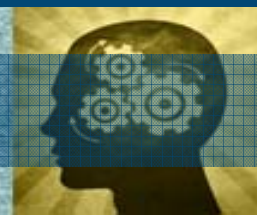
Fiorano
Peer-to-Peer Dataflow Pipelines™

ESB BEST PRACTICES



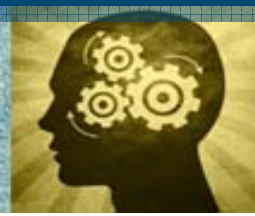
3 INTEGRATION SCENARIOS

DATA CONSISTENCY: INDEPENDENT APPLICATIONS “GET THE FACTS STRAIGHT”.

MULTI-STEP PROCESS: INDEPENDENT APPLICATIONS IMPLEMENT A BUSINESS PROCESS

COMPOSITE APPLICATION: NEW FUNCTIONALITY BY TYING TOGETHER EXISTING APPLICATIONS.

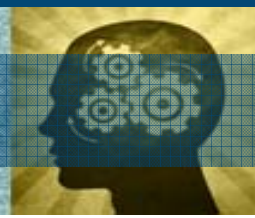
AGENDA



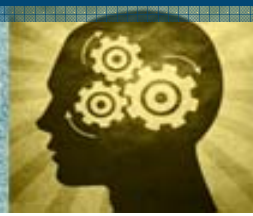
- INTRODUCTION TO SOA
- DEMYSTIFYING ESB
- IMPLEMENTATION METHODOLOGY OF ESB
  - STEP I – REQUIREMENTS GATHERING
  - STEP II – IDENTIFYING “MUST HAVE” FEATURES
  - STEP III – COMPONENTIZING BUSINESS PROCESS
  - STEP IV – DEFINING DISTRIBUTED ESB PROCESS
  - STEP V – DEPLOYING ESB PROCESS
  - STEP VI – LAUNCHING AND MONITORING ESB PROCESS
  - STEP VII – CHANGE MANAGEMENT, VERSION ROLLOVER
- INTEGRATION SCENARIOS – PUTTING THEORY TO PRACTICE

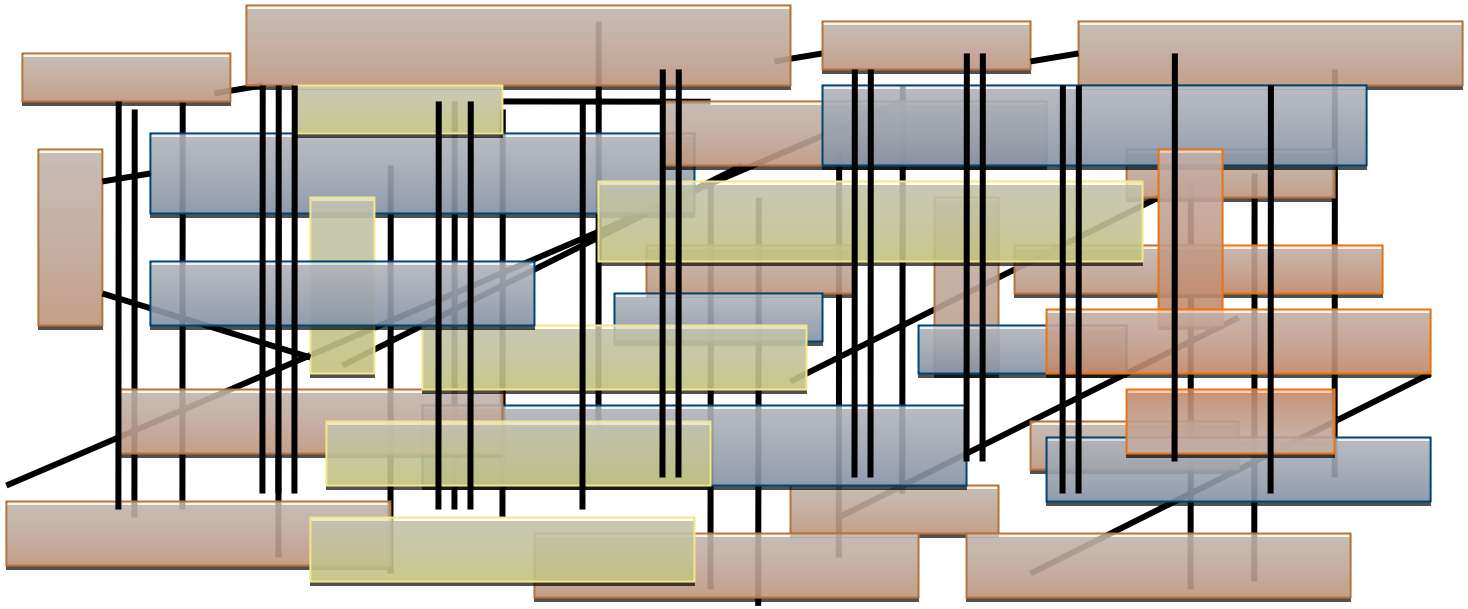
Fiorano
Peer-to-Peer Dataflow Pipelines™

INTRODUCTION TO SOA



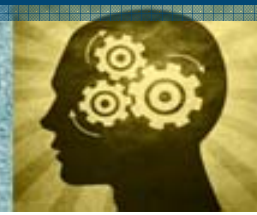
INTRODUCTION TO SOA
POINT TO POINT INTEGRATION



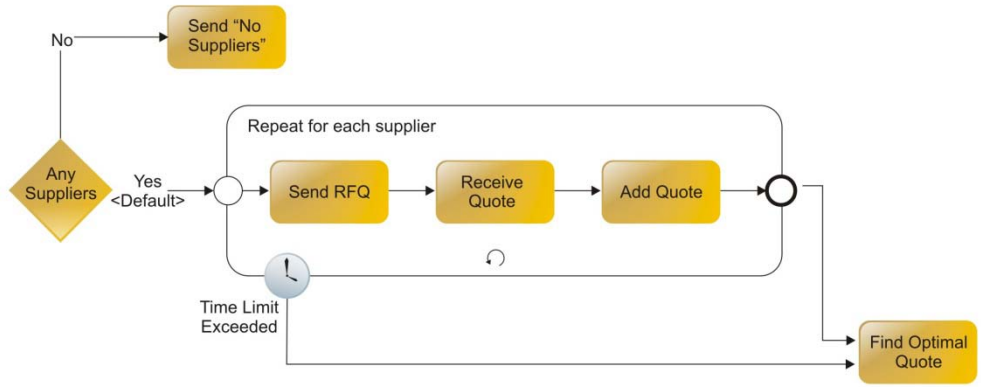


- "Accidental" Architecture (Synchronous, Fine-grained, Not scalable, Many connections and data formats, Extremely difficult to manage, Expensive to maintain and extend)
- Hinders business change: new products, channels, suppliers, etc.
- Slow and expensive maintenance, No Reusability
- Expensive to Manage, Monitor and Extend
- => Need realistic way to evolve new infrastructure and add value

BUSINESS PROCESSES BECOME AGILE WHEN IMPLEMENTED AS SERVICES

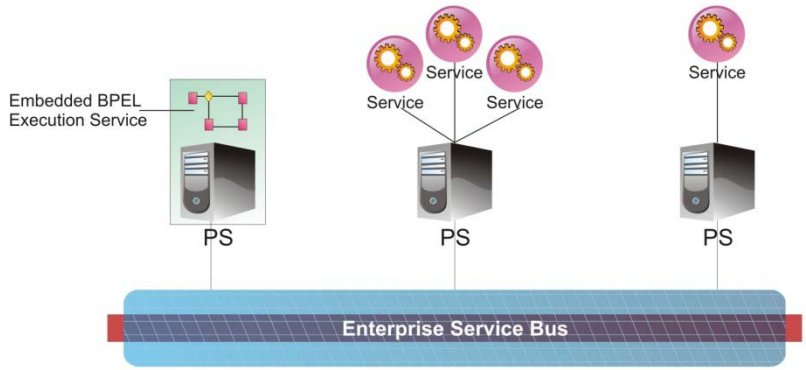


Business Perspectives
- Simplification
- Elimination
- People
- Process Performance Measurement
- Simulation modeling
- Agility
- etc.



BPMN Notation representing a business process with an expanded Sub-process

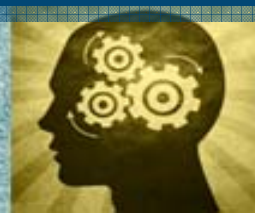
Systems Perspectives
- Working Software
- Reliability
- Configuration Management
- Scalability / Performance
- Reusability
- Speed of Delivery
- etc.

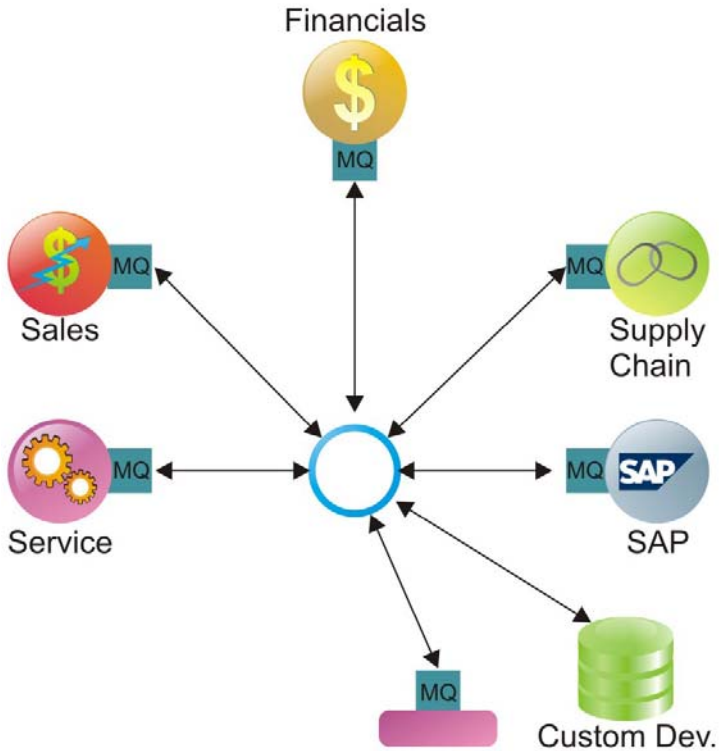


Example: Processes implemented as Services

With SOA – Processes Can Be Layered on Top of Existing Infrastructure

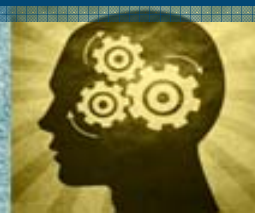
INTRODUCTION TO SOA – MOM
CHALLENGES

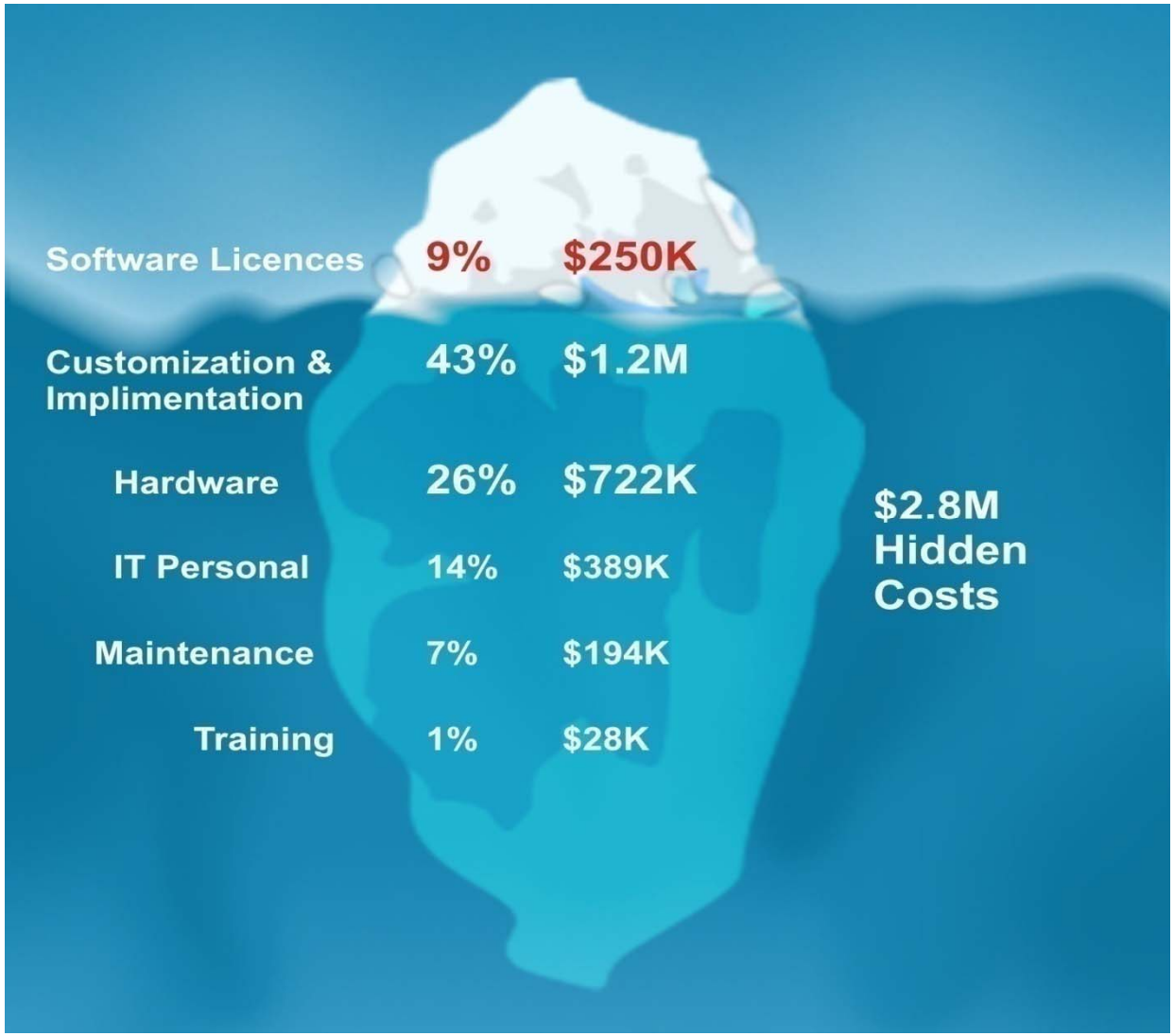




- Easy Inter-Operability
- Easy Common Data Model for integration
- Robust Process Management
- Scalability and High-Availability
- Distributed services management – Real time configuration changes
- Reusability of Components & Services in an effective IDE
- Portable Adapters
- Cost, Speed, Quality & Control

THE INTEGRATION BROKER ICEBERG

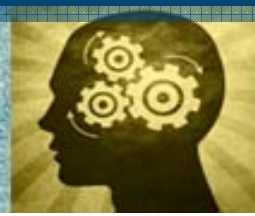




Fiorano
Peer-to-Peer Dataflow Pipelines™

INTRODUCTION TO SOA - WEB SERVICES

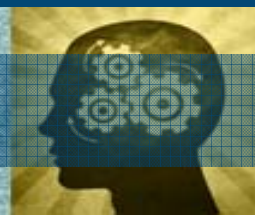
CHALLENGES



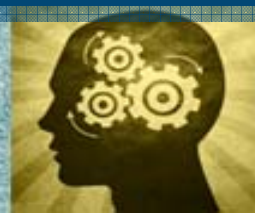
- Suffer from incomplete and competing standards definition that might not ensure easy interoperability across different implementation
- Interoperabilty
  - ✓ SOAP
  - ✓ Only binding for HTTP; no bindings for SMTP, JMS
  - ✓ “SOAP encoding”: XML ←→ Objects
  - ✓ RosettaNet, EDIINT: no SOAP
  - ✓ SOAP 1.1 → 1.2 (W3C)
- WSDL: RPC/Encoded vs. Document/Literal
- Do not offer a complete package but provide only the functionality of access and invocation leaving majority of the development work to the user
- Business Semantics - Orchestration & Choreography
- Security
  - ✓ HTTPS is not enough
  - ✓ No agreement regarding attachments
  - ✓ S/MIME in RosettaNet and EDIINT
  - ✓ Single Signon
- Transactional Integrity
- Reliable Asynchronous Message Handling
  - ✓ WS-Reliability (IBM, MS) vs. WS-ReliableMessaging (OASIS)
- Transformation Services

Fiorano
Peer-to-Peer Dataflow Pipelines™

DEMYSTIFYING ESB



DEMYSTIFYING ESB

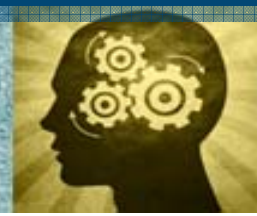


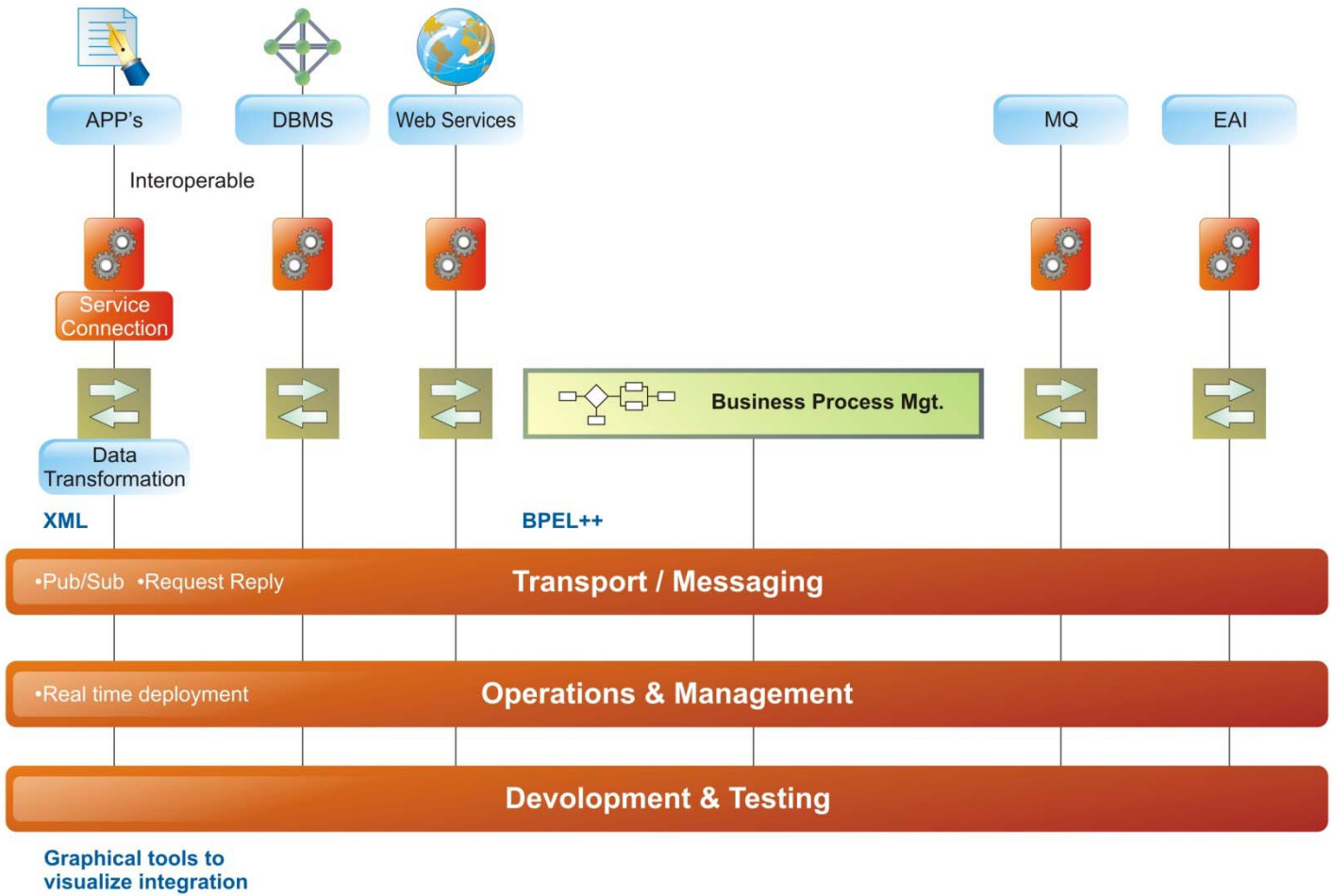
Enterprise Service Bus evolved out of SOA

- IT Components can be accessed as services
- Defined form of invocation and entry points to service
- Business process - Event-driven/Asycnhronous Inovcation of Services
- Composite Application – mix of existing and new components

Fiorano
Peer-to-Peer Dataflow Pipelines™

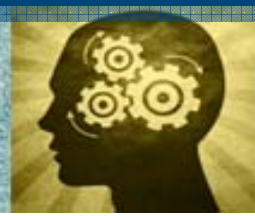
FUNDAMENTALS OF SOA FOR INTEGRATION





Fiorano
Peer-to-Peer Dataflow Pipelines™

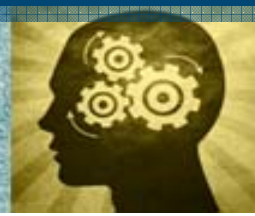
DEMYSTIFYING ESB



Definition (An Affordable EAI) ?

An ESB is a standards-based, service-oriented backbone capable of connecting hunders of application endpoints. ESBs combine messaging, Web Services, XML, data transformation and management to reliably connect and coordinate application interaction. The ESB deployment model is an integrated network of collaborating service nodes, deployed in service containers.

