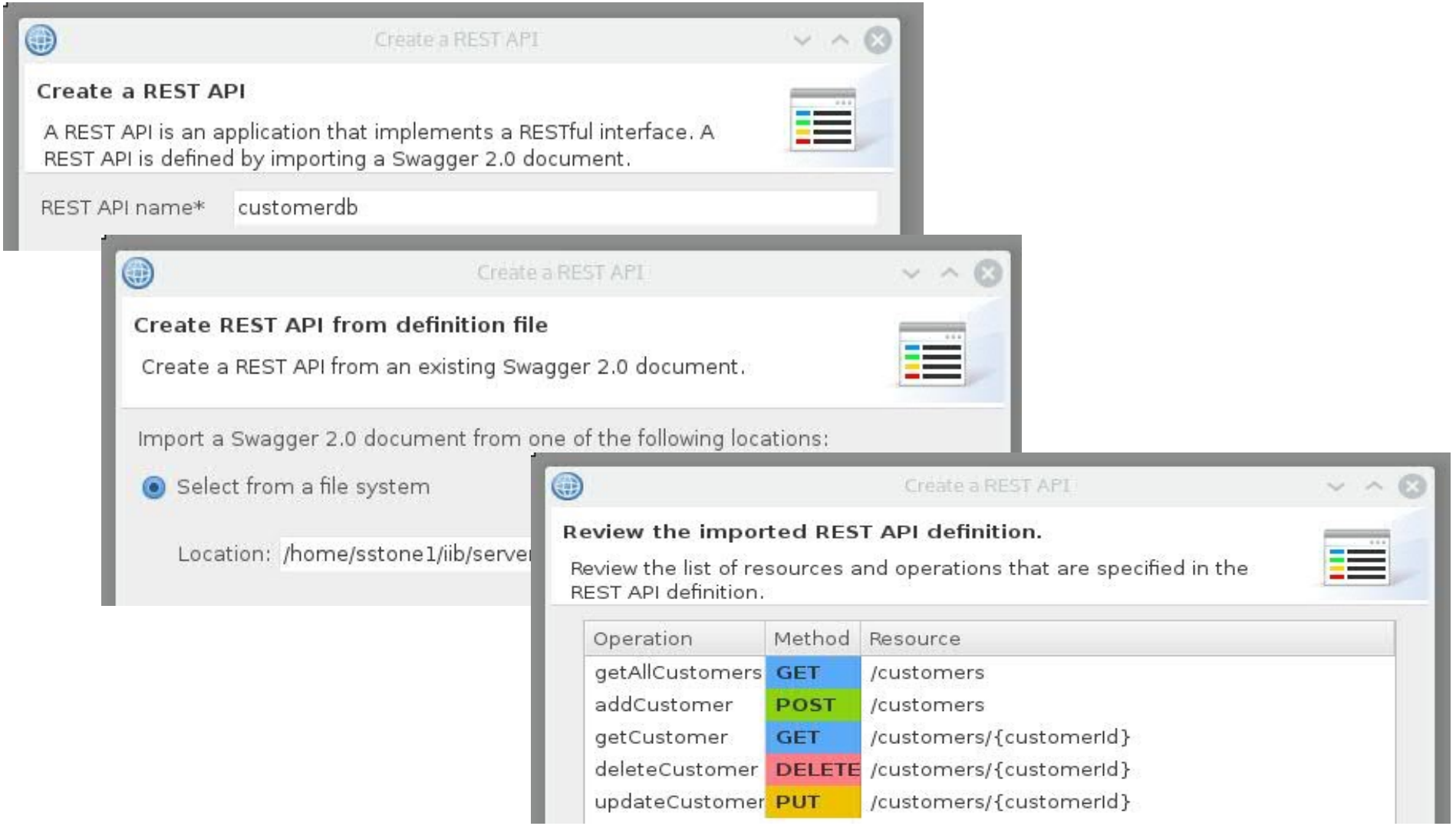
REST APIs – Swagger

- Swagger is an open standard for defining a REST API:
  - http://swagger.io/
- Along with the specification, there is a set of open source tooling that can be used to interact with Swagger documents and the REST APIs that they describe.
- A Swagger document includes definitions of the resources, operations, and parameters in a REST API. It can also include JSON Schema that describes the structure of the request and response bodies to an operation.
- A Swagger document can be thought of as the REST API equivalent of a WSDL document for a SOAP web service.
- App Connect Enterprise supports Swagger 2.0. The specification for Swagger 2.0 can be found at:
  - https://github.com/swagger-api/swagger-spec/blob/master/versions/2.0.md
- In order to build a REST API in App Connect Enterprise, you must develop and supply a Swagger 2.0 document that describes the REST API you are going to build.