DEMYSTIFYING ESB

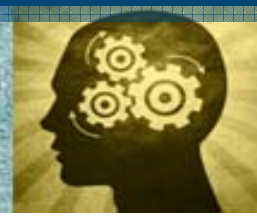


Enterprise Service Bus – Standards based Integration

- Communication and data routing (JMS)
- Data protocols (XML)
- Transformation (XSLT)
- Content Based Routing (CBR)
- Connectivity (JCA)
- WebServices
- Security
- *Business Process Management (BPM)*
- *Pre-built Business Components*
- *Business Process Modelling (BPEL)*
- *B2B – trading partner management*

Fiorano
Peer-to-Peer Dataflow Pipelines™

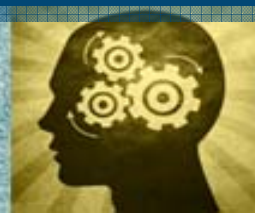
WHY STANDARDS FOR INTEGRATION?
BUSINESS AND TECHNOLOGY DRIVERS



- Increases ability to integrate
  - ✓ No platform or vendor technology dependencies
- Lowers cost of integration
  - ✓ Minimizes need for expensive proprietary adapters
  - ✓ Larger talent pool, broader education offerings
- Integration projects become more predictable

STANDARDS-BASED INTEGRATION

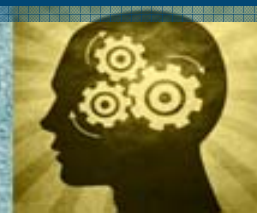
BACKBONE OF STANDARDS BASED INTEGRATION



- Deploying a Service-Oriented Architecture (SOA)
- Use of XML
- Rise of JMS as the de-facto for underlying communication mechanism
- BPEL for business process
- Connections to Packaged Apps, Legacy Systems using J2EE Connector Architecture (JCA)
- Web Services

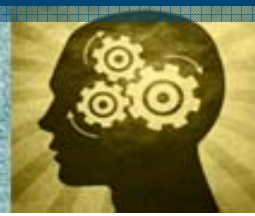
SERVICE-ORIENTED ARCHITECTURE (SOA)

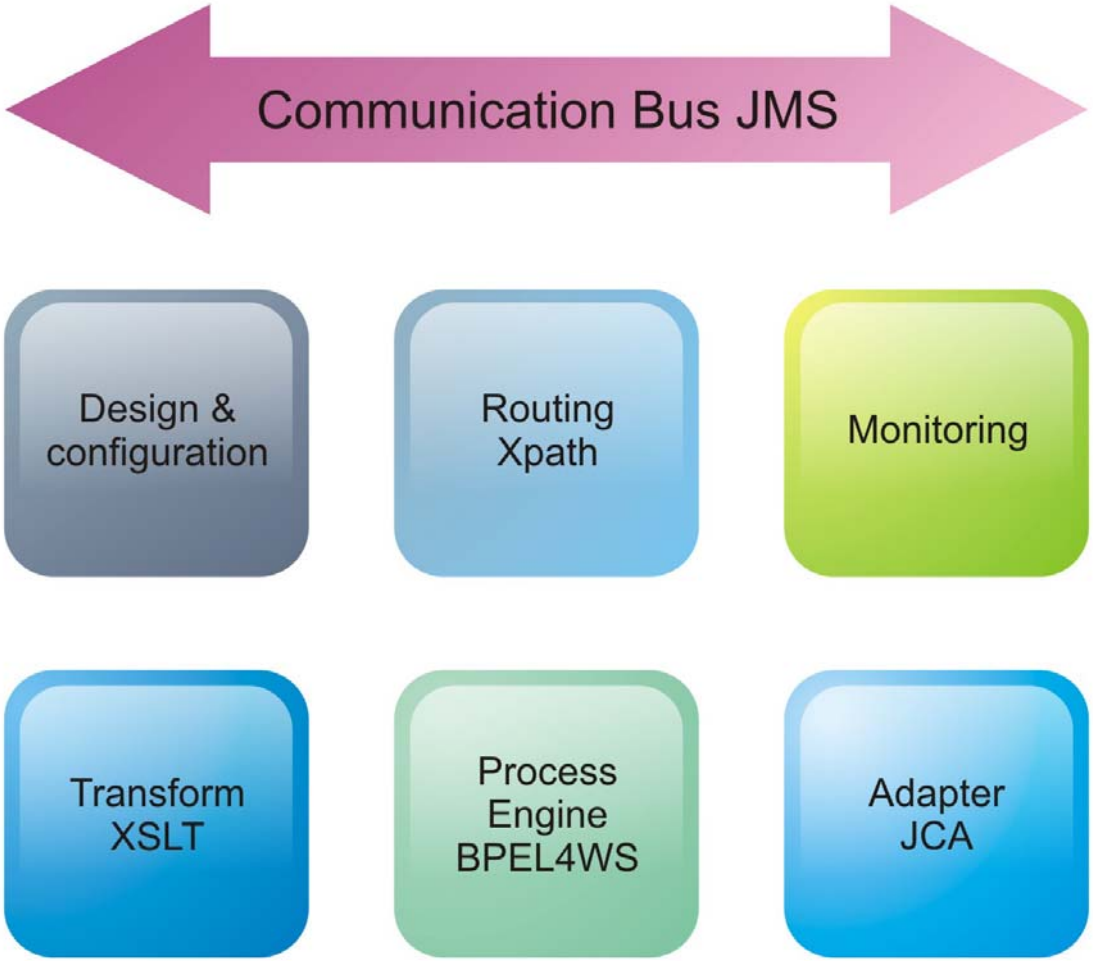
FLEXIBLE, EXTENSIBLE INTEGRATION



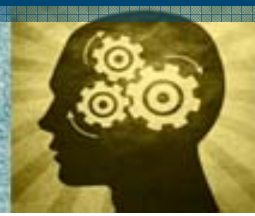
- Design methodology for distributed systems
- Applications expose functionality through service interfaces
- Loosely-Coupled
- Platform and language neutral
- Impervious to implementation changes
- Coarse-Grained- business level Interfaces
- Asynchronous
- No single points of failure

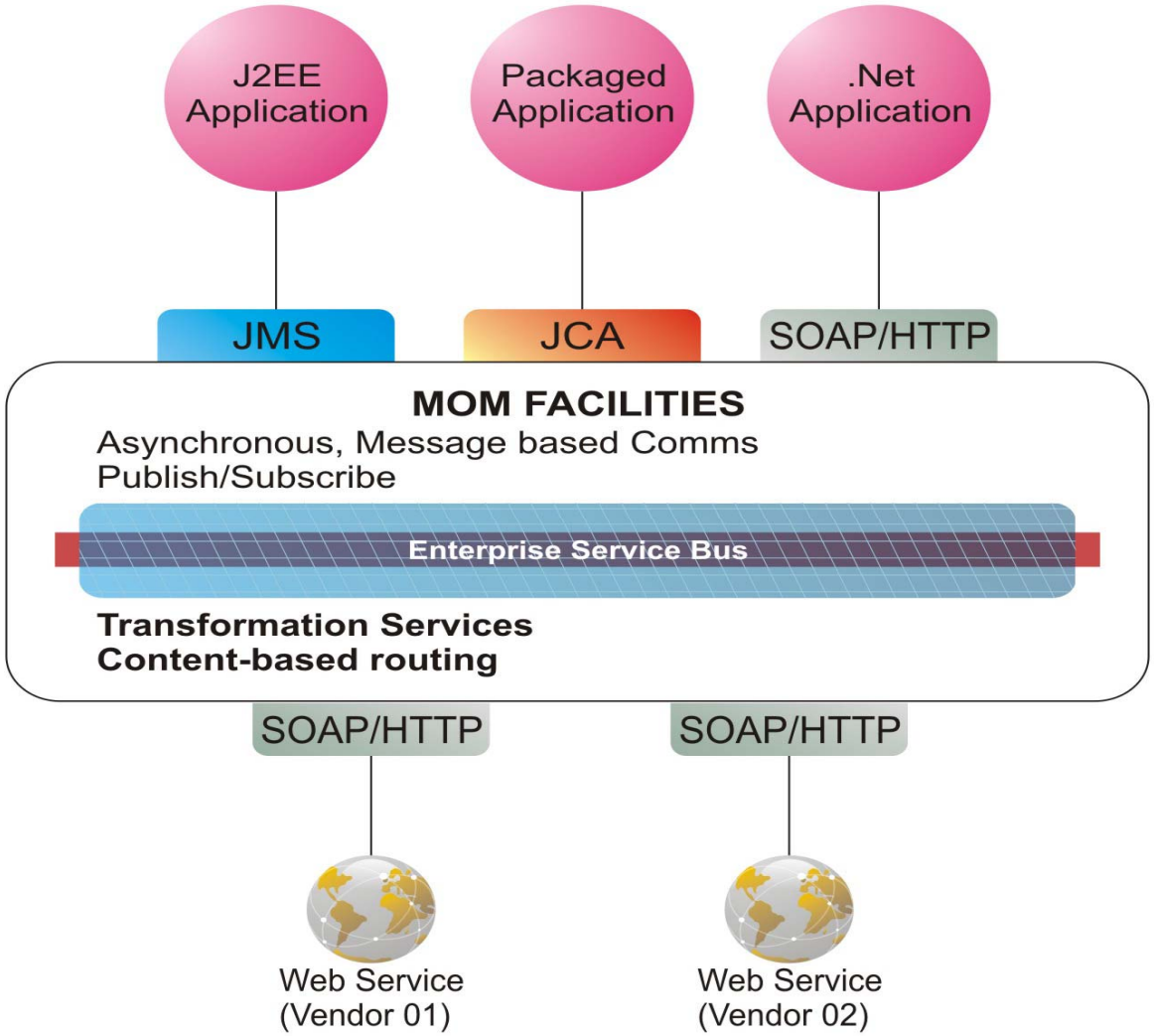
DEMYSTIFYING ESB



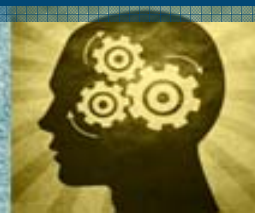


DEMYSTIFYING ESB



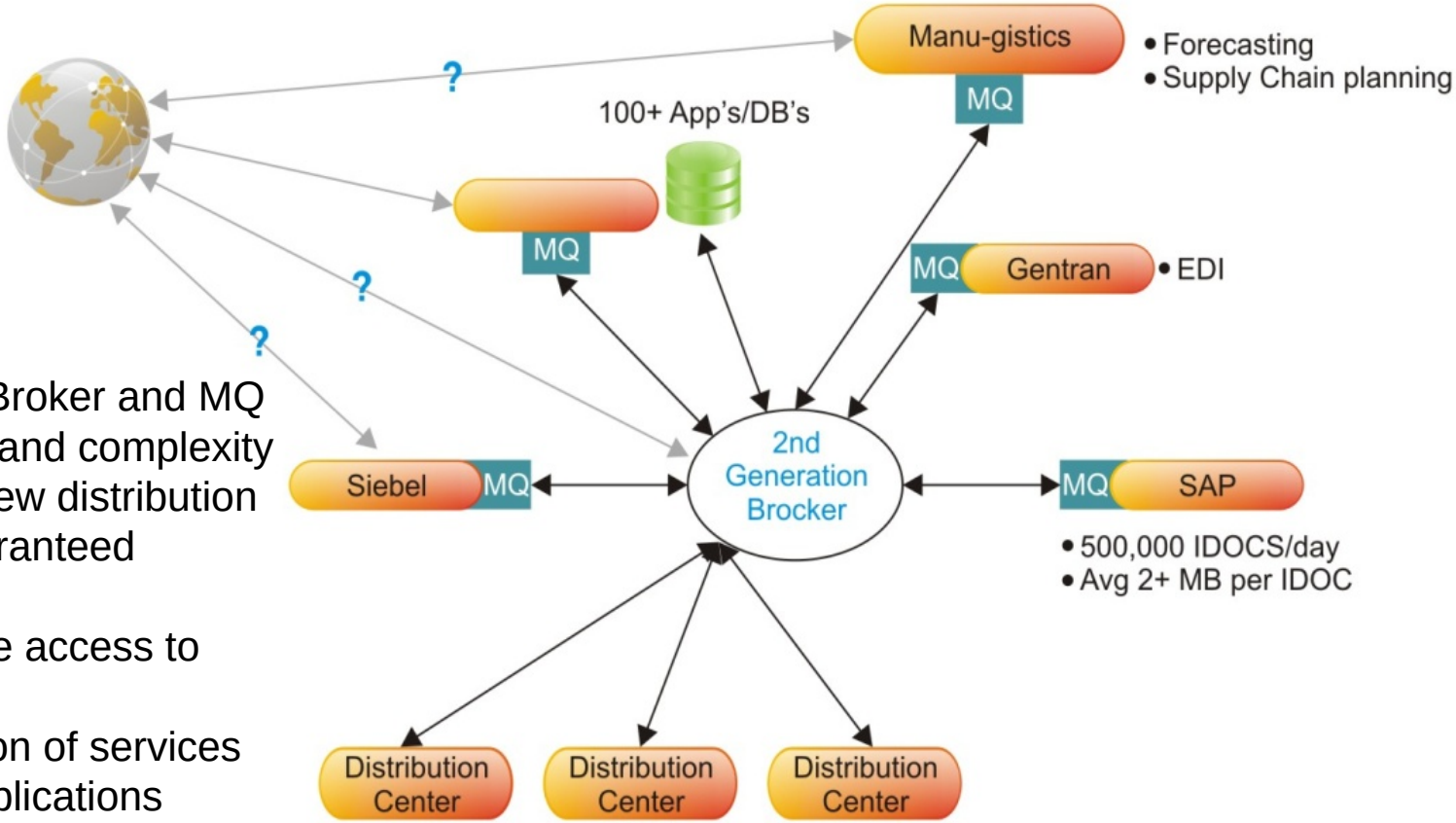


EXAMPLE: BUILD ON EXISTING INFRASTRUCTURE



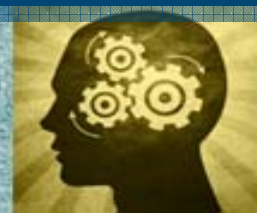
Business Needs

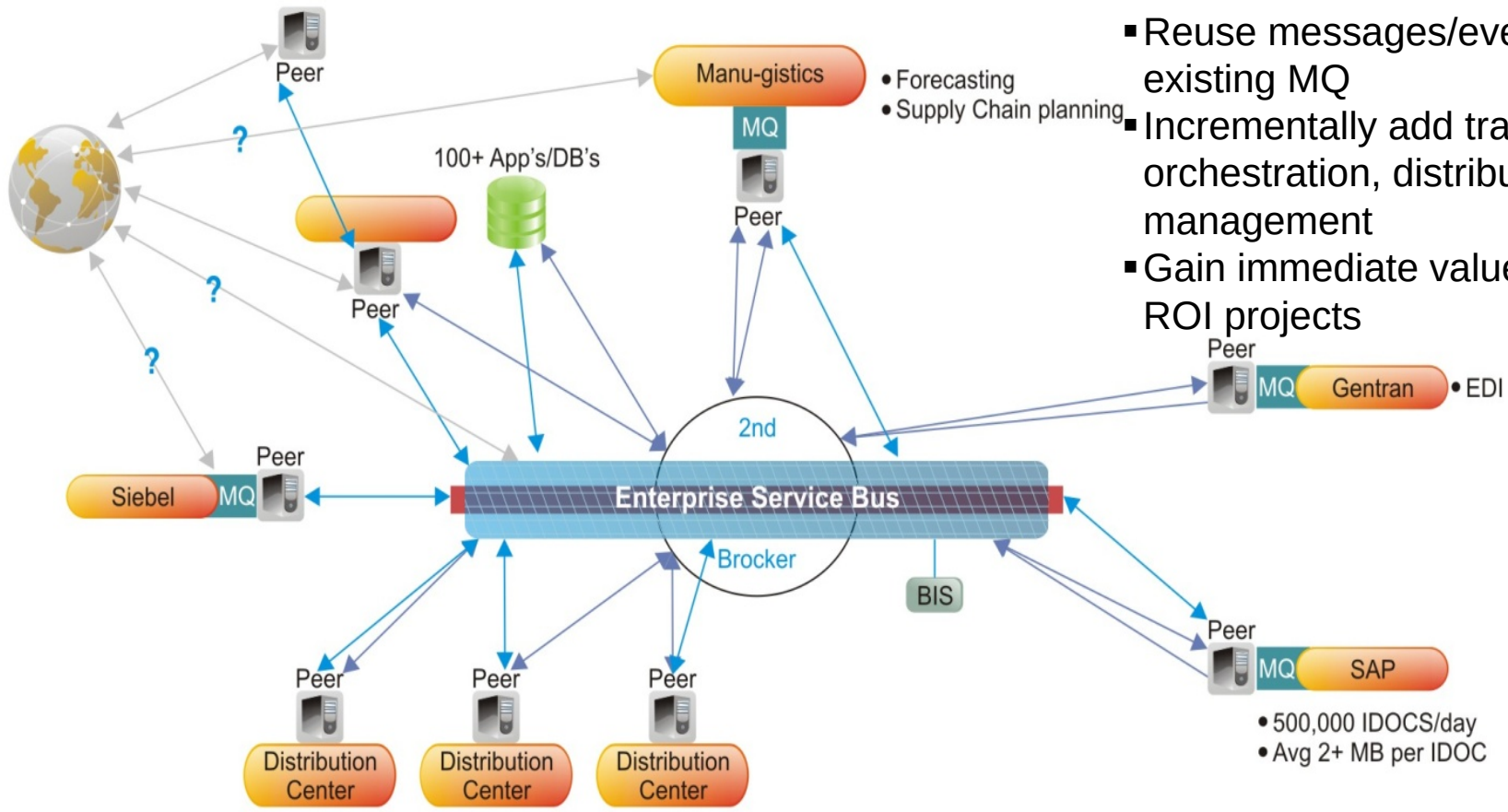
- Want to replace Broker and MQ due to high TCO and complexity
- Reduce cost of new distribution centers, with guaranteed messaging
- Need web-service access to data
- Need orchestration of services
- Integrate new applications faster and cheaper



Fiorano
Peer-to-Peer Dataflow Pipelines™

EXAMPLE: BUILD ON EXISTING INFRASTRUCTURE



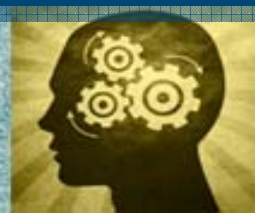


Solution

- Services Oriented Integration
- Reuse messages/events with existing MQ
- Incrementally add transformation, orchestration, distributed management
- Gain immediate value from higher ROI projects

Fiorano
Peer-to-Peer Dataflow Pipelines™

DEMYSTIFYING ESB

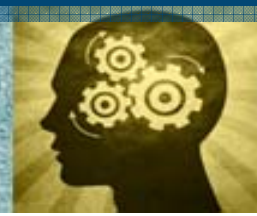


ESB's best suited for

- Projects that will mix heterogeneous application servers (for example, Microsoft or Java portals with disparate Java or Microsoft server back ends)
- Enterprises who want to start with a basic SOA and add other features later.
- Enterprises that want to assemble their own best-of-breed comprehensive integration suites
- Mix and match off-the-shelf adapters, BPM, B2B and BAM tools from other vendors.
- Distributed services (different programming languages) running on disparate nodes

Fiorano
Peer-to-Peer Dataflow Pipelines™

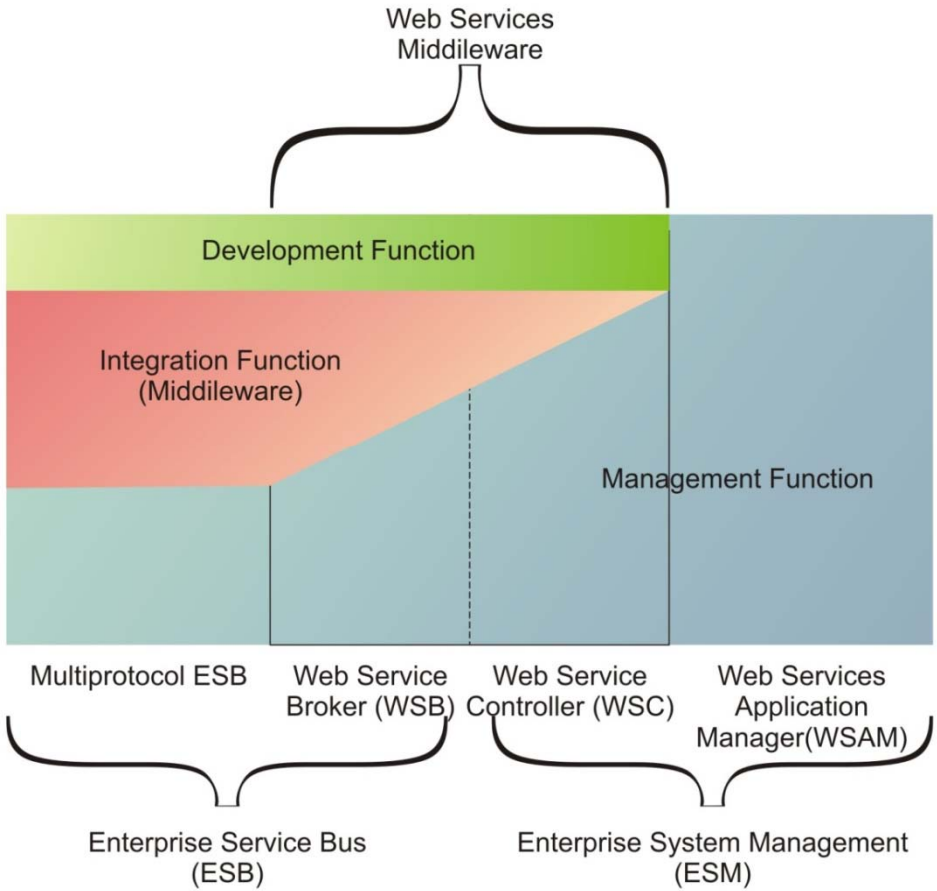
DEMYSTIFYING ESB



Types of ESB

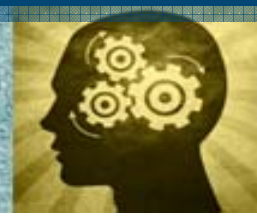
1. ESBs based solely on SOAP -Web services brokers (WSBs)

2. Multiprotocol ESBs that support Web services and other communication mechanisms



Source: Gartner, Who's who in middleware, 1Q04

ESB - VENDORS

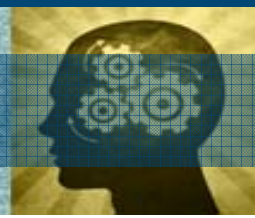


Support SOAP/HTTP and additional protocols guaranteed delivery, publish-and-subscribe often following the JMS standard

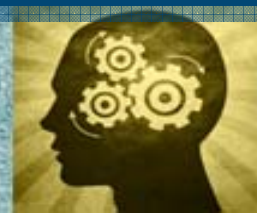
- IBM
- Oracle
- Microsoft
- Software AG
- Progress Software
- TIBCO