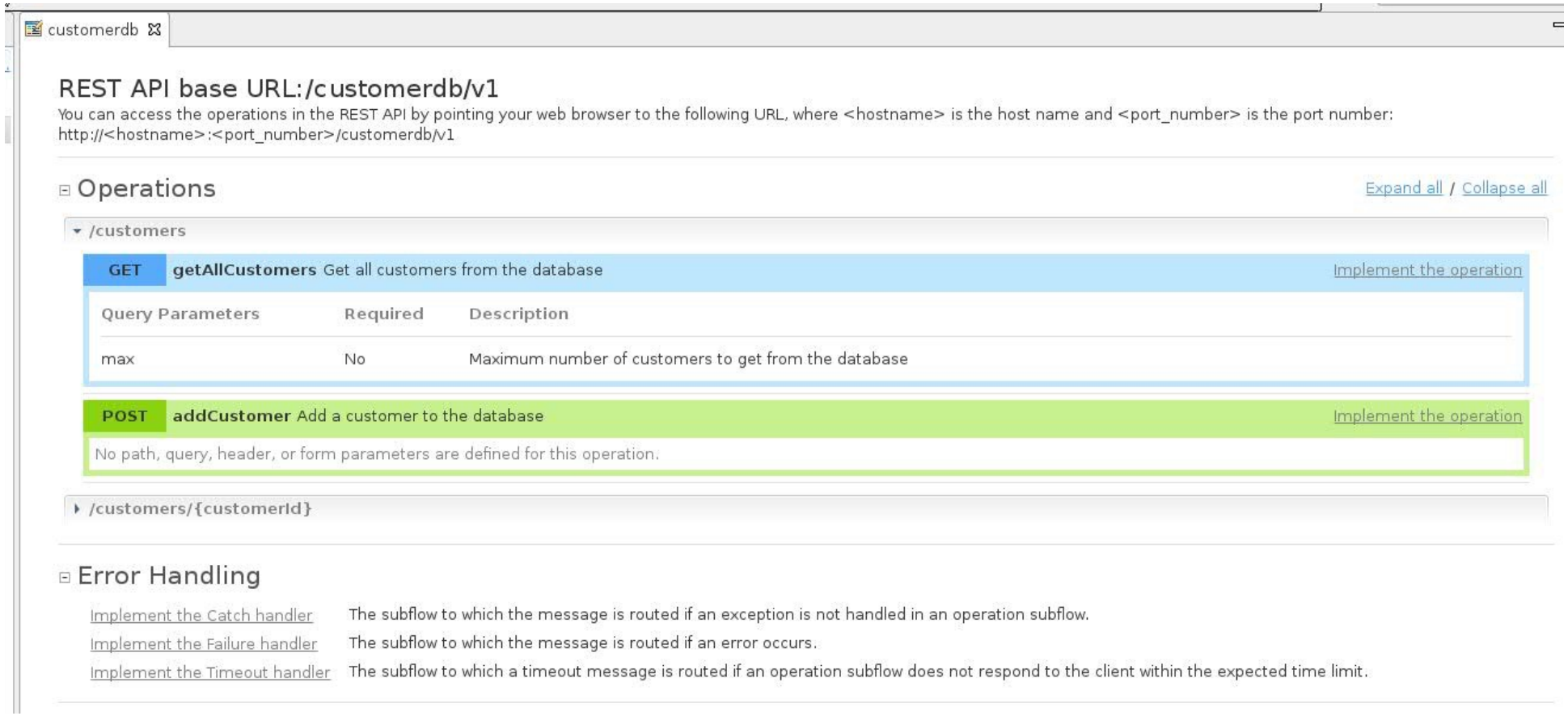
REST APIs – REST APIs in App Connect Enterprise

- Operations defined in the REST API are implemented as normal subflows.
- The REST API container automatically takes care of the routing of inbound HTTP requests to the correct subflow for the operation being called.
- You simply need to connect the dots between the Input and Output nodes in each subflow!
- REST APIs support all of the App Connect Enterprise features that you can use with applications (such as shared libraries, monitoring, activity log), and all message flow nodes can be used within a REST API.