Fiorano
Peer-to-Peer Dataflow Pipelines™

IMPLEMENTATION METHODOLOGY OF ESB

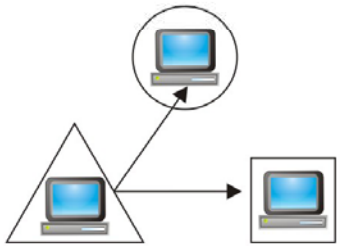


3 BASIC INTEGRATION PATTERNS



DB SYNCHRONIZATION

Data Consistency



- Multiple Processes
- Parallel unrelated steps
- One-Way, asynchronous
- Batch or Immediate
- Physically Independent
- Logically dependent

Source: Gartner Group

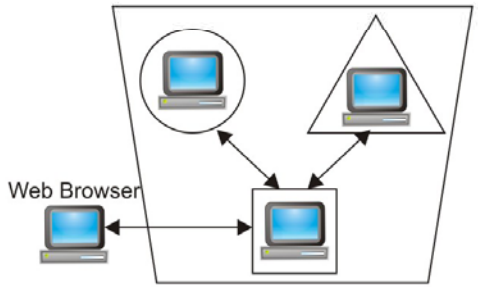
ORDER FULFILLMENT APPLICATION

Multistep Process

- One Business Process
- Multi Steps
- One-Way, asynchronous
- Batch or Immediate
- Physically Independent
- Logically dependent

QUOTE TO BIND APPLICATION

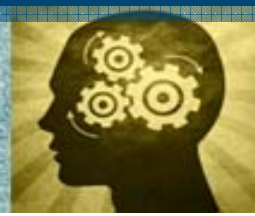
Composite Application

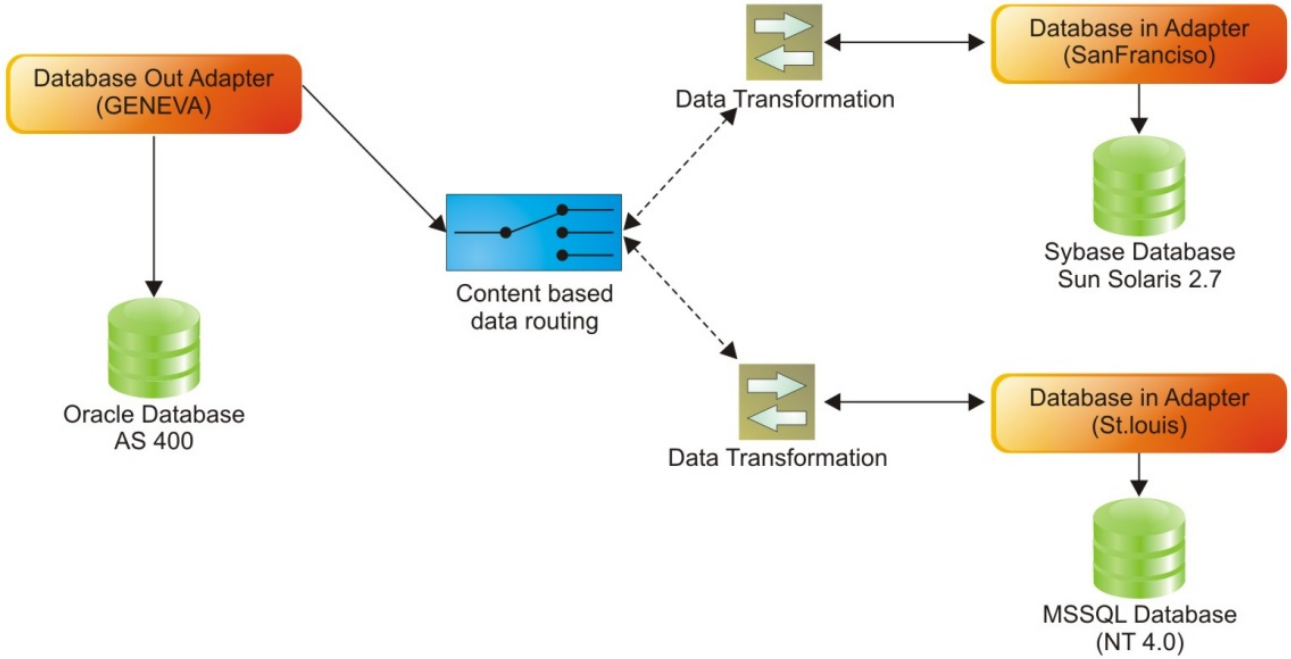


- One Business Process
- One Step
- Two-Way, synchronous
- Physically Independent
- Logically dependent

Fiorano
Peer-to-Peer Dataflow Pipelines™

STEP 1 – REQUIREMENTS GATHERING
DATA CONSISTENCY – DATABASE SYNCHRONIZATION



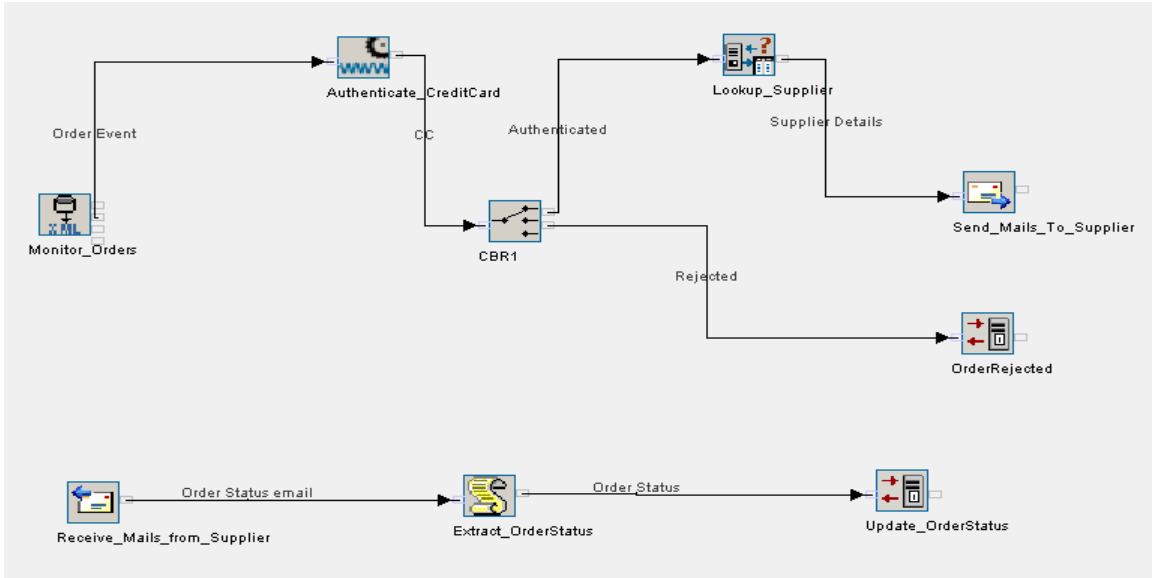


The technical requirements of ABC Corporation can be summarized as

- Data needs to be replicated across multiple data centers in asynchronous fashion using a Message Bus.
- Any changes made to the Oracle Database instance in Geneva needs to be reflected (based on a selection criteria) in either the Sybase database instance or the MSSQL database instance located in San Francisco and St.Louis respectively.
- The data needs to be transformed from the source table data format to target table data format(s).
- All data transfers need to be secure

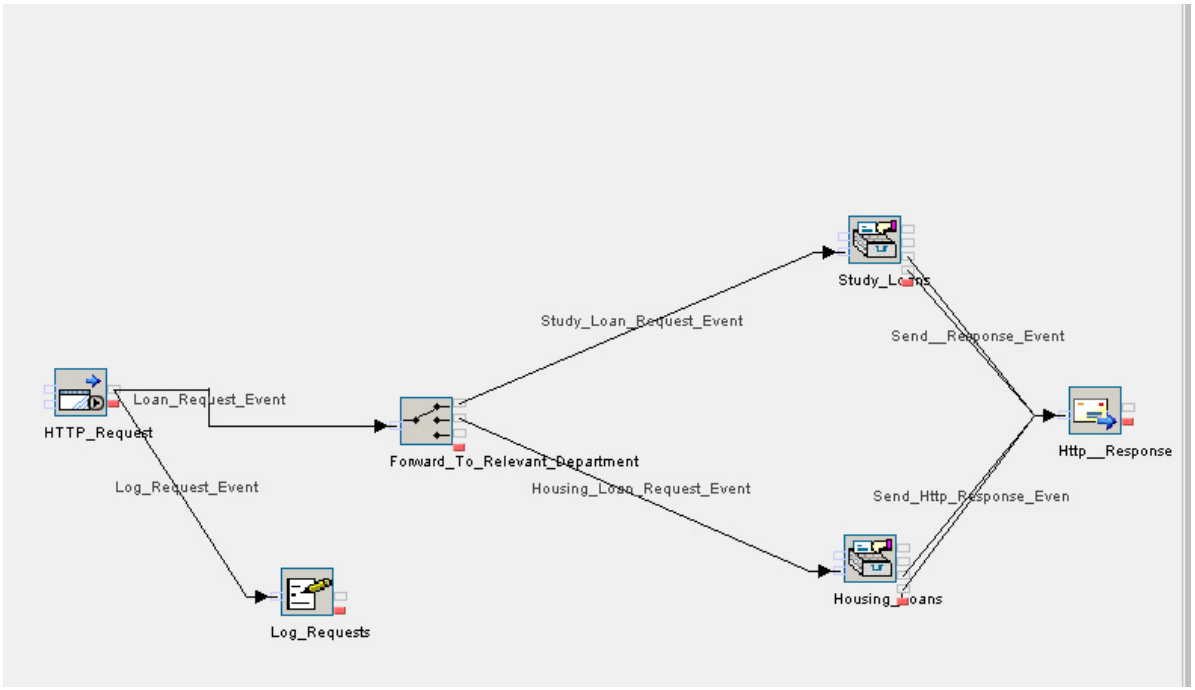
STEP 1 – REQUIREMENTS GATHERING

MULTI STEP PROCESS – ORDER FULFILLMENT



Company X runs an online market place for electronics product. Orders are accepted over web and are saved in Oracle DB. For processing an order, the company needs to perform credit-card verification using a 3'rd party hosted credit card gateway and then send out orders to suppliers over email. The supplier sends back acceptance or rejection of the order (along with expected delivery schedule) by a return email. This information needs to be updated in the Oracle DB, so that customer can track the status of his order using online means.

STEP 1 – REQUIREMENTS GATHERING
COMPOSITE APPLICATION – QUOTE TO BIND

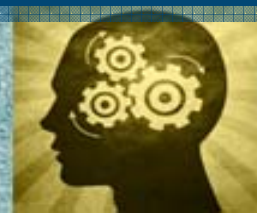


End to End business process for opening a new insurance policy that includes Capture of customer information, Evaluation of risk of requestor, Process the Application, Develop a Price Quote, Send Response (Quote)

Fiorano
Peer-to-Peer Dataflow Pipelines™

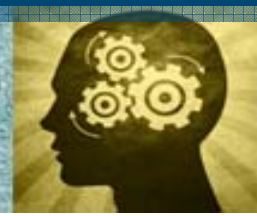
STEP 1 – REQUIREMENTS GATHERING

BUSINESS – DATA SYNCHRONIZATION



- Data synchronization for Disaster recovery planning and business continuance, Content distribution, Backup consolidation, Server migration.
- Completely eliminate all manual data synchronization without having to stop the system to accommodate new outlets or changes to the tables in the source or target systems
- Business process monitoring Dash Board
- Add new retail outlets
- Business users should be able to orchestrate business processes
- Timelines – ASAP
- Reduce development, maintenance cost

STEP 1 – REQUIREMENTS GATHERING
TECHNICAL/SYSTEMS – DATA SYNCHRONIZATION

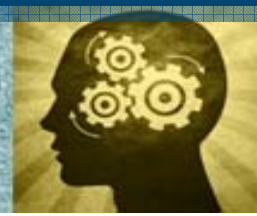


- TECHNICAL
  - High Level Business process representation – see figure shown earlier
  - Databases (SQL, Oracle, DB2 and files) across disparate OS.
  - Data flow representation
  - Intended users –
    - Business Analyst (Orchestration and Execution)
    - System Administrator (setup, configuration, data transformation)

- SYSTEM
  - Platforms and Operating Systems
  - Overall network topology

STEP 1 – REQUIREMENTS GATHERING

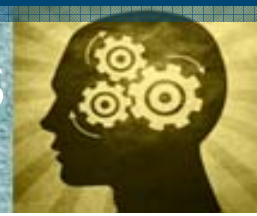
OUTPUT DOCUMENTS



- HL Design documents that addresses the integration requirements (business & technical)
- Document all the identified data and process flows.
- List of different networks, and the servers available in each of this network
- Recommended hardware specifications for the specific ESB product
- Any 3rd party software that is essential to the functioning of ESB

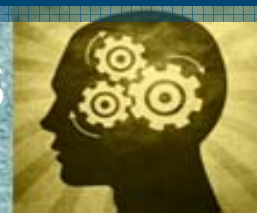
STEP 2 – INDENTIFYING “MUST HAVE” FEATURES

DATA CONSISTENCY



- Architecture
  - SOA ? (Hub & Spoke, Peer to Peer, Brokered Peer to Peer)
    - ✓ Brokered P2P with store and forward at end points of the network
    - ✓ Automatic reconnects
- Supported Standards
    - ✓ XML, JCA, CBR
- Message Bus (Asynchronous, Transformation etc)
    - ✓ Asynchronous, Transformation (at source or destination)
- Support for Distributed Applications (Compose, Execute and Monitor distributed Apps)
    - ✓ Location and Technology transparency, Intelligent routing, Single point of control, Deployment support
- Connectivity services (Web services,J2EE Connectors, JMS, WebSphere MQ)
    - ✓ JMS, Database Connectivity (JDBC)

STEP 2 – INDENTIFYING “MUST HAVE” FEATURES
DATA CONSISTENCY

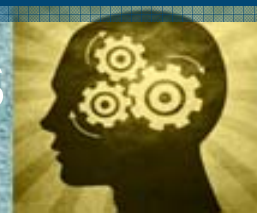


- Administration / Deployment
  - ✓ Single point of control
  - ✓ Dynamic changes
  - ✓ Single point of control
  - ✓ Remote access capability
  - ✓ Start / stop facilities
  - ✓ Manual routing support
  - ✓ Tracing
  - ✓ Message editing
- Monitoring
  - ✓ Problem determination
  - ✓ Problem prediction
  - ✓ Internal and external support
  - ✓ Support for enterprise management frameworks

Fiorano
Peer-to-Peer Dataflow Pipelines™

STEP 2 – INDENTIFYING “MUST HAVE” FEATURES

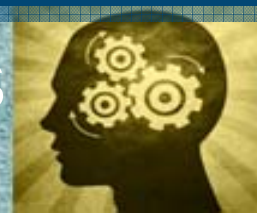
DATA CONSISTENCY



- Robustness (Service and the Infrastructure level)
  - ✓ Fault avoidance
  - ✓ Fault tolerance
- Scalability and Performance (Service and the Infrastructure level)
  - ✓ Asynchronous messaging
  - ✓ Multi-threading
  - ✓ Load balancing
  - ✓ Large data handling
- Security (Infrastructure and Application level)
  - ✓ Access control
  - ✓ Information security
  - ✓ Tools usage

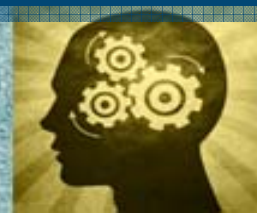
STEP 2 – INDENTIFYING “MUST HAVE” FEATURES

DATA CONSISTENCY



- Breadth of Connectivity (Configure, Modify, Connect with other Adapters)
  - ✓ DBMS access (Oracle, SQL, DB2 etc)
  - ✓ Legacy systems (Mainframe Applications)
  - ✓ Application servers
  - ✓ .NET
  - ✓ COM / CORBA - Multi-Language Adapters
  - ✓ WebServices (Publisher and Consumer)
- Tools
  - ✓ Configuration (Business Processes, Services, Infrastructure)
  - ✓ Incremental deployment
  - ✓ Life cycle support

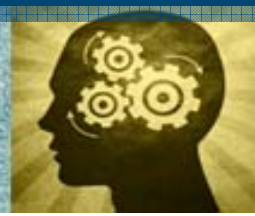
STEP 3 – COMPONENTIZING BUSINESS PROCESSES



- Component/Service/Business Service – definition
- Fain Grained Components - Issues
- Coarse Grained Components – Advantages
- ESBs and Components
- Pre-built Components
- User Defined Components – Best practices
- Reusing existing resources

STEP 3 – COMPONENTIZING BUSINESS PROCESSES

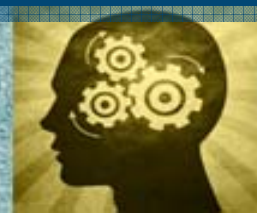
FINE GRAINED COMPONENTS - ISSUES



- Typically represent function call (static input and outputs)
- Low reusability (impossible to build a general purpose Database Adapter)
- Development overheads (tight coupling between client and component)
- Change of service functionality needs re-coding (Static data formats for input and output)
- Does not lend well to business process composition
- Synchronous invocation of function calls
- Skilled developers needed to use these components

STEP 3 – COMPONENTIZING BUSINESS PROCESSES

COARSE GRAINED COMPONENTS - ADVANTAGES



- Represents a high level business component
- Orchestrate business processes using these components
- Reusable across business process (changes only in the design time properties)
- Dynamic data formats for input and output
- Synchronous & Asynchronous invocation
- Low development cost (middleware is hidden)
- Business Analyst can orchestrate business processes

STEP 3 – COMPONENTIZING BUSINESS PROCESSES

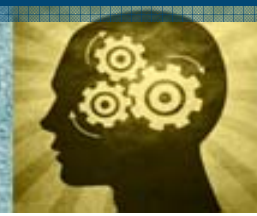
ENTERPRISE SERVICES - REUSABLE, BUSINESS LEVEL BUILDING BLOCKS

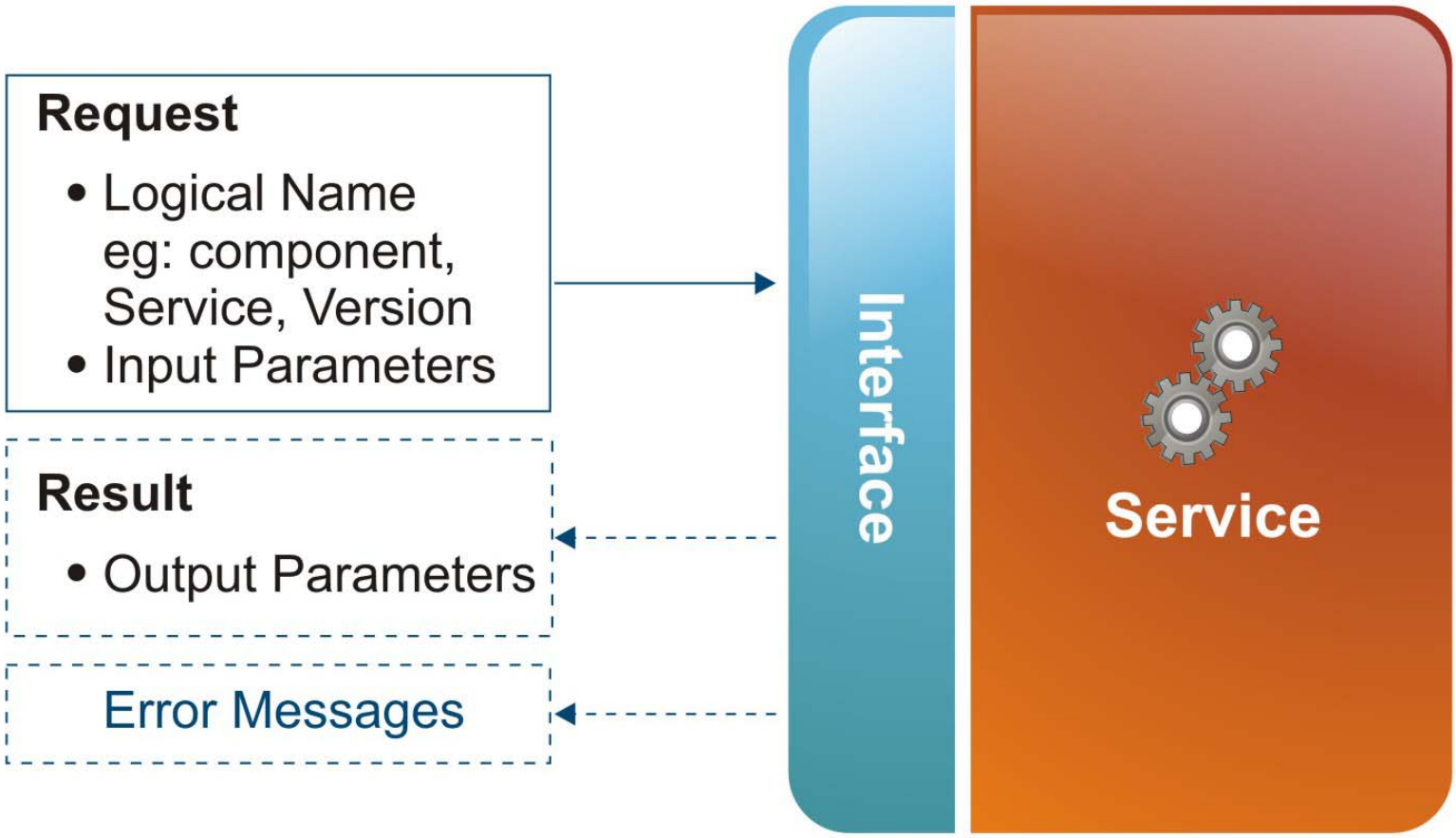
- Coarse-Grained
  - ✓ Level of abstraction easily understood by business people
- Event-Enabled Interfaces
  - ✓ Easily composed into Event Processes
- Multi-Language
  - ✓ No development restrictions
- Generalization of Web-Services
  - ✓ Standards-based WSDL interfaces

Fiorano
Peer-to-Peer Dataflow Pipelines™

STEP 3 – COMPONENTIZING BUSINESS PROCESSES

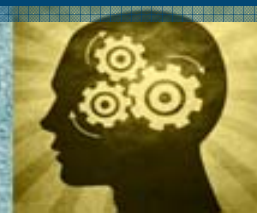
SERVICE → INTERFACE





STEP 3 – COMPONENTIZING BUSINESS PROCESSES

COARSE GRAINED COMPONENTS



“A component is a non-trivial, nearly independent, and replaceable part of a system that fulfills a clear function in the context of a well-defined architecture. A component conforms to and provides the physical realization of a set of interfaces."