REST APIs – New project wizard



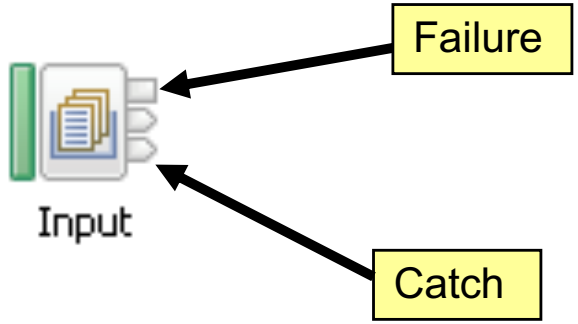
REST APIs – REST API Description view



Exception handling and debugger

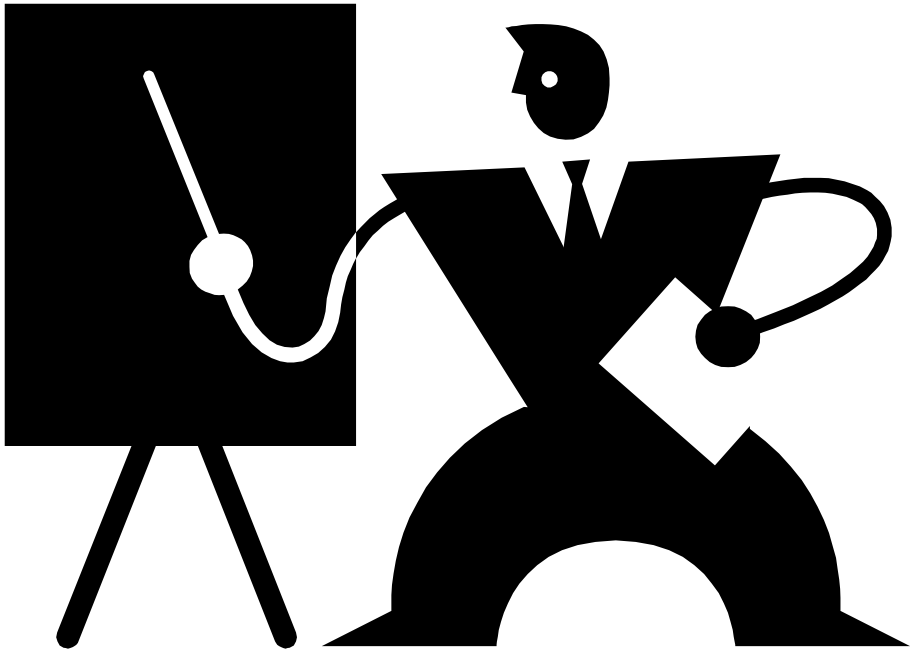
Robust exception handling features

- **Should be part of any message flow**
  - ***Failure*** terminals catch runtime exceptions within node
  - Input nodes have ***Catch*** terminals
  - Common use for a reusable subflow
    - Handle downstream exceptions



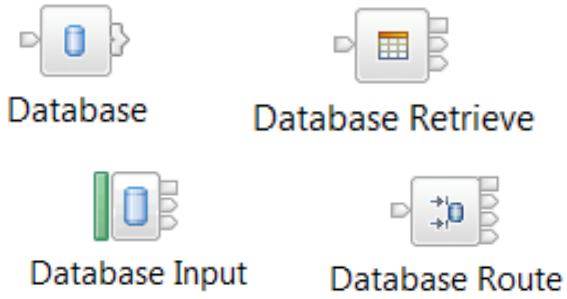
- **Exception handling nodes**
  - Catch unhandled exceptions (**Try/Catch**)
    - More granular error handling
  - Throw exceptions (**Throw**)
    - Or rethrow “caught” exceptions
  - ***Trace*** node can log diagnostic information