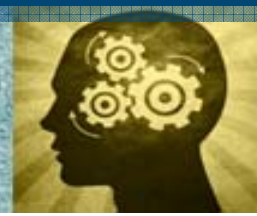
- Philippe Krutchen, Rational Software

- Well defined interfaces & contracts
- Provides implementation of interfaces
- Coarse grained functionality

Fiorano
Peer-to-Peer Dataflow Pipelines™

STEP 3 – COMPONENTIZING BUSINESS PROCESSES

COARSE GRAINED COMPONENTS



"A software component is a unit of composition with contextually specified interfaces and explicit context dependencies only. A software component can be deployed independently and is subject to third-party composition."

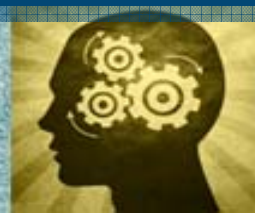
- Clemens Szyperski, Component Software

- Developed, tested and deployed in complete isolation
- Can be configured for different environment during deployment
- Supports contextual dependencies.

Fiorano
Peer-to-Peer Dataflow Pipelines™

STEP 3 – COMPONENTIZING BUSINESS PROCESSES

COARSE GRAINED COMPONENTS



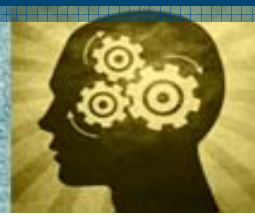
"A component is a physical and replaceable part of a system that conforms to and provides the realization of a set of interfaces...typically represents the physical packaging of otherwise logical elements, such as classes, interfaces, and collaborations."

- Grady Booch, Jim Rumbaugh, Ivar Jacobson

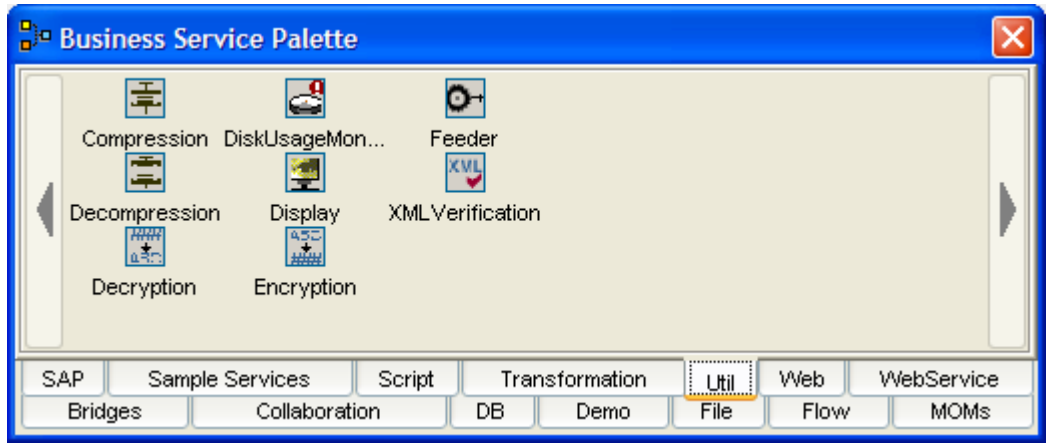
- Includes physical packaging of files, executables, jars, dlls etc.
- Aggregation of classes & functions into 1 complete functionality.
- Higher level of reusability than Classes

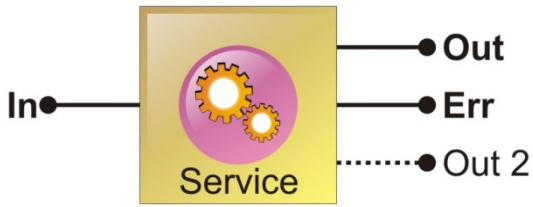
STEP 3 – COMPONENTIZING BUSINESS PROCESSES

PRE-BUILT SERVICES



- Services
  - ✓ Bridges
  - ✓ Database/File Adapters
  - ✓ Web/WebService Adapters
  - ✓ Routing (flow services)
  - ✓ Transformation
  - ✓ Signing, encryption, encoding Services
- Adapters
  - ✓ Middleware
  - ✓ Application

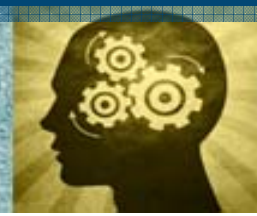




Fiorano
Peer-to-Peer Dataflow Pipelines™

STEP 3 – COMPONENTIZING BUSINESS PROCESSES

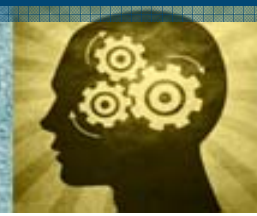
USER DEFINED – BEST PRACTICES



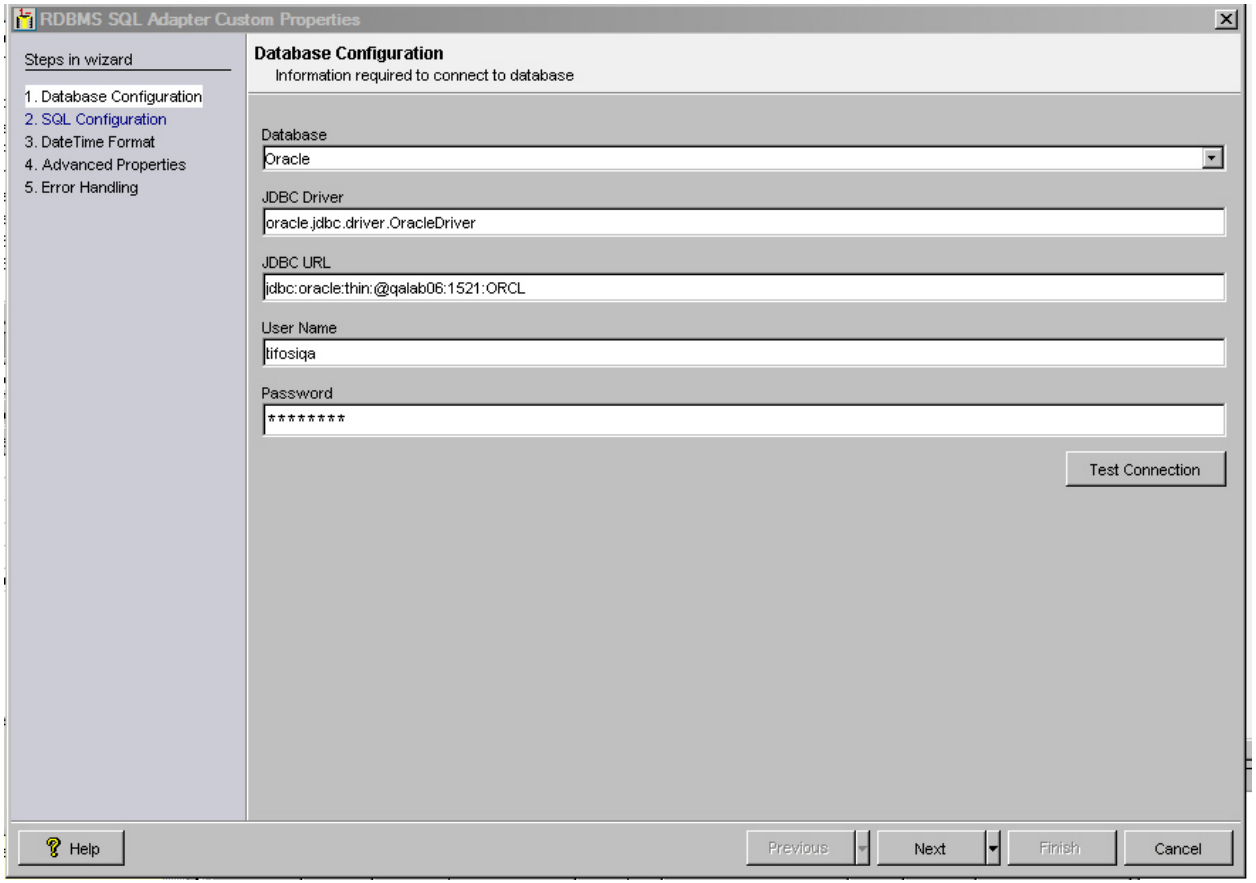
- Identifying set of user defined services
- Choosing Appropriate programming language
  (Java, C, C++, C#, VB, ActiveX, Perl, Python, JavaScript etc)
- Identifying input and output ports
  (dynamic or static)
- Custom Property Sheet
  (Design & runtime properties)
- Exception and Error Handling mechanisms
- Tracing and Logging support
  - Trace Modules and trace levels
  - Logging details
- Transaction support
  - Local
  - Global

STEP 3 – COMPONENTIZING BUSINESS PROCESSES

USER DEFINED – BEST PRACTICES

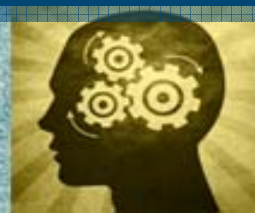


- Custom Property Sheet (Design & runtime properties)

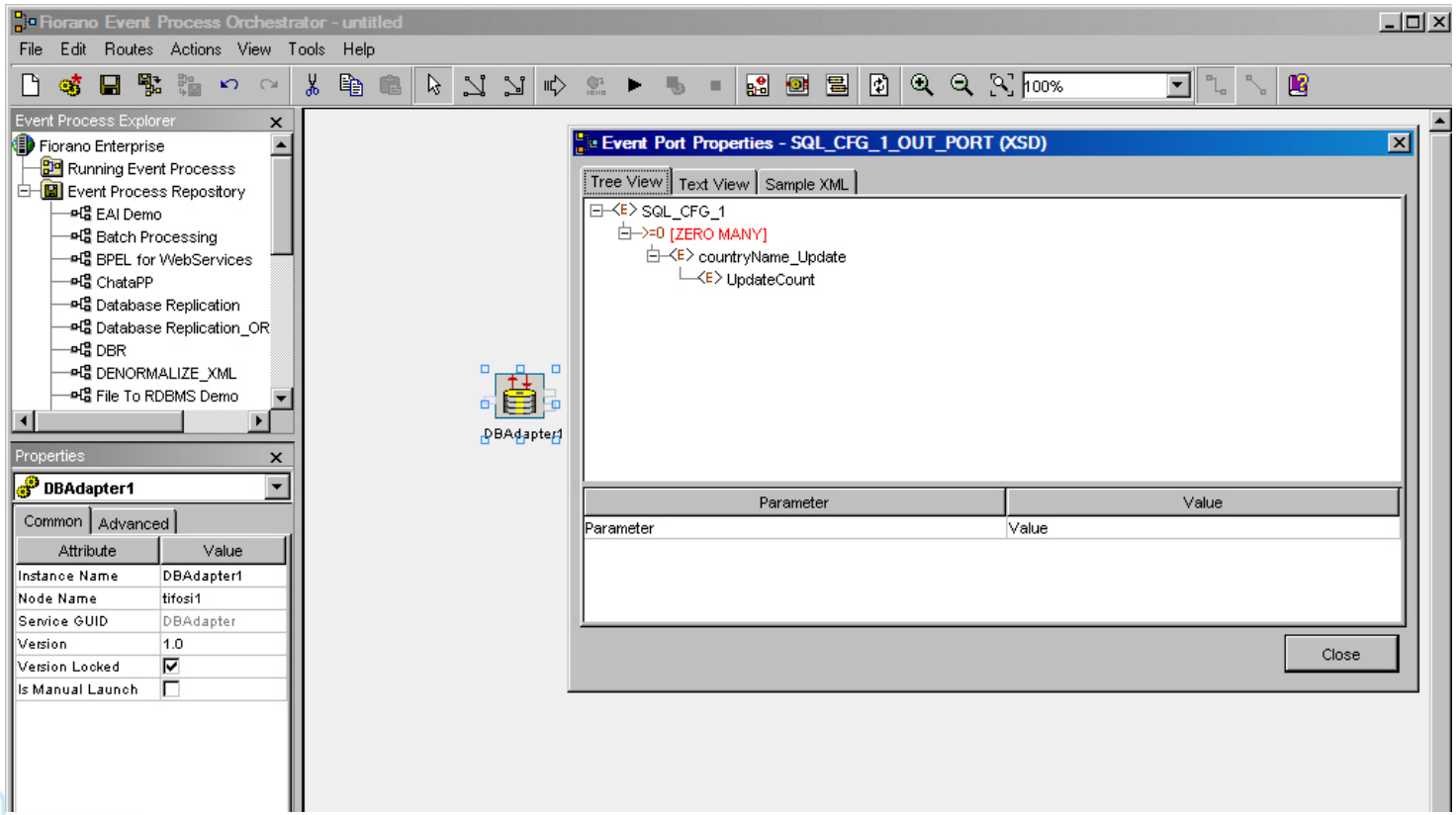


STEP 3 – COMPONENTIZING BUSINESS PROCESSES

USER DEFINED – BEST PRACTICES

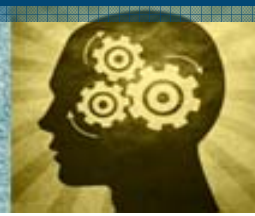


- Identifying input and output ports (dynamic or static)
- Data format definitions (XSD, DTD, XML)

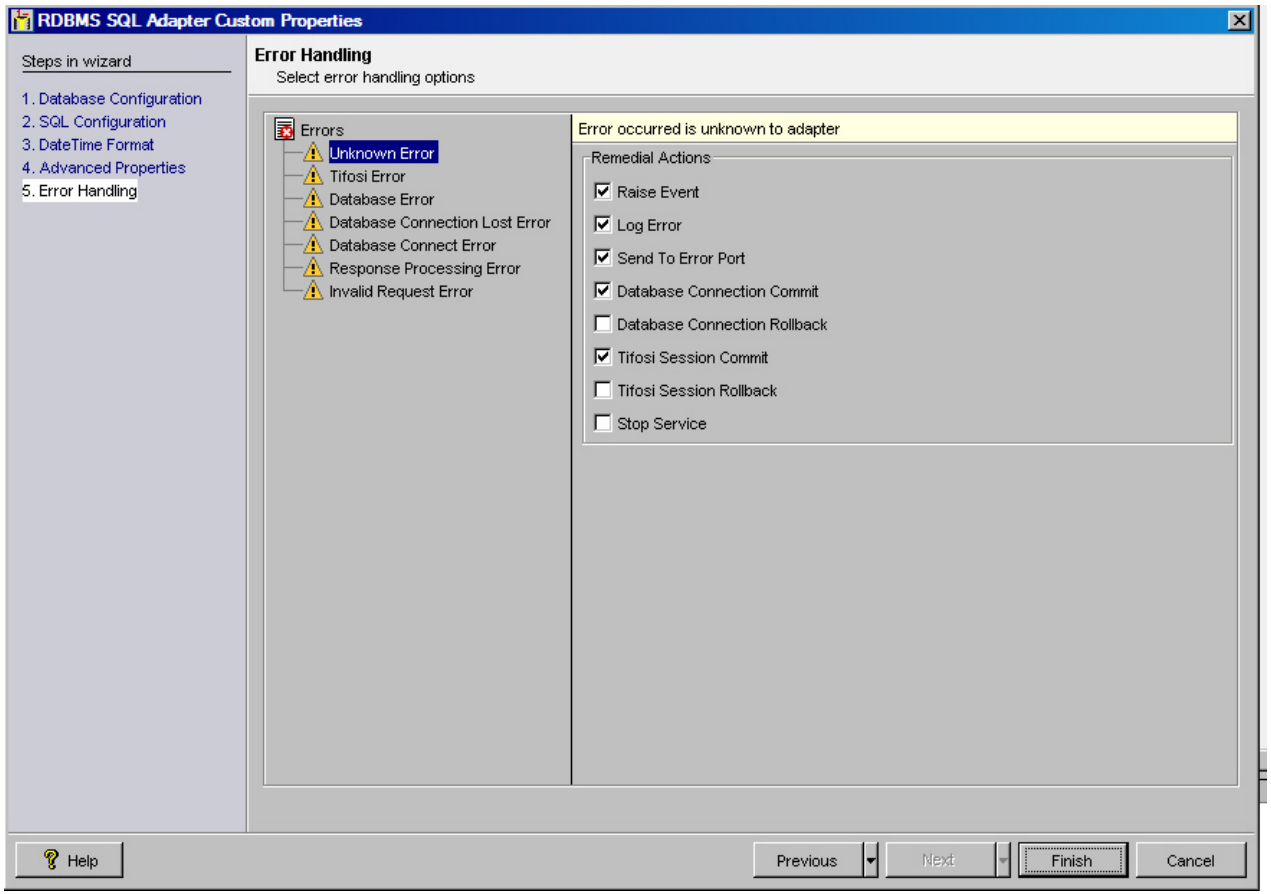


STEP 3 – COMPONENTIZING BUSINESS PROCESSES

USER DEFINED – BEST PRACTICES

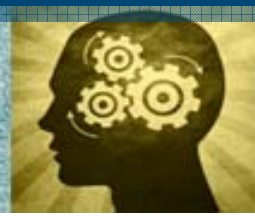


- Exception and Error Handling mechanisms

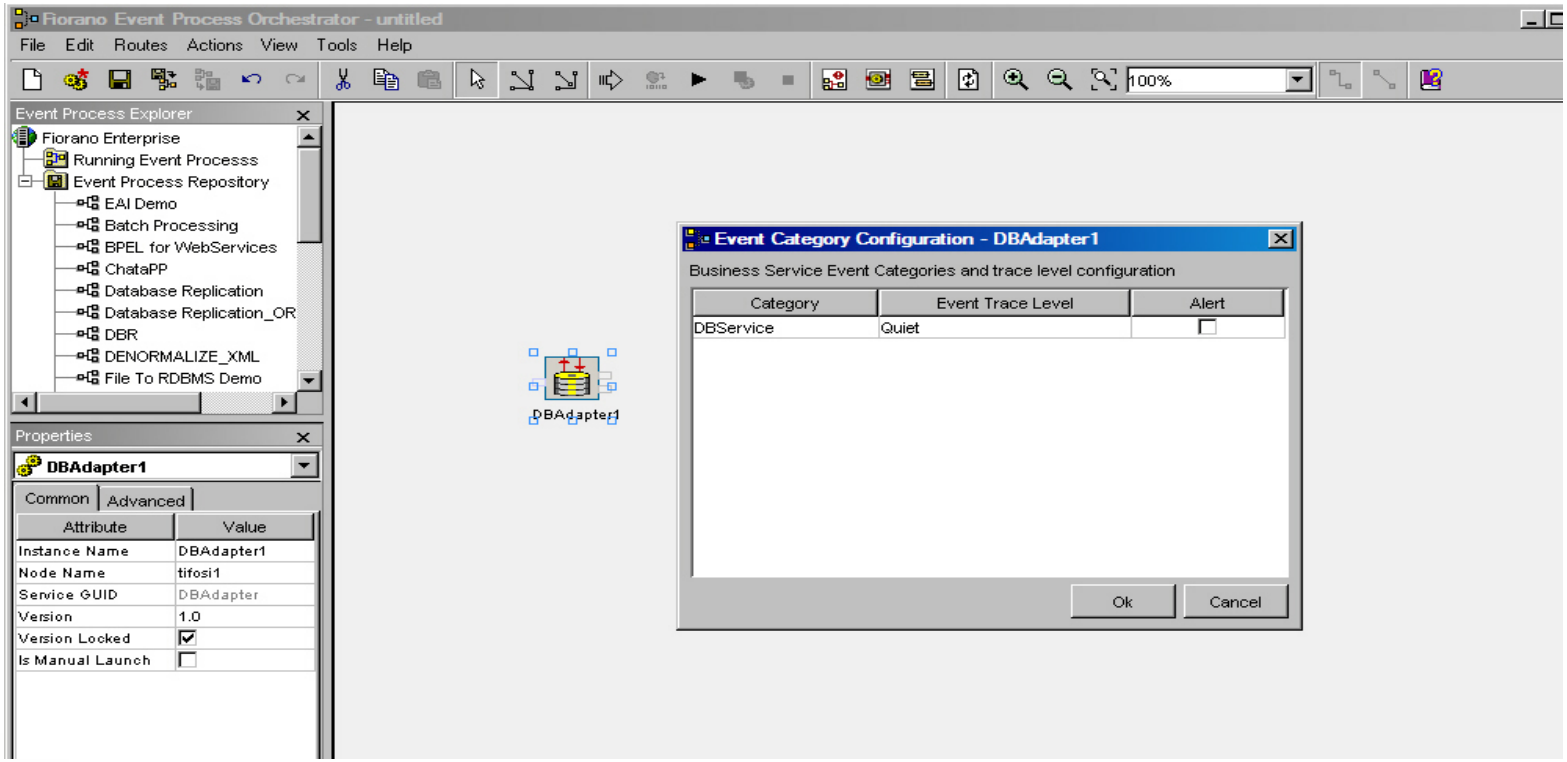


STEP 3 – COMPONENTIZING BUSINESS PROCESSES

USER DEFINED – BEST PRACTICES

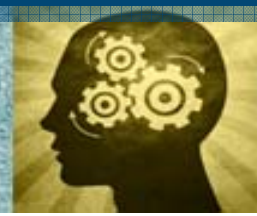


- Tracing and Logging support
  - Trace Modules and trace levels
  - Logging details