Additional features worth mentioning



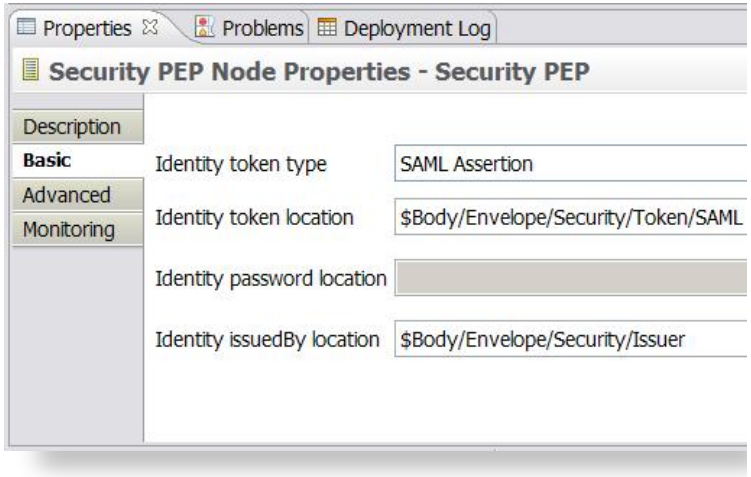
Extensive database support

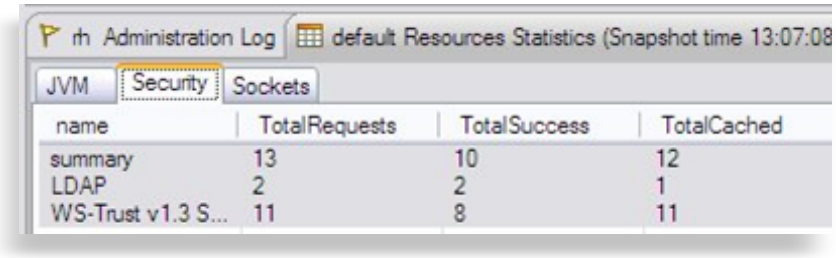
- Convenient and easy access to databases
- Read, insert, update, delete database content
- Uses include:
  - Routing decisions via DB lookup
  - Triggering flow from DB event
  - Simple retrieval of row from DB table
  - Insert/Update databases for tracking purposes
- Extended SQL/2000
- Extract database definitions
  - Data Source Explorer tool
- Database source/target in a map
- Databases accessible in Compute, JavaCompute, .NETCompute, and Filter nodes
- Stored procedures support
- JDBC and ODBC supported
- IBM DB2, IBM Informix, Oracle, Sybase, SQL Server, solidDB
- Open Driver Manager allows ACE to connect to even more ODBC data sources
  - For example, MySQL, PostgreSQL, Teradata, Cache, Progress…



Extensive security capabilities

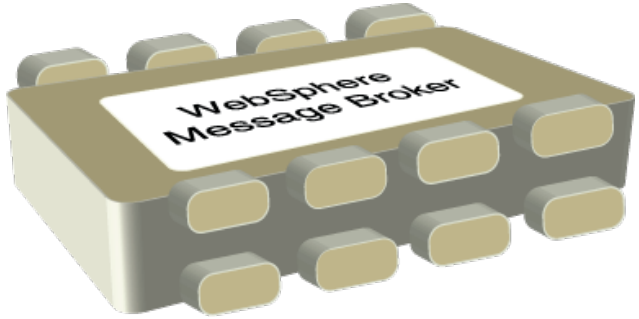
- **Input nodes can act as policy enforcement point (PEP)**
  - Authentication, authorization and identity mapping
  - UserID/Password, X509 tokens, SAML, Kerberos, LTPA & RACF PassTickets supported
  - Users can easily extend to support custom and non-standard tokens
- **PEP node enables mid-flow checking**
  - Placed anywhere in message flow to perform security functions
- **WS-Trust 1.3, TFIM and LDAP supported PDPs for token authentication and authorization**
- **Operational control**
  - Security profiles allow operational reconfiguration of PDP without redeploy
  - Resource manager security statistics; totals, passes, exceptions, cached





App Connect Enterprise – Summary

- **Powerful, flexible, extensible, production-strength product**
- **Key concepts**
  - ***Message flows*** represent application connectivities
  - ***Message nodes*** modularize integration operations
  - ***Message Tree*** and ***Logical Message Model*** provides focus on business data
  - ***Patterns*** enable rapid, top-down development of connectivity solutions
- **Rich, simple message and database processing using Graphical Mapping**
  - Java, ESQL, .NET can also be used
- **Support for Web Services, Enterprise Messaging, and a variety of transports**
- **Scalable architecture for high capacity**
- **App Connect Enterprise is a key IBM connectivity technology**
  - Unparalleled range of connectivity options and capabilities
  - Supports users' range of experience and needs
  - Industry-leading performance in a broad range of scenarios



Questions?