STEP 3 – COMPONENTIZING BUSINESS PROCESSES

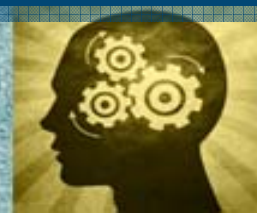
USER DEFINED – BEST PRACTICES



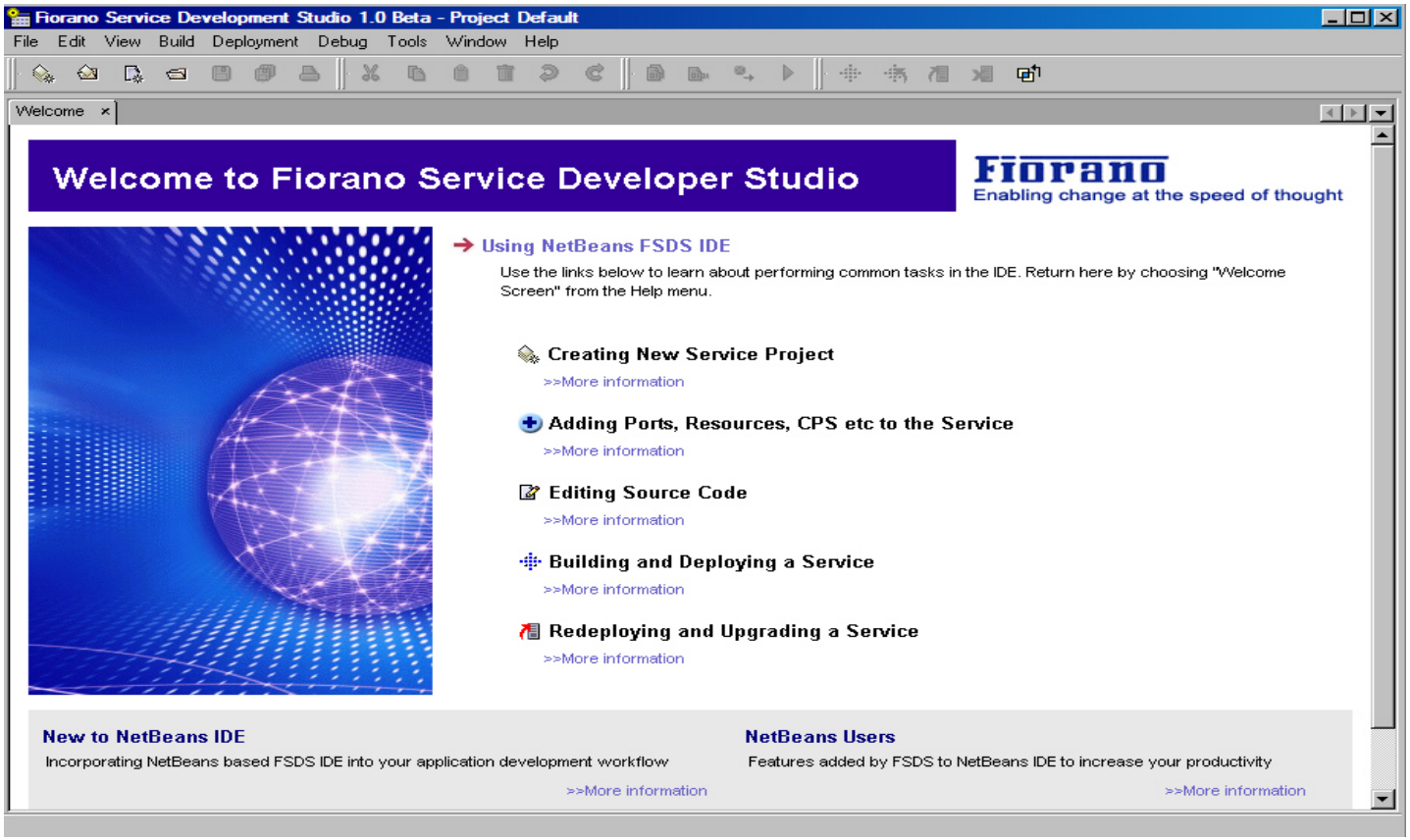
- Transaction support
  - ✓ Local (Client level transactions)
  - ✓ Global (XA)
  - ✓ Compensation transactions
- Performance Requirements
  - ✓ Expected transaction throughput
  - ✓ Multi-threading support
- Synchronous/Asynchronous

STEP 3 – COMPONENTIZING BUSINESS PROCESSES

USER DEFINED – BEST PRACTICES

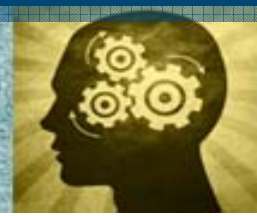


- Tools for service development
- IDE plug-ins



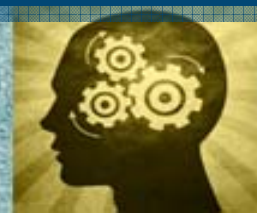
STEP 3 – COMPONENTIZING BUSINESS PROCESSES

REUSING EXISTING SERVICES – BEST PRACTICES



- Wrap existing code (Java, C, C++, VB etc)
- Adapters to COM, DCOM, EJB, CORBA, RMI etc
- Reusing business processes

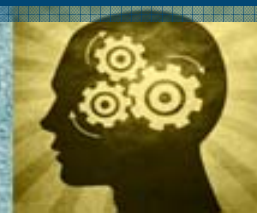
STEP 4 – DEFINING DISTRIBUTED ESB PROCESS



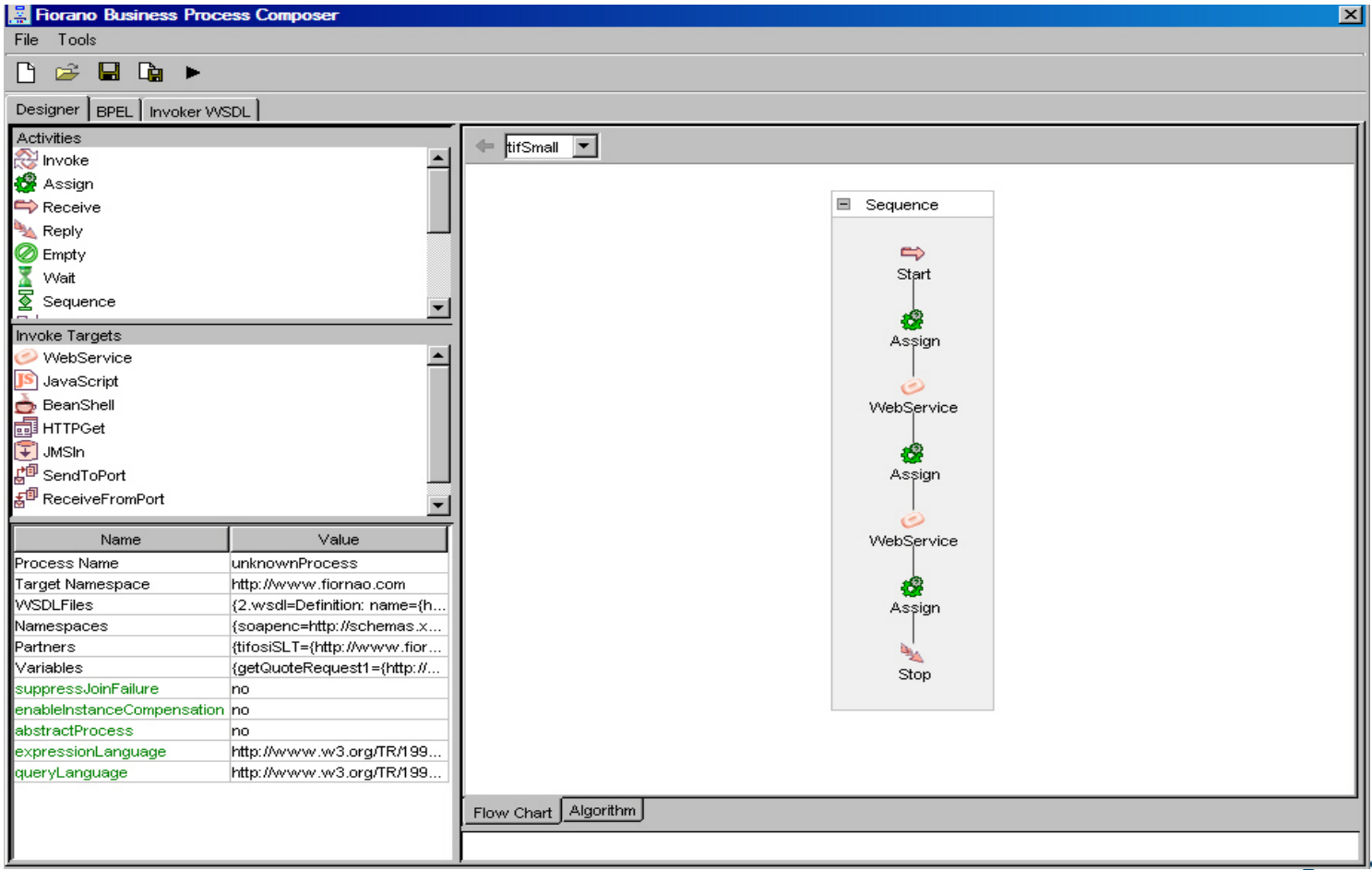
- Control vs. Data Flow
- Business Process Orchestration
- Event Warehousing
- Security
- Logging, Tracing & Alerts
- Data format impedance mismatch
- Configuring Business process for failover
- Configuring Business process for performance

STEP 4 – DEFINING DISTRIBUTED ESB PROCESS

CONTROL VS DATA FLOW

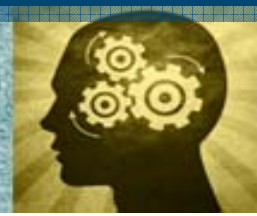


- Control Flow

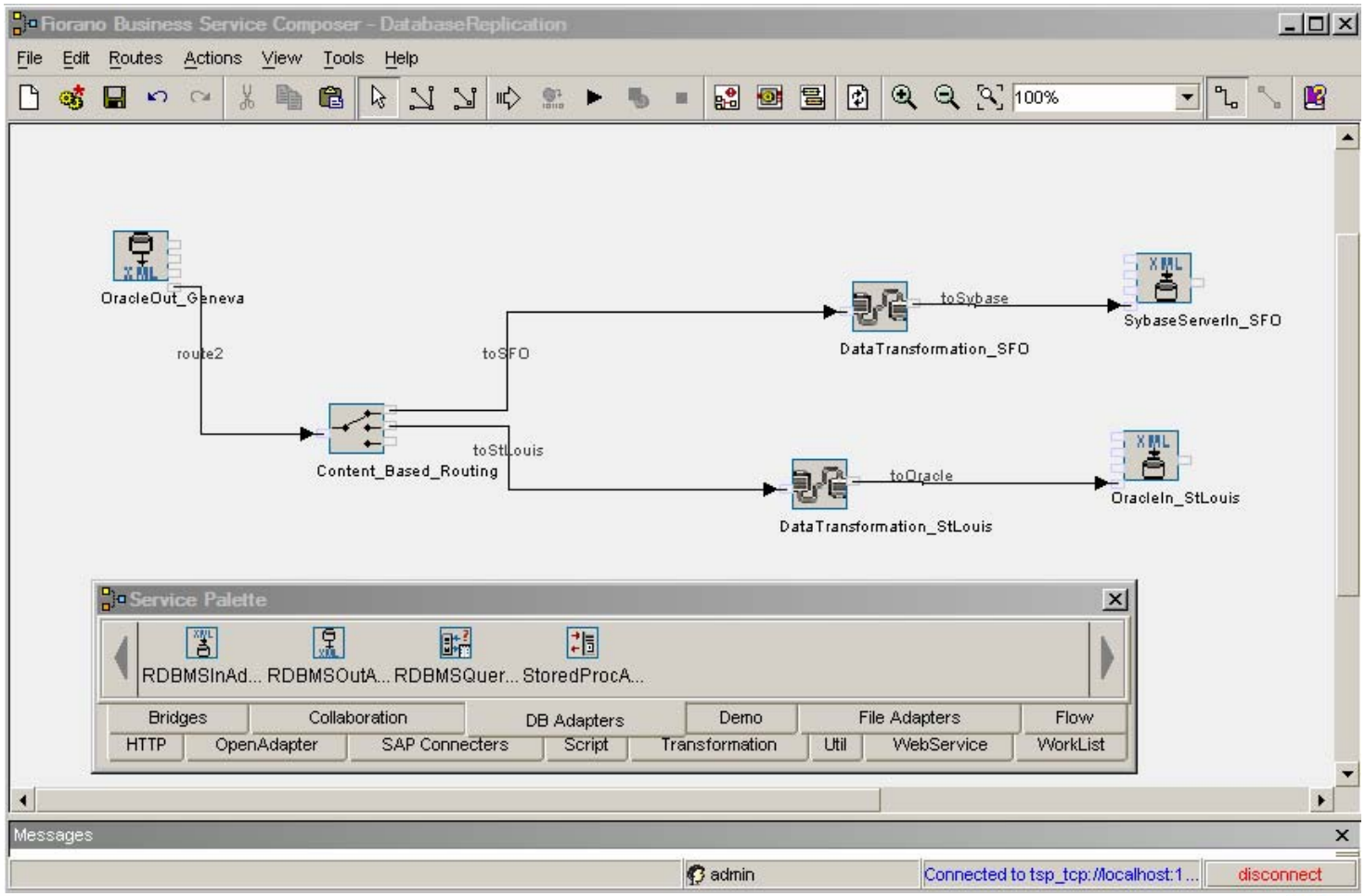


STEP 4 – DEFINING DISTRIBUTED ESB PROCESS

CONTROL VS DATA FLOW

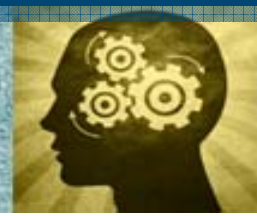


- Data Flow



STEP 4 – DEFINING DISTRIBUTED ESB PROCESS

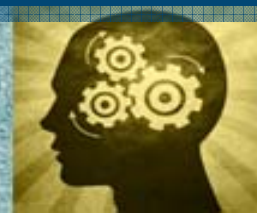
BUSINESS PROCESS ORCHESTRATION



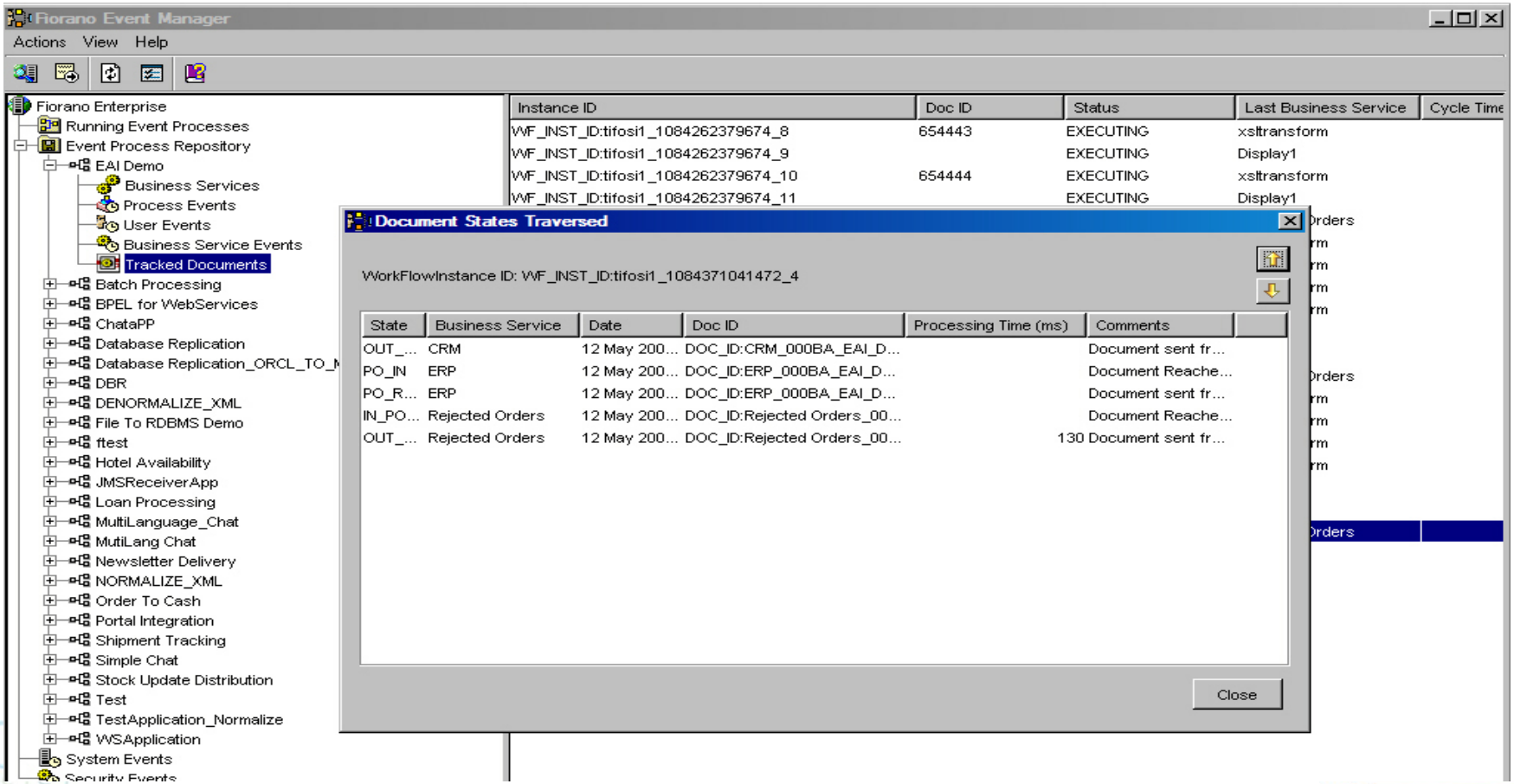
- Composition using pre-built services
- Data routing
- Control information
- Data transformation
- Identifying Node Names
- Configuring Service Design time properties

Fiorano
Peer-to-Peer Dataflow Pipelines™

STEP 4 – DEFINING DISTRIBUTED ESB PROCESS
EVENT WAREHOUSING

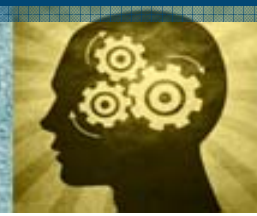


- Enable at runtime

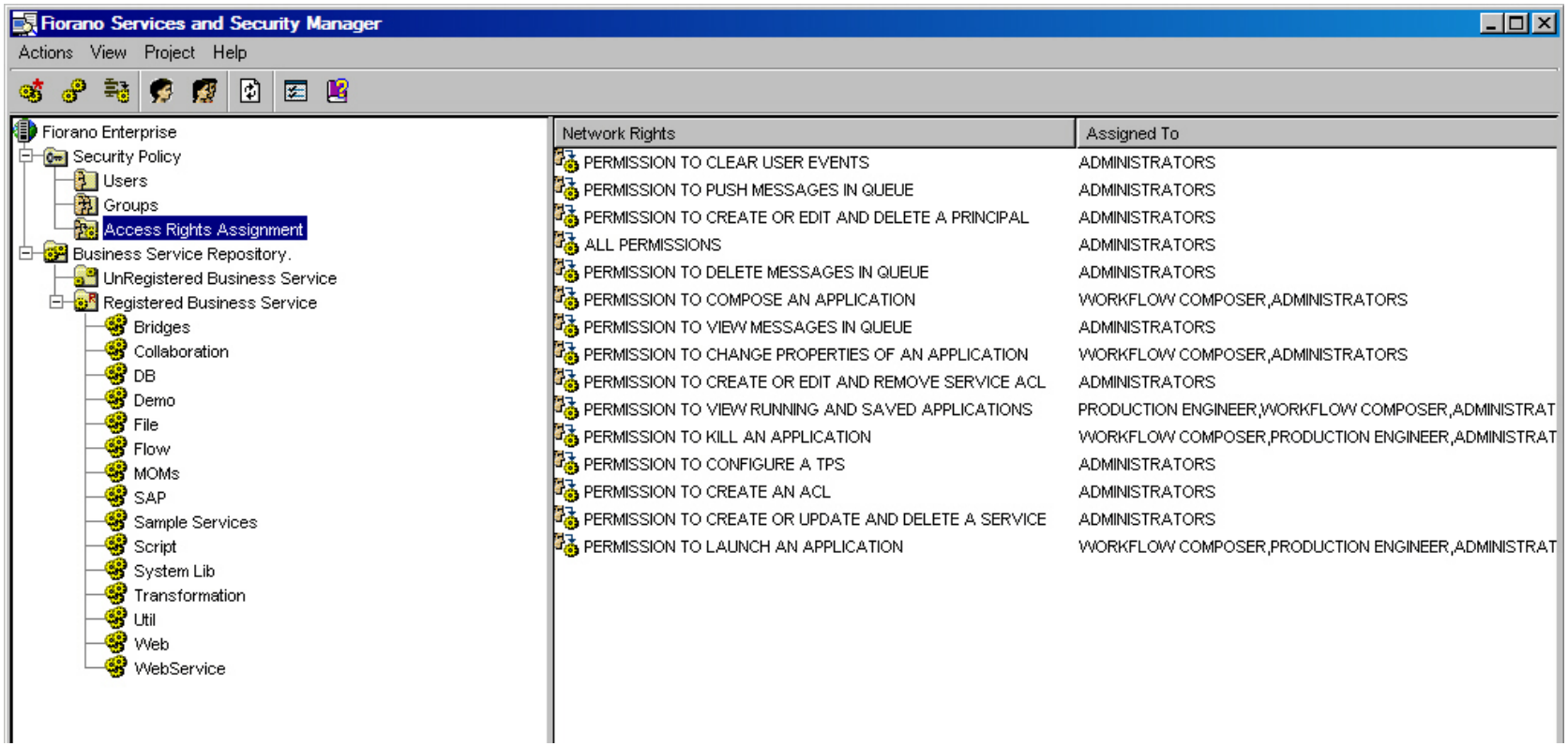


Fiorano
Peer-to-Peer Dataflow Pipelines™

STEP 4 – DEFINING DISTRIBUTED ESB PROCESS
SECURITY

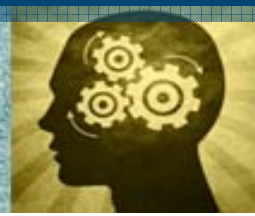


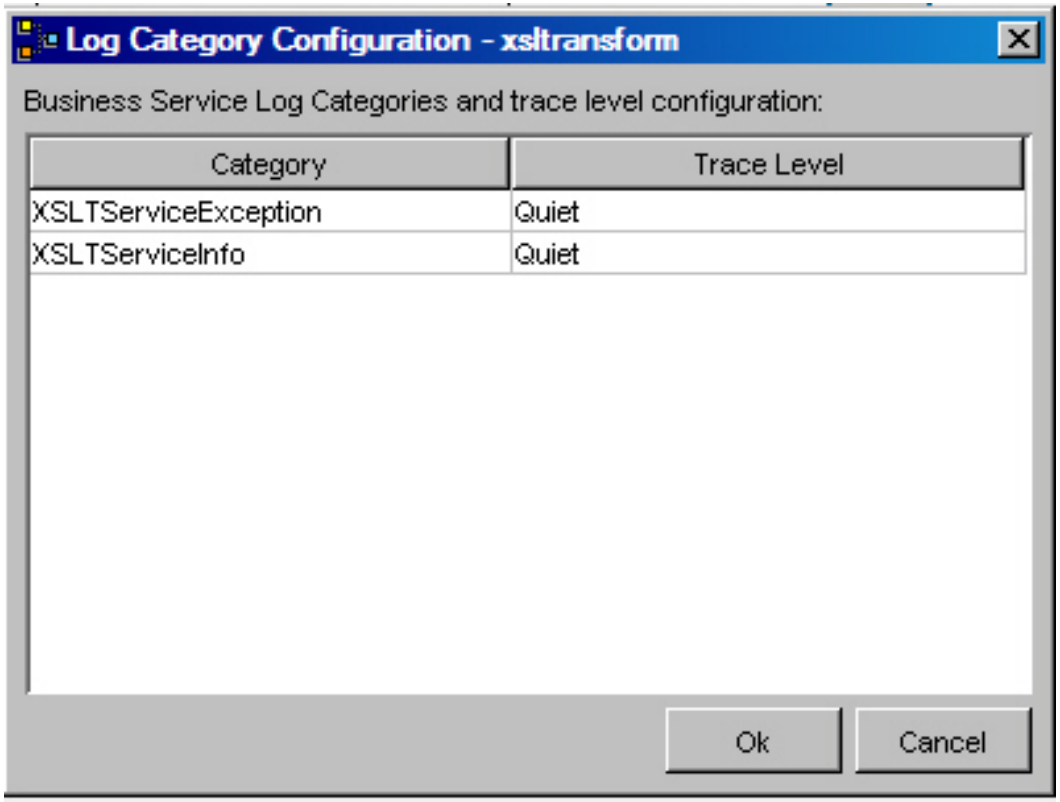
Security (Role based Security)



STEP 4 – DEFINING DISTRIBUTED ESB PROCESS

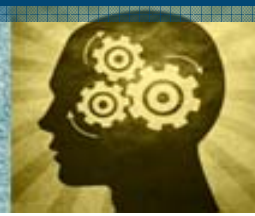
LOGGING, TRACING & ALERTS

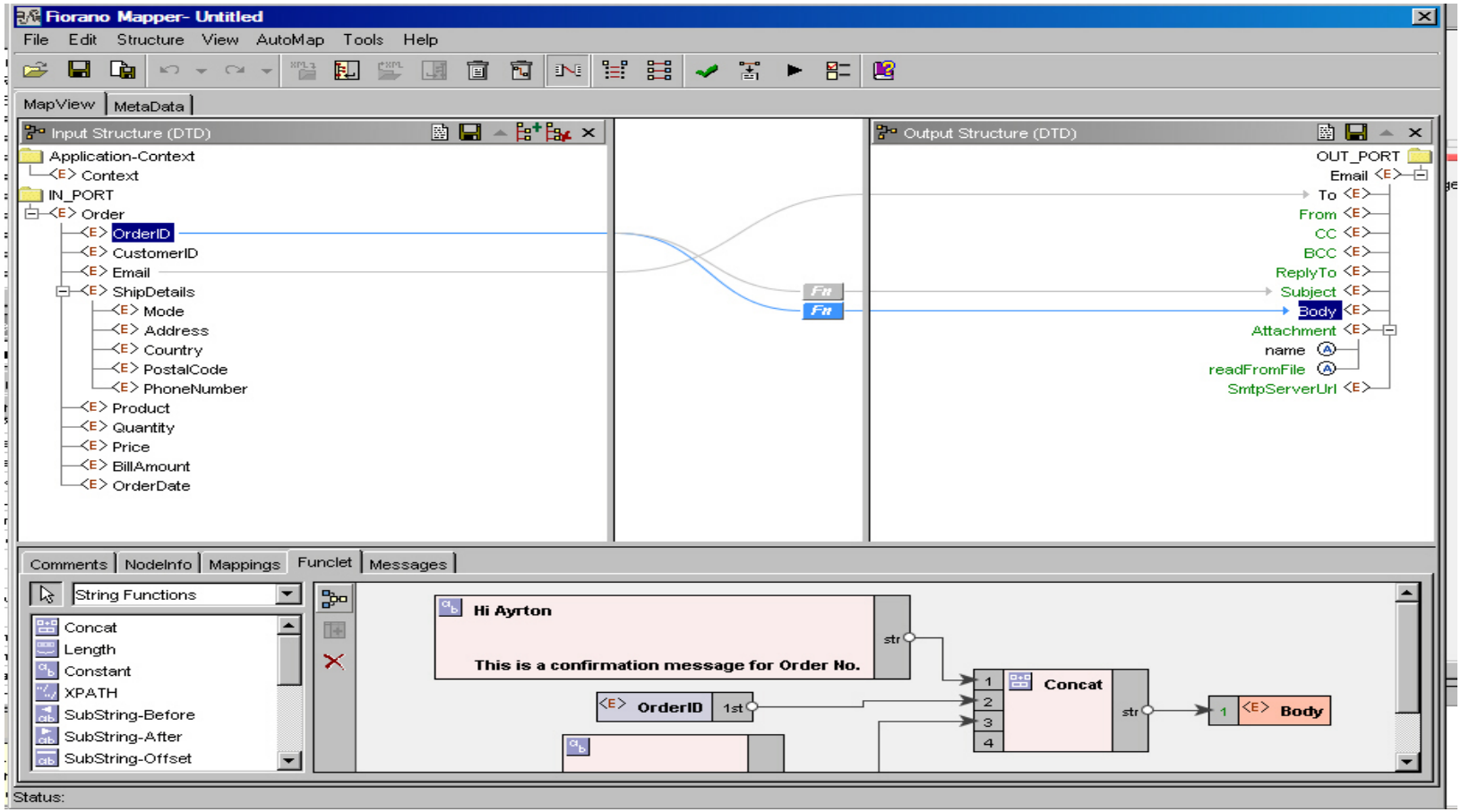




STEP 4 – DEFINING DISTRIBUTED ESB PROCESS

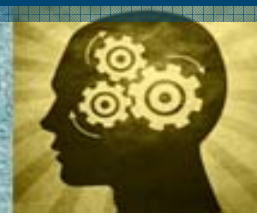
DATA FORMAT IMPEDANCE MISMATCH



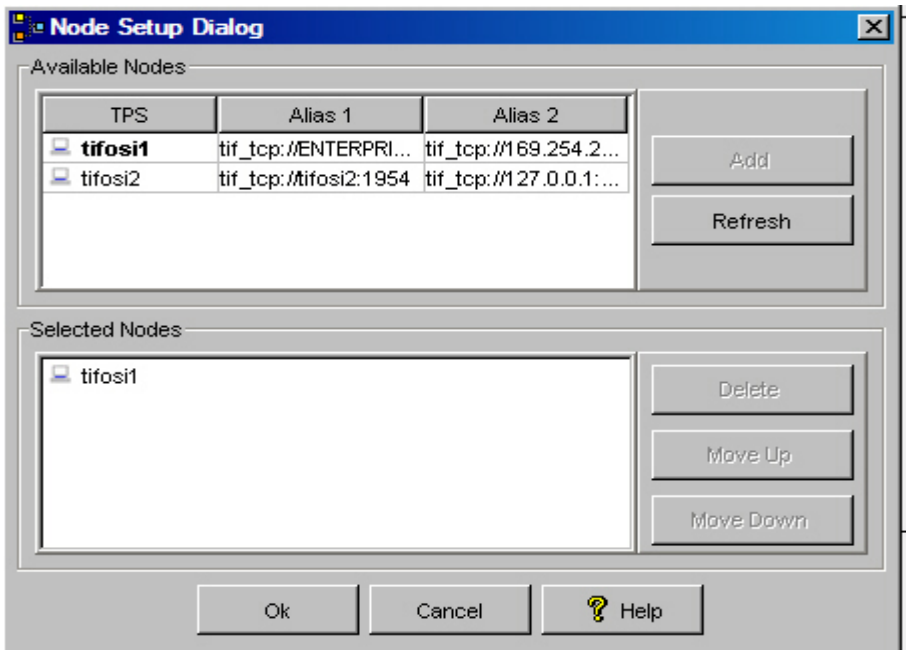


STEP 4 – DEFINING DISTRIBUTED ESB PROCESS

CONFIGURING BUSINESS PROCESS FOR FAILOVER

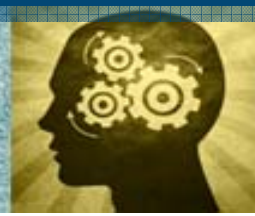


- Multiple node names for Services
- State failover for services
- Server level failover



STEP 4 – DEFINING DISTRIBUTED ESB PROCESS

CONFIGURING BUSINESS PROCESS FOR PERFORMANCE

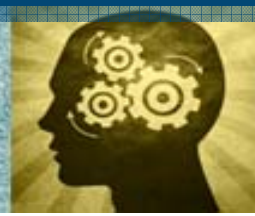


- Identifying Parallel data flows
- Dynamic rerouting of data based on load
- Identifying Heavy weight services (80/20 Rule)
- Running multiple instances of “heavy weight” services on different nodes
- Sub-Flows and Sub-Processes for effective BP representations
- Log/Trace level optimization
- Event tracing optimization

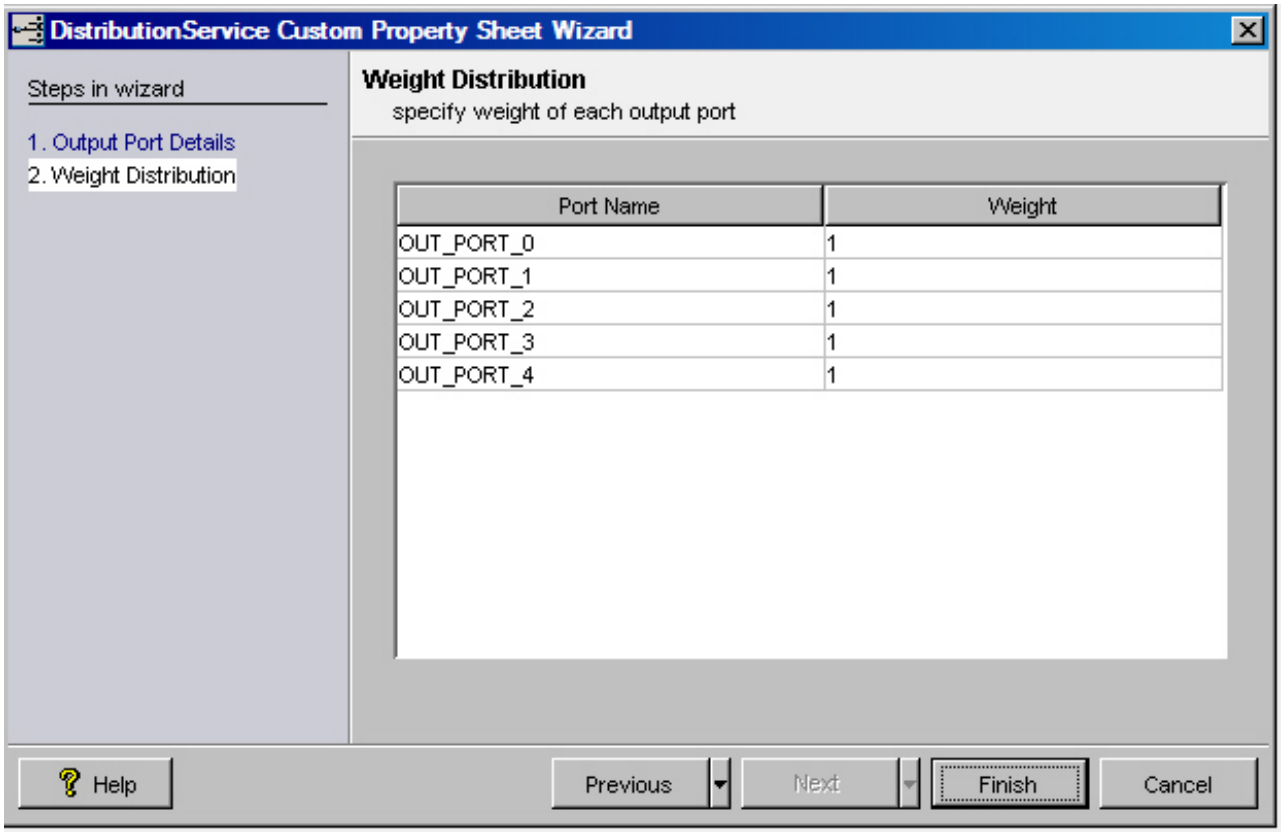
Fiorano
Peer-to-Peer Dataflow Pipelines™

STEP 4 – DEFINING DISTRIBUTED ESB PROCESS

CONFIGURING BUSINESS PROCESS FOR PERFORMANCE

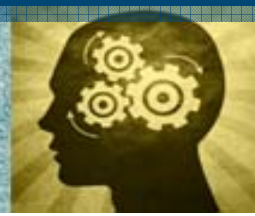


- Service level Load Balancing (static/Dynamic)



Fiorano
Peer-to-Peer Dataflow Pipelines™

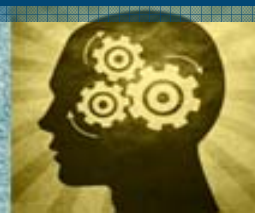
STEP 5 – DEPLOYING ESB PROCESS



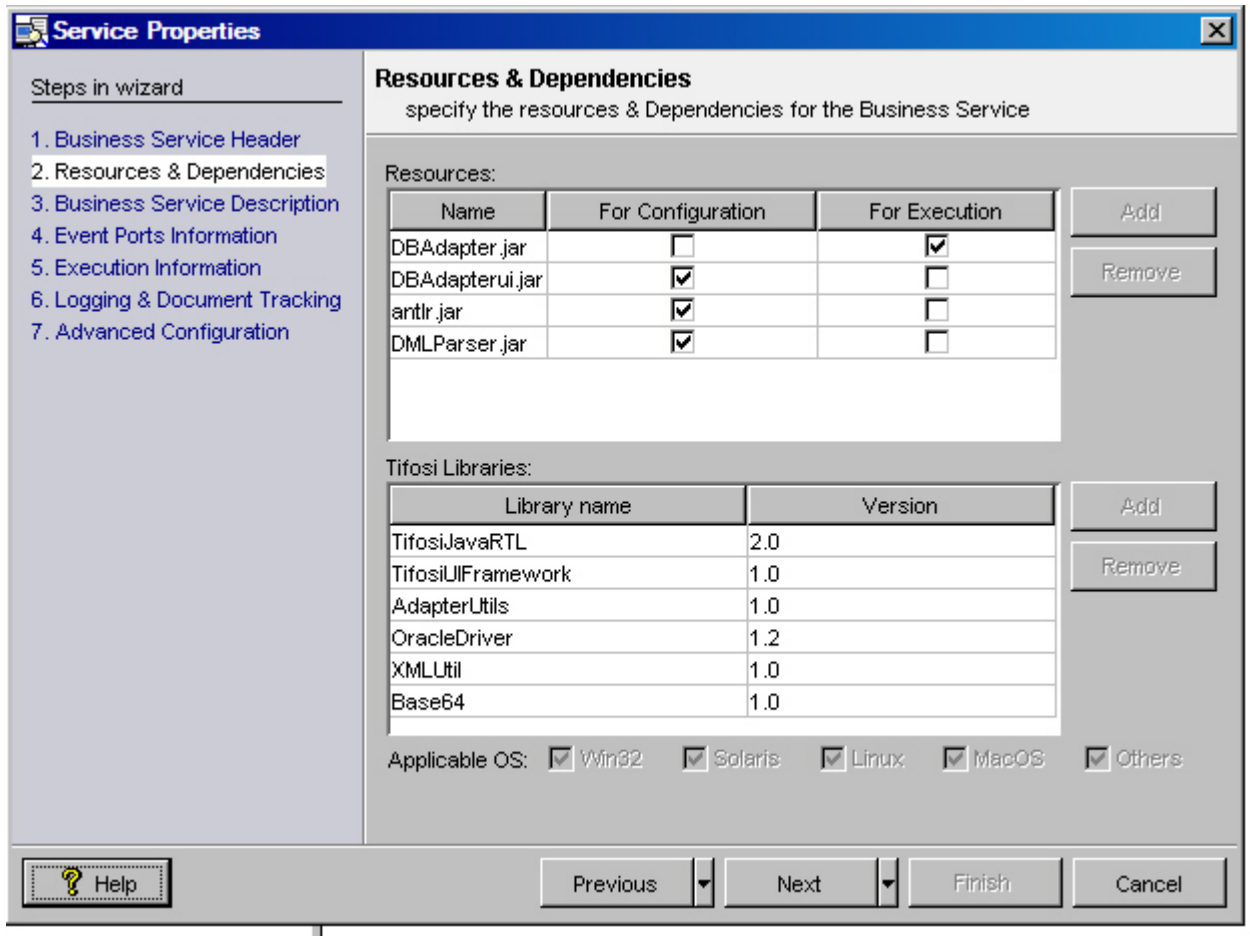
- Deployment - what does it mean ?
- Identifying Network Domain/Topology
- Manual Services vs. Auto-Launched Services
- Security issues for Deployment
- Service Development Languages and Platforms

STEP 5 – DEPLOYING ESB PROCESS

WHAT DOES IT MEAN?

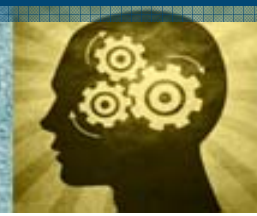


- Deployment Descriptor

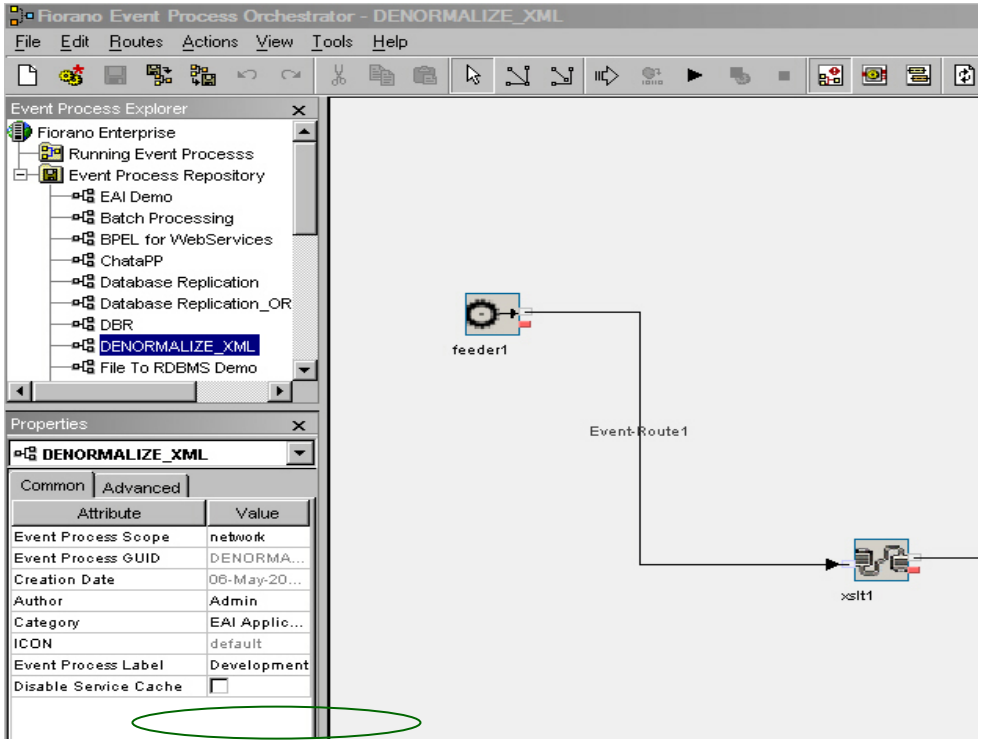


STEP 5 – DEPLOYING ESB PROCESS

WHAT DOES IT MEAN?

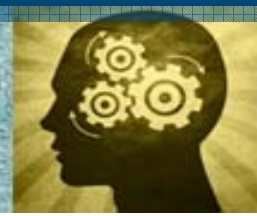


- Remote Deployment of Service and its dependencies
- Remote installation of service dependencies
- Caching

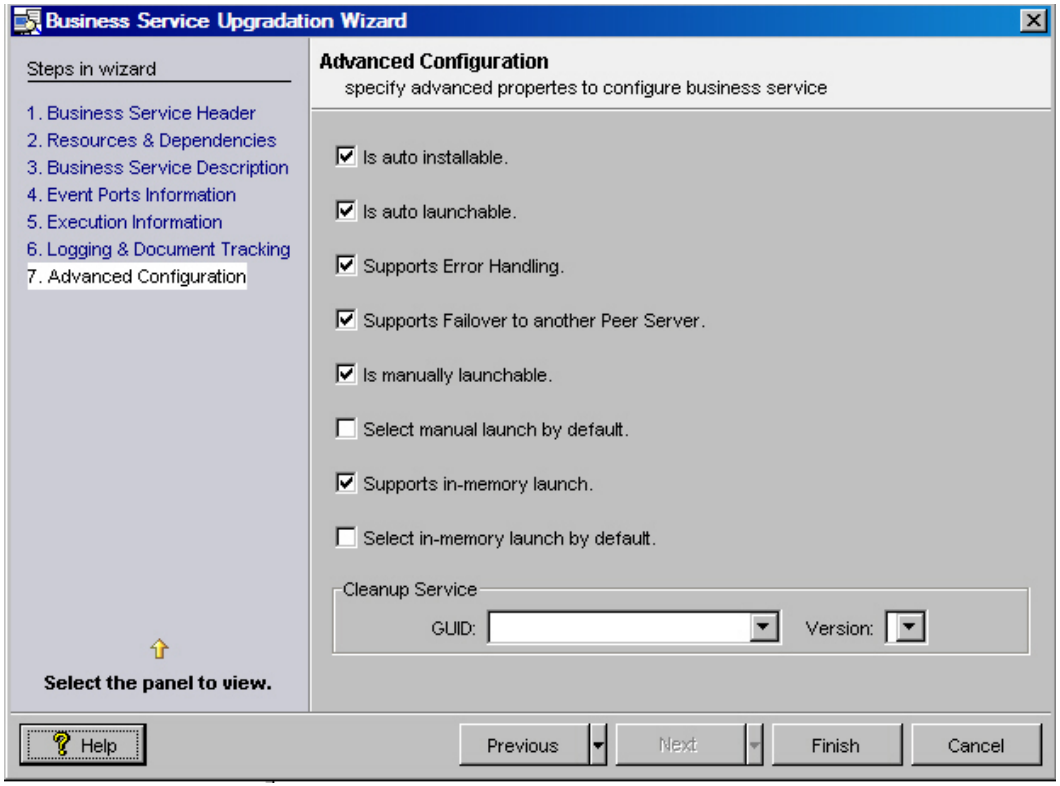


STEP 5 – DEPLOYING ESB PROCESS

WHAT DOES IT MEAN?

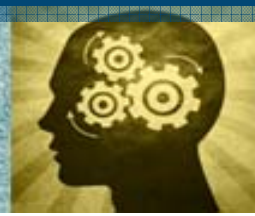


- Managing Dependencies / Order of installation /Order of Launch

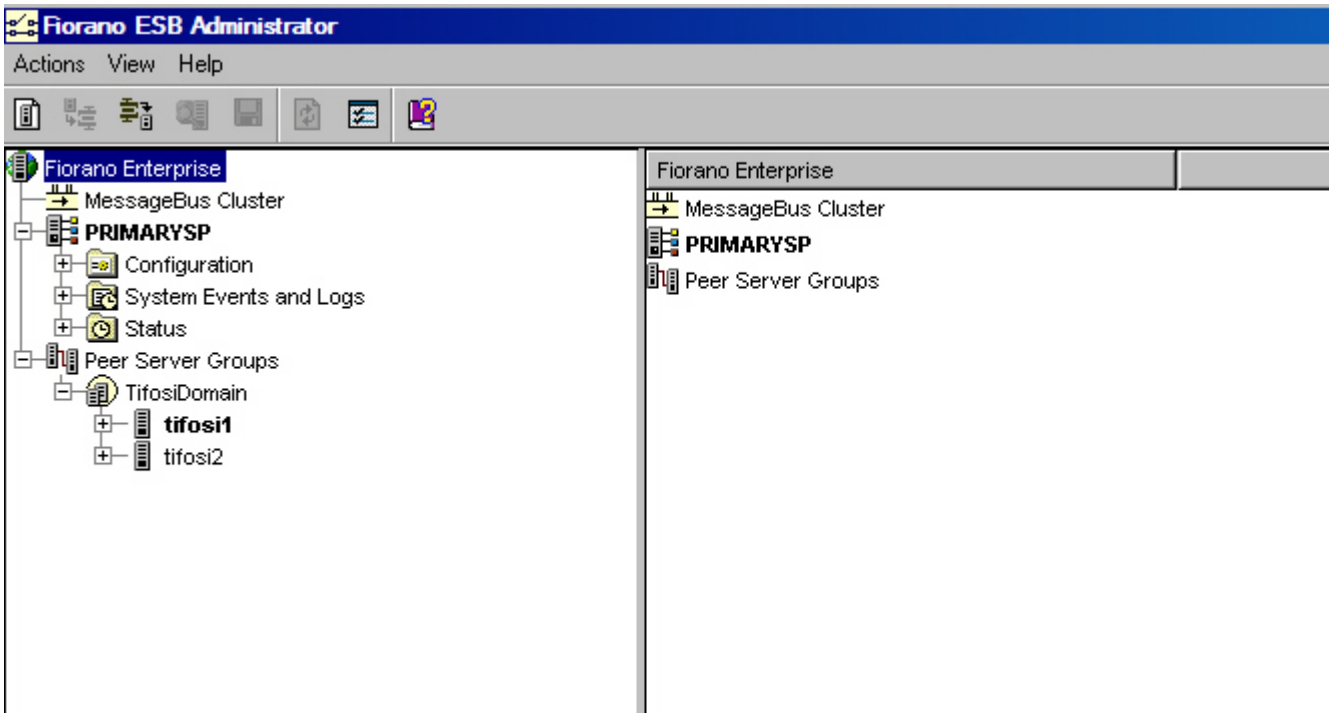


Fiorano
Peer-to-Peer Dataflow Pipelines™

STEP 5 – DEPLOYING ESB PROCESS
IDENTIFYING NETWORK DOMAIN/TOPOLOGY

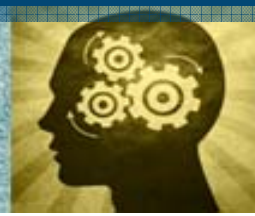


- Deployment Domain/Nodes



STEP 5 – DEPLOYING ESB PROCESS

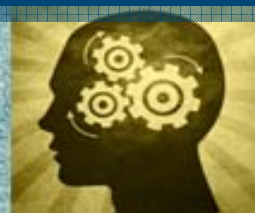
IDENTIFYING NETWORK DOMAIN/TOPOLOGY



- Domain/Node Name Alias (Hierarchical Domains)
- Configuration Alias

STEP 5 – DEPLOYING ESB PROCESS

MANUAL VS AUTO LAUNCH

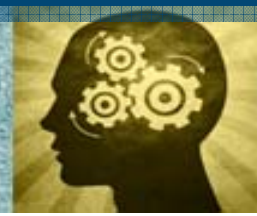


Manual Services

- Executed externally to the ESB (Servlets, EJBs etc)
- Combination of managed and unmanaged components
- Managed Components like:-
  - ✓ A Webservice deployed in a Webservice container
  - ✓ An EJB deployed in J2EE container
  - ✓ A COM Object deployed in COM+ server
  - ✓ A CORBA based server Object deployed in an ORB
  - ✓ Windows NT/2K Service
- Unmanaged Components like:-
  - ✓ Java executable archive
  - ✓ A C/C++ executable
  - ✓ Legacy Application running in a mainframe environment.
  - ✓ A Unix shell program (functioning within a pipe-and-filter style architecture).

STEP 5 – DEPLOYING ESB PROCESS

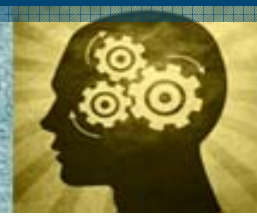
MANUAL VS AUTO LAUNCH



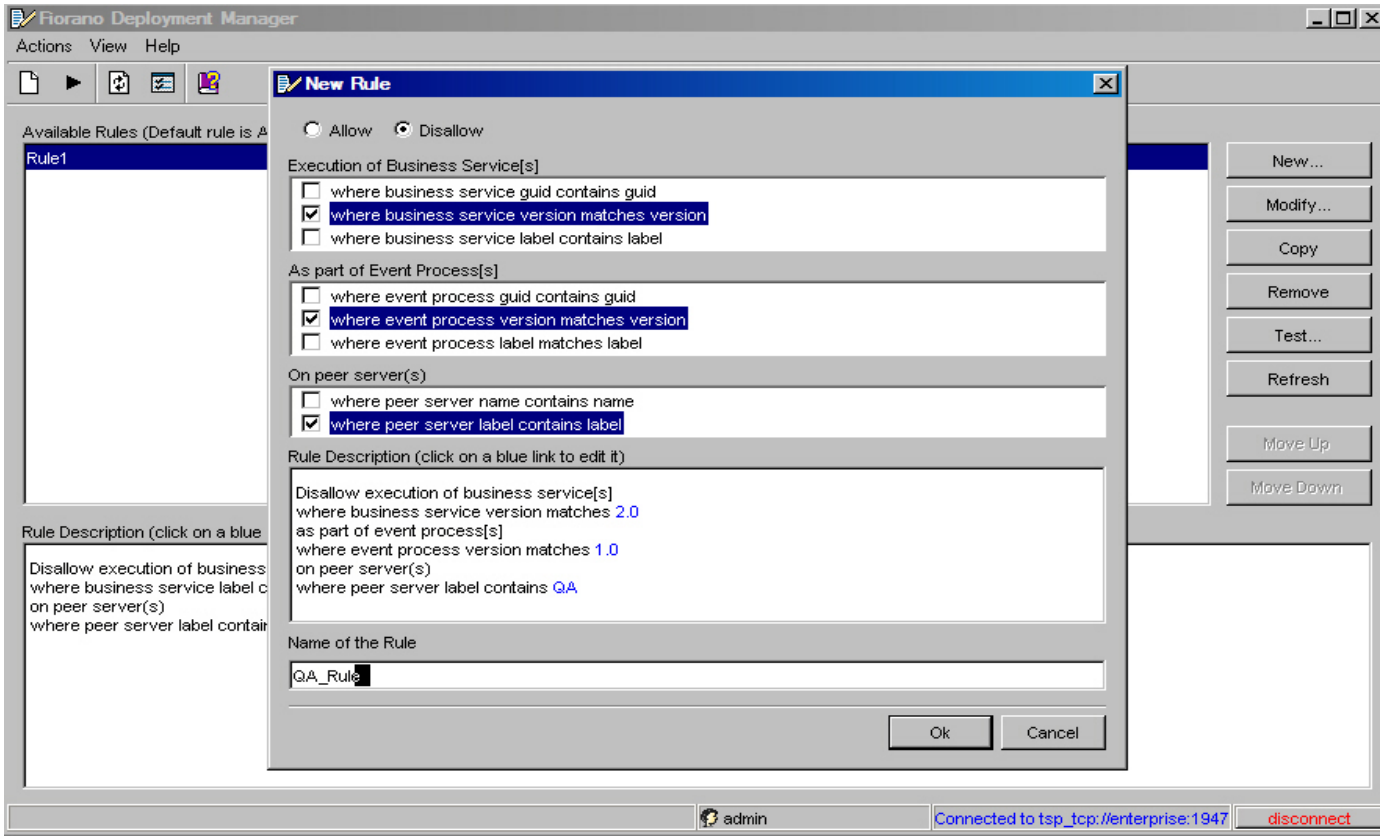
Auto Launched Services

- Native ESB services- managed and launched by ESB containers
  - Auto start/stop and restart of these services
  - Connectivity management
  - Fain grained monitoring

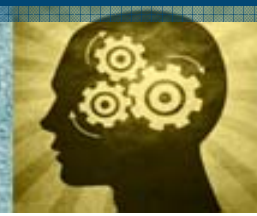
STEP 5 – DEPLOYING ESB PROCESS
SECURITY ISSUES



- Deployment Manager to restrict service deployment

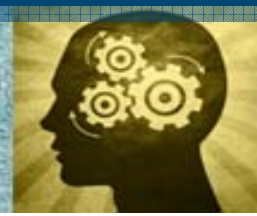


STEP 6 – LAUNCHING & MONITORING ESB PROCESS



- Remote Launch of Business Process
- Monitoring state of Application and all its associated service instances
- Runtime hooks to determine state of service
- Real-time data debugging
- Business Process Monitoring
  - ✓ SNMP, JMX etc
  - ✓ Monitoring of Servers

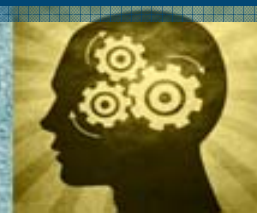
STEP 6 – LAUNCHING & MONITORING ESB PROCESS



Launch of ESB process

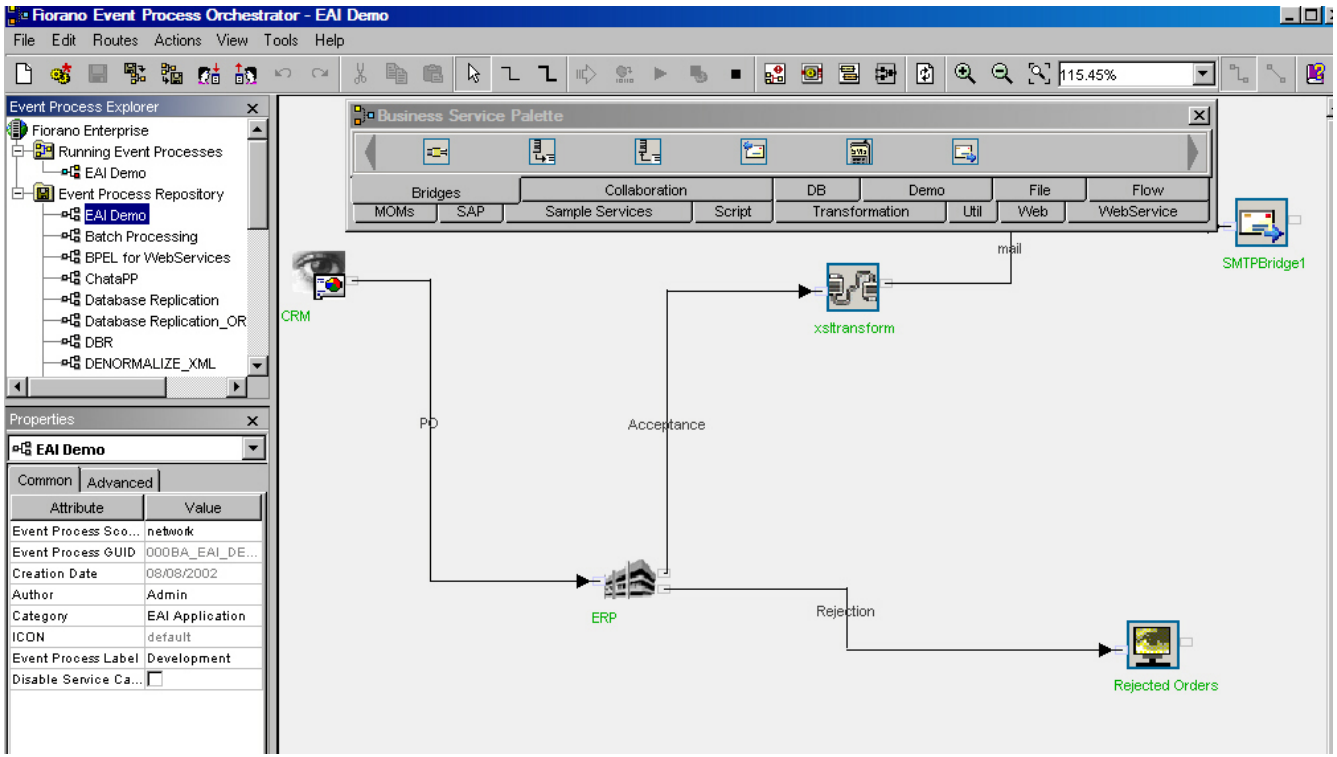
- Remote Launch of ESB Services
- Service Instantiation
  - ✓ On-demand/On-Event Instantiation
  - ✓ Auto Instantiation
- Re-Launch of Business Process
  - ✓ Auto re-launch on different set of nodes
  - ✓ Rules based re-launch of business process

STEP 6 – LAUNCHING & MONITORING ESB PROCESS



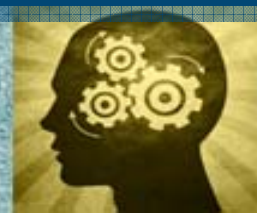
Monitoring state of Application and all its associated service instances

- ✓ Local Queue Monitoring and Management
- ✓ Identify performance/scalability bottlenecks in Application
- ✓ Identify parallel flows in Application



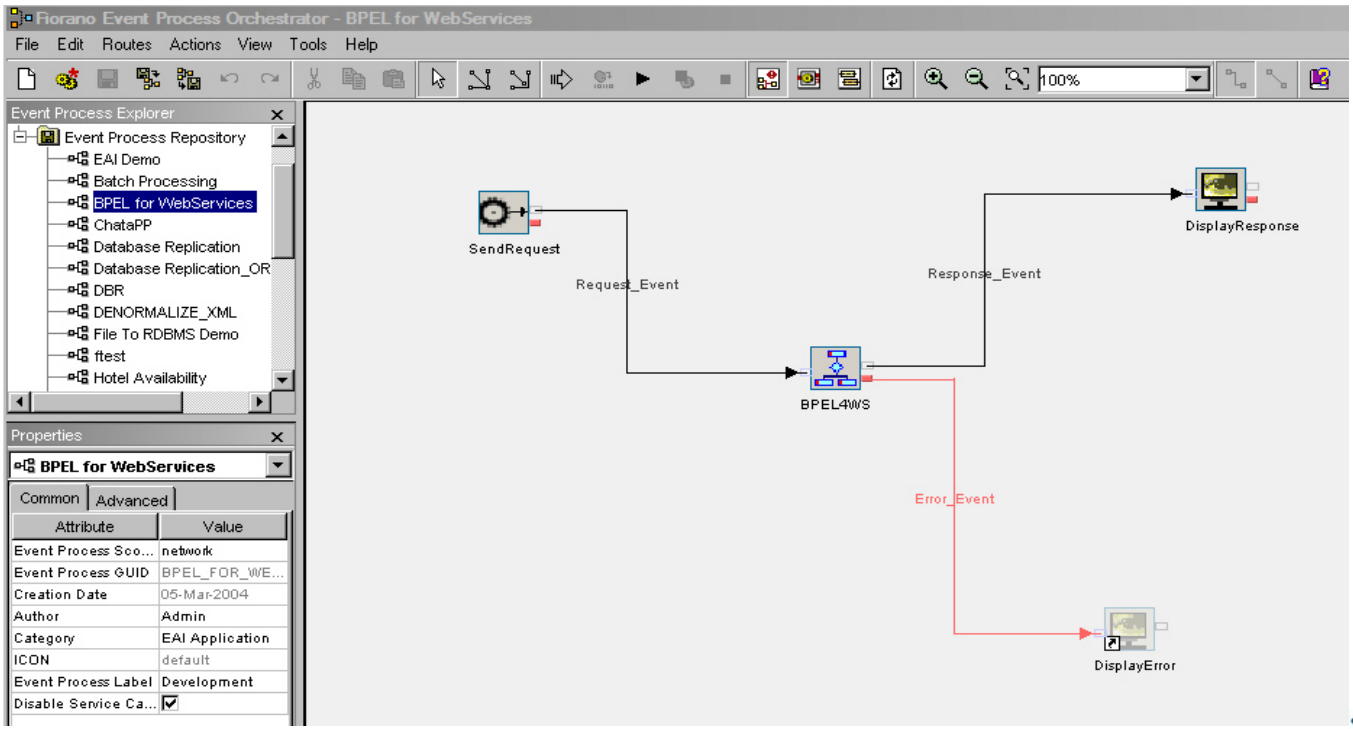
Fiorano
Peer-to-Peer Dataflow Pipelines™

STEP 6 – LAUNCHING & MONITORING ESB PROCESS



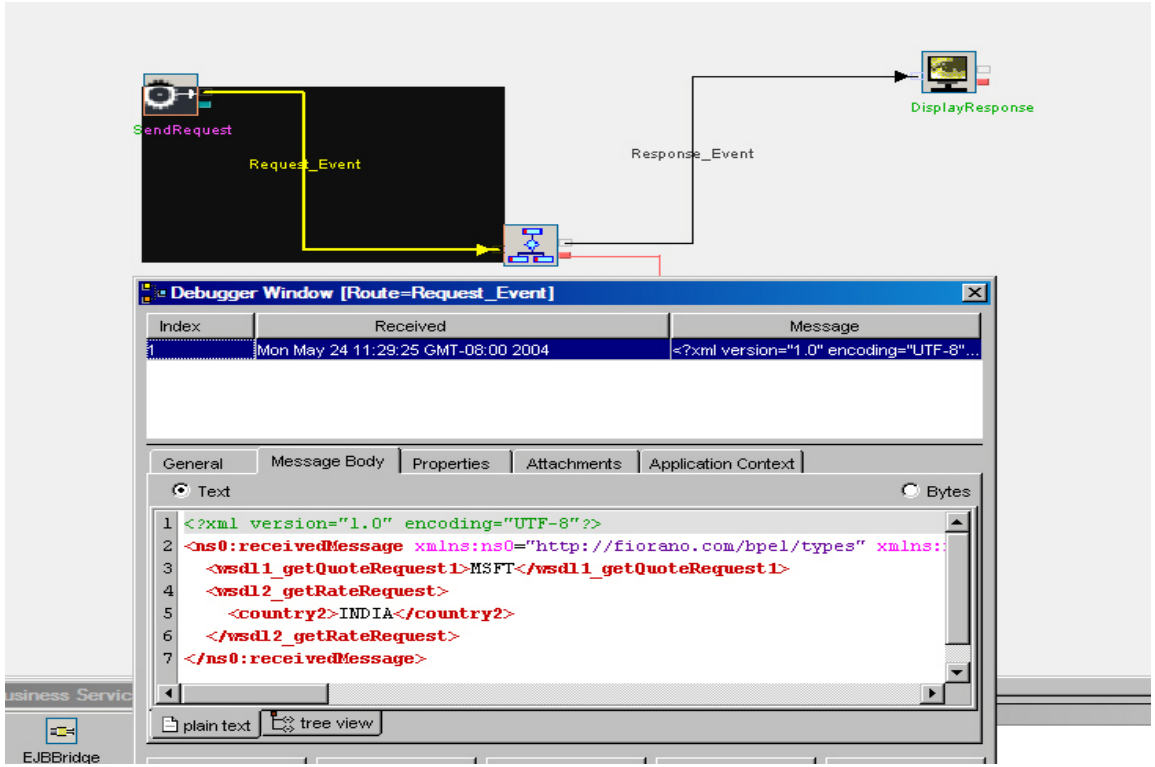
Runtime hooks to determine state of service

- ✓ Runtime tracing and Logging
- ✓ Sub-Flows to handle Error conditions
- ✓ Alert Handlers



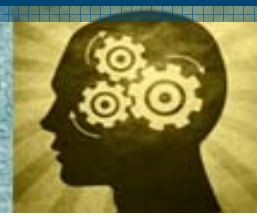
STEP 6 – LAUNCHING & MONITORING ESB PROCESS

Real-time data debugging



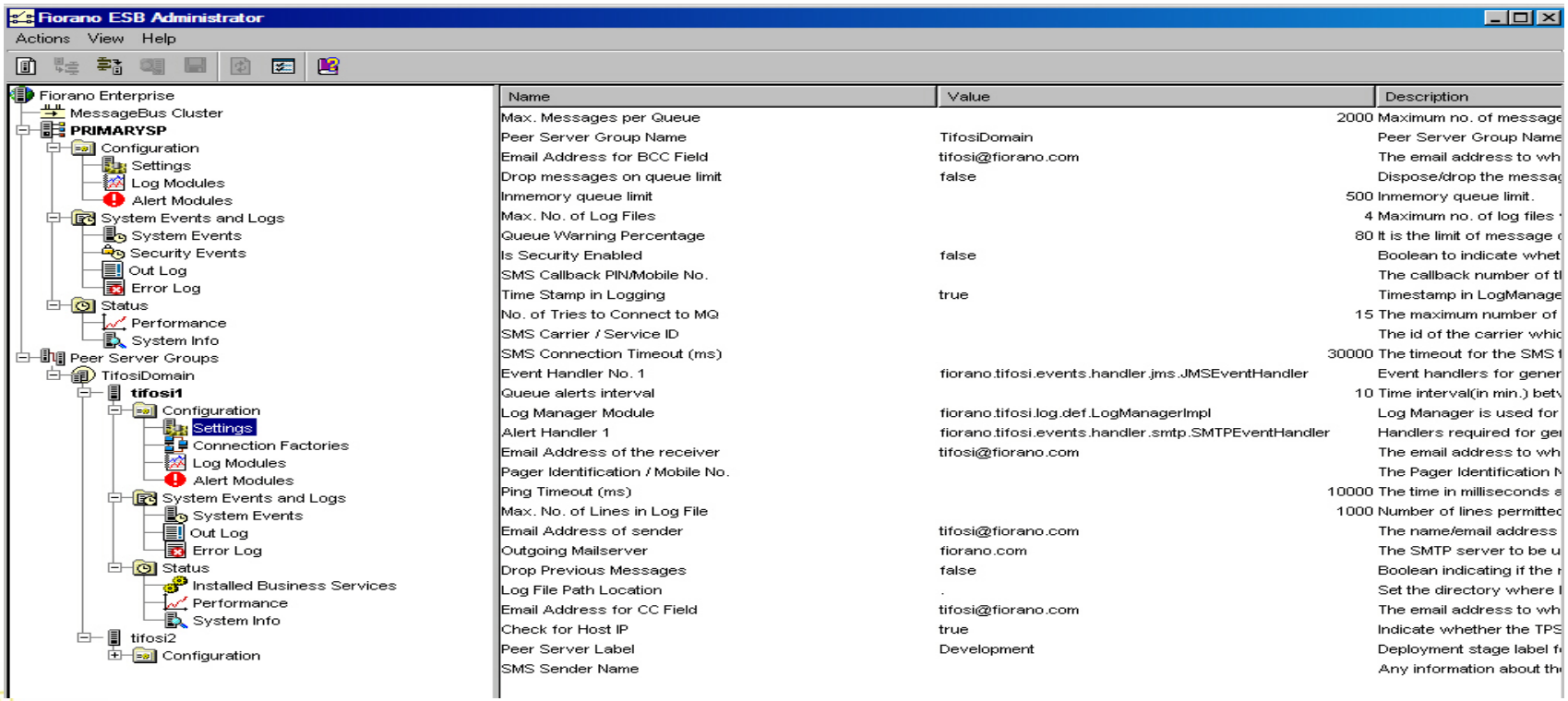
Fiorano
Peer-to-Peer Dataflow Pipelines™

STEP 6 – LAUNCHING & MONITORING ESB PROCESS



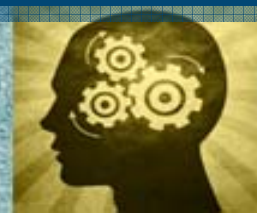
Business Process Monitoring

- SNMP, JMX etc
- Monitoring of Servers



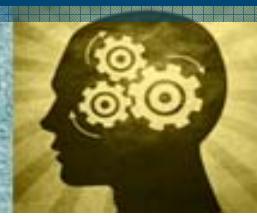
Fiorano
Peer-to-Peer Dataflow Pipelines™

STEP 6 – CHANGE MANAGEMENT & VERSIONING



- Dynamic Extensions to Business process
  - ✓ Static vs. Dynamic
  - ✓ Impact Analysis
  - ✓ Extend Business process to include new services
  - ✓ Extend Business process to include data services with different data formats
  - ✓ Optimize Business process for performance, scalability
  - ✓ Extend Business processes to handle network configuration changes
- Dynamic changes to Business process
- Service and Application Versioning

STEP 6 – CHANGE MANAGEMENT & VERSIONING

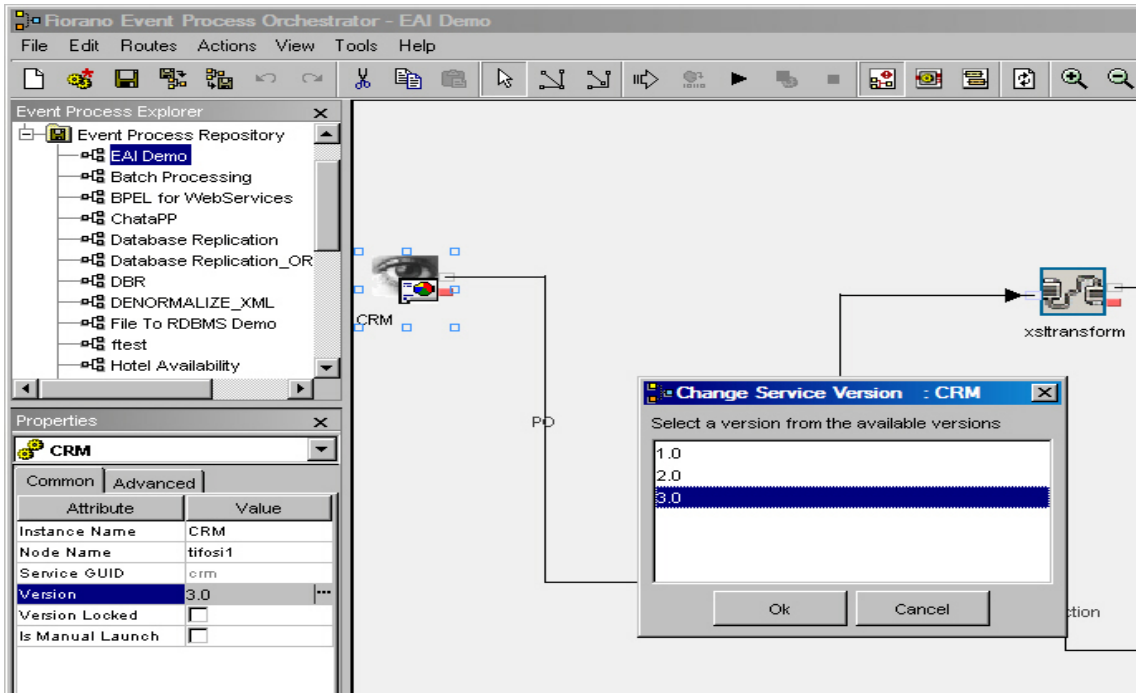


- Configuration Management
  - ✓ Moving from one stage to another (QA to Deployment)
  - ✓ Moving from one environment to another (customer to internal testing)
- Extension to Business process
  - ✓ Updating Data Consistency Application to include the data center in San Francisco
- Change to Business Process
  - ✓ Flexibility to adapt to new technologies (WebService instead of a C++ stand alone Application)

STEP 6 – CHANGE MANAGEMENT & VERSIONING

Service and Application Versioning

- Manage and maintain multiple versions of services along with Labels
- Quickly allow migration from one service version to another

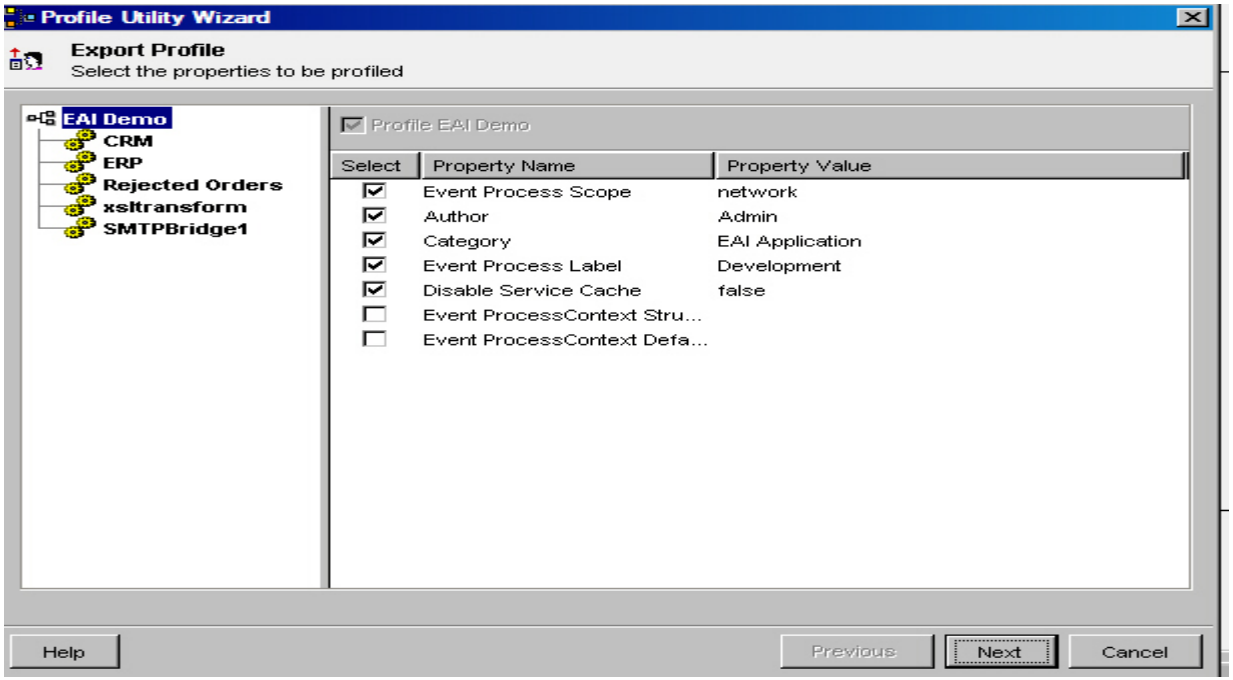


Fiorano
Peer-to-Peer Dataflow Pipelines™

STEP 6 – CHANGE MANAGEMENT & VERSIONING

Configuration Management

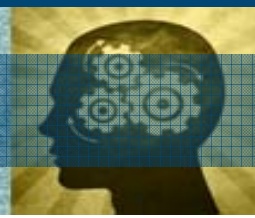
- Moving from one stage to another (QA to Deployment)
- Moving from one environment to another (internal to client deployment)



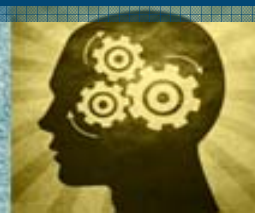
Fiorano
Peer-to-Peer Dataflow Pipelines™

Fiorano
Peer-to-Peer Dataflow Pipelines™

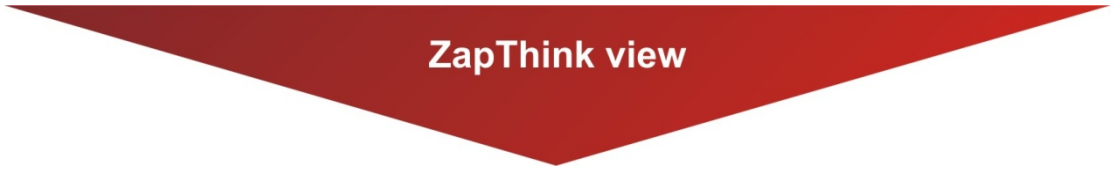
PUTTING THEORY TO PRACTICE



TECHNOLOGY HYPE IS CONFUSING THE ISSUES WITHOUT PROVIDING A CLEAR PATH FORWARD



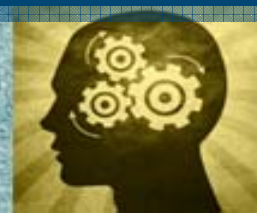
Example:ZapThink view



- Need to choose standards that work today and will evolve for future
- Solve the real integration problem – more than a Proof of Concepts
- Must build incrementally on top of existing systems

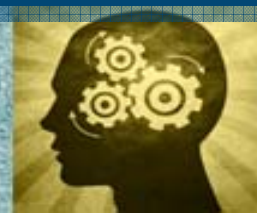
Fiorano
Peer-to-Peer Dataflow Pipelines™

STANDARDS REDUCE COSTS BUT…
FUNDAMENTAL PROBLEMS REMAIN!



- Business Person's View
  - ✓ High-level model of business process flow
- IT Level View
  - ✓ Implementation flow differs substantially from business process view
- Impedance mismatch creates fundamental problems
  - ✓ Implementations have too many "moving parts"
  - ✓ Business-level change requirements difficult and time-consuming to implement

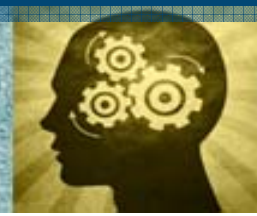
THE FUNDAMENTAL PROBLEM – DIVERGENT BUSINESS TECHNOLOGY

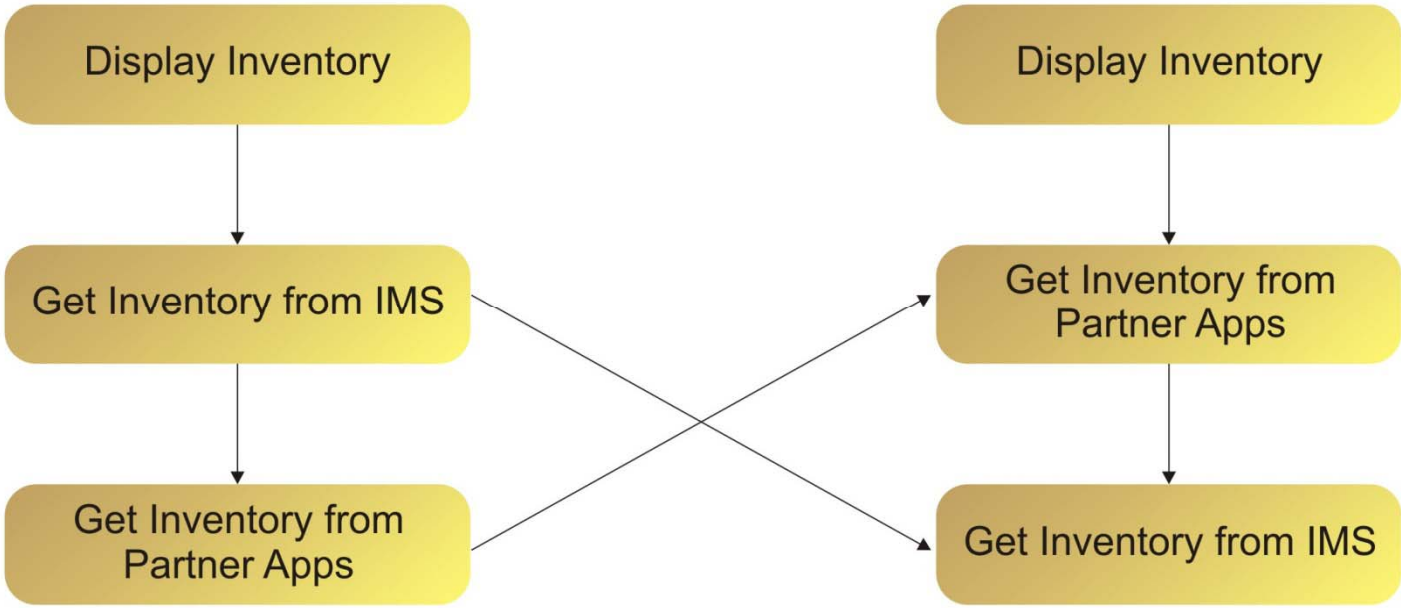


BUSINESS		TECHNOLOGY
Check Inventory		Web Apps
Get Inventory from IMS	→	Inventory Management System
Get Inventory from Partner Apps		Partner Apps

Fiorano
Peer-to-Peer Dataflow Pipelines™

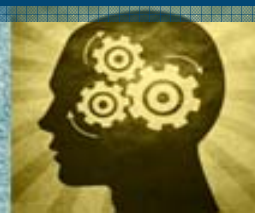
A CHANGE IN BUSINESS PROCESS

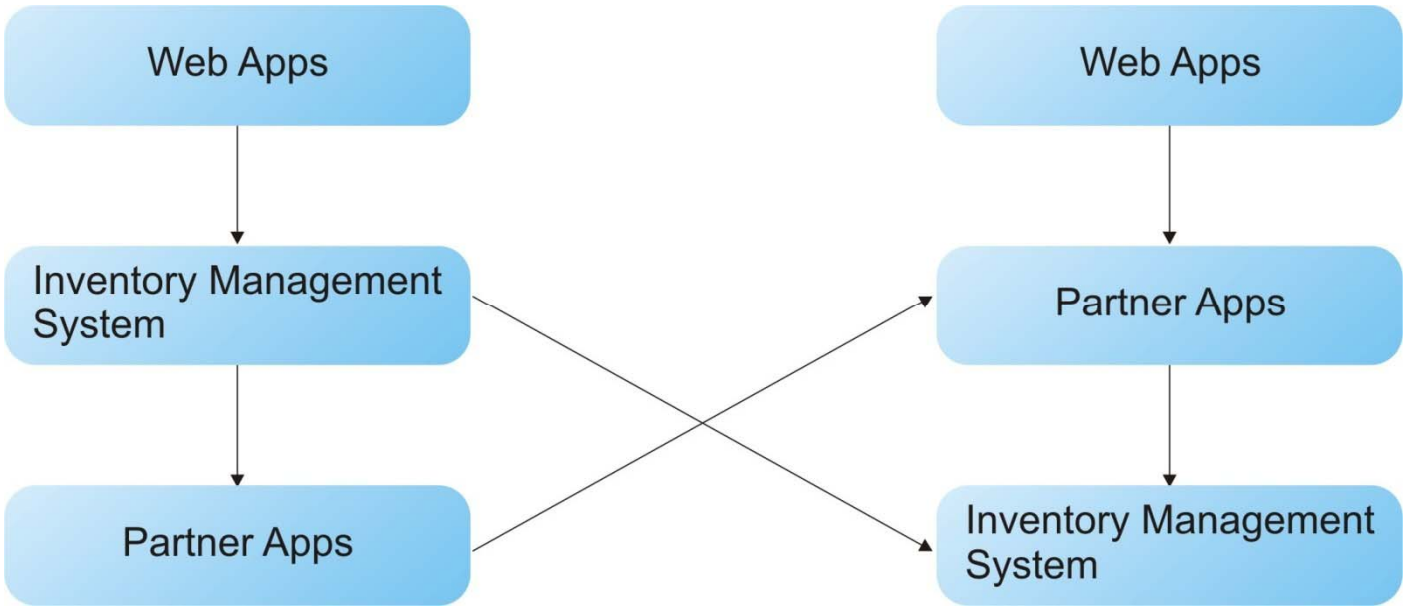




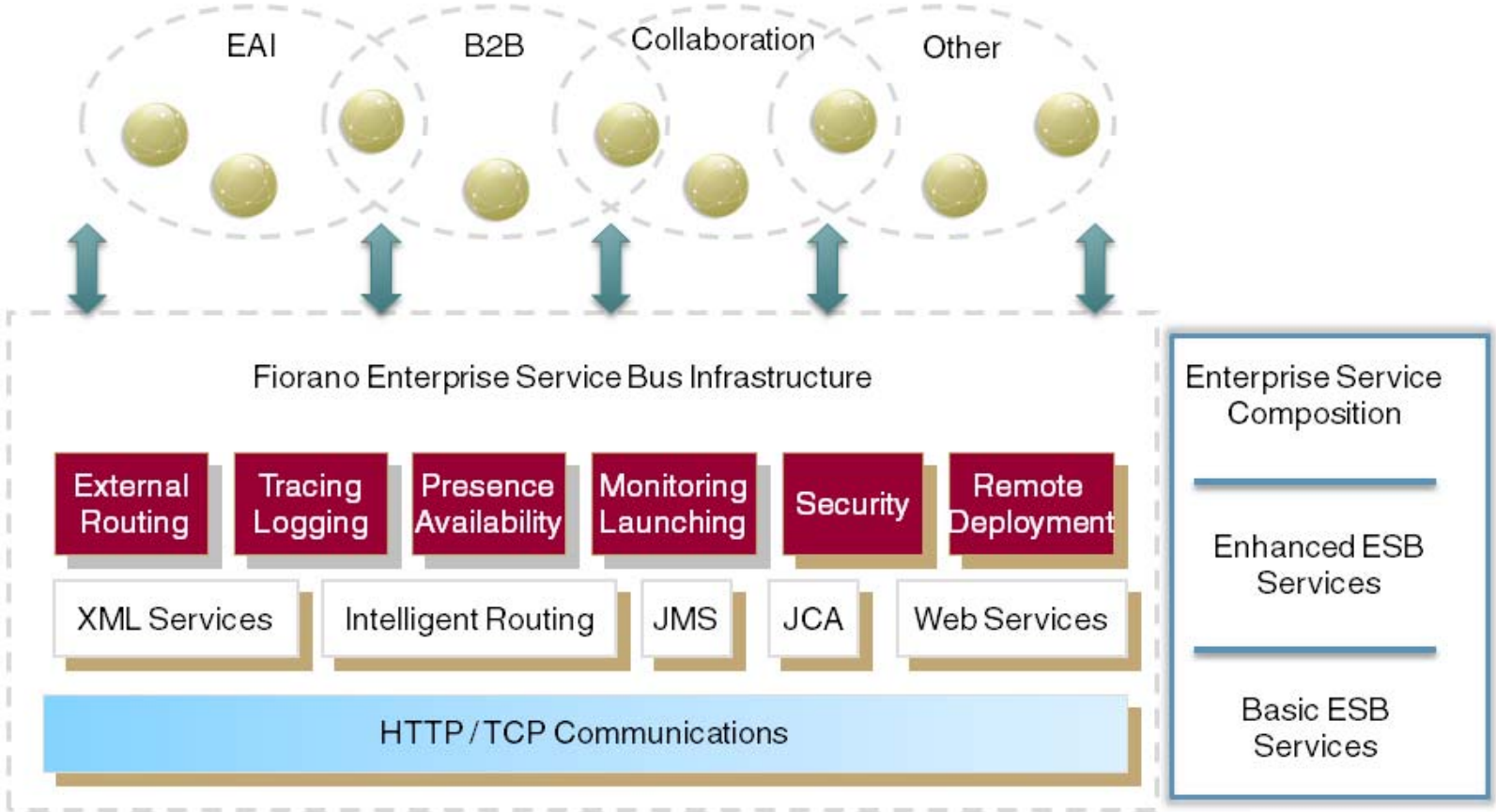
Fiorano
Peer-to-Peer Dataflow Pipelines™

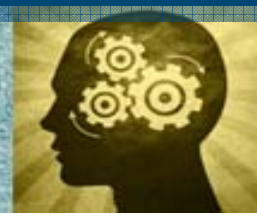
IS NOT EASILY MAPPED TO IMPLEMENTATION LEVEL





FIORANO ESB
SECOND GENERATION ESB





THANK YOU!

FOR MORE DETAILS, VISIT www.fiorano.com

Fiorano
Peer-to-Peer Dataflow Pipelines™