June 2005

Interoperability between Microsoft BizTalk Server 2004 and SAP XI 3.0

Tilo Böttcher, SAP Program Manager CTSC, Global SAP Alliance, Microsoft Corporation
Jürgen Daiberl, SAP Program Manager CTSC, Global SAP Alliance, Microsoft Corporation
Thomas Reimer, Senior Manager IT Consulting, Resco GmbH
Christian Brückner, Senior Consultant Microsoft Technologies, Resco GmbH
Rüdiger Lühr, Senior Consultant SAP Technologies, Resco GmbH

Summary

This white paper provides a concept on how to integrate Microsoft BizTalk Server 2004 and SAP XI based upon a scenario of both integration platforms acting as cooperating systems and integrating different parts of the application landscape. The white paper combines conceptional work on a double-hub integration architecture and practical experiences from existing projects. A double-hub integration architecture consisting of Microsoft BizTalk Server 2004 and SAP XI 3.0 encapsulates all SAP centric integration scenarios and interfaces within the XI hub and all other scenarios within the BizTalk hub. To achieve a seamless and comprehensive connection between those two hubs several aspects regarding the different integration layers have to be considered. The white paper covers all relevant aspects required to establish integration at transport, messaging, process management and monitoring layer. At the transport and messaging layer practical walkthroughs demonstrate the steps necessary to connect SAP XI and BizTalk using the existing mechanisms and adapters. In order to provide a blue print for real-life scenarios the tradeoff between Web Services and conventional RFC/IDoc is discussed.

Applies to

- Microsoft BizTalk Server 2004 SP1
- BizTalk Adapter v2.0 for mySAP Business Suite
- SAP® Exchange Infrastructure (SAP XI) 3.0
- SAP Connector for Microsoft .NET 1.0.3

Keywords

Microsoft BizTalk Server, SAP Exchange Infrastructure, SAP XI, Web Services, SOAP, ALE, IDoc, WS-I, WS-*

Microsoft®

Level of difficulty

Technical consultants, Architects, Developers, IT Managers

Contact

This document is provided to you by the Collaboration Technology Support Center of Microsoft.
For feedback or questions you can contact the CTSC at ctsc@microsoft.com.
Please check the .NET interoperability with SAP area on

- Microsoft Website: http://www.microsoft-sap.com -> Technology or Interoperability
- Microsoft Developer Network: http://msdn.microsoft.com Web Services developer corner -> Interoperability
- SAP Developer Network (http://sdn.sap.com) for any updates or further information.

The information contained in this document represents the current view of the Co-Editors on the issues discussed as of the date of publication. Because the Co-Editors must respond to changing market conditions, it should not be interpreted to be a commitment on the part of the Co-Editors, and the Co-Editors cannot guarantee the accuracy of any information presented after the date of publication.

This document is for informational purposes only. NEITHER OF THE CO-EDITORS MAKES ANY WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Contents

Introduction

While Microsoft BizTalk Server 2004 has become one of the leading integration and business process management servers with more than 4600 installations world-wide SAP XI is becoming the de facto standard for SAP integration scenarios. As a consequence several companies are facing the question whether to use one of the platforms or combine both within a best of breed architecture. This whiter paper embarks on the strategy to establish a double-hub integration architecture consisting of Microsoft BizTalk Server 2004 and SAP XI 3.0 where all SAP centric integration scenarios and interfaces are being encapsulated within the XI hub and all other scenarios within the BizTalk hub.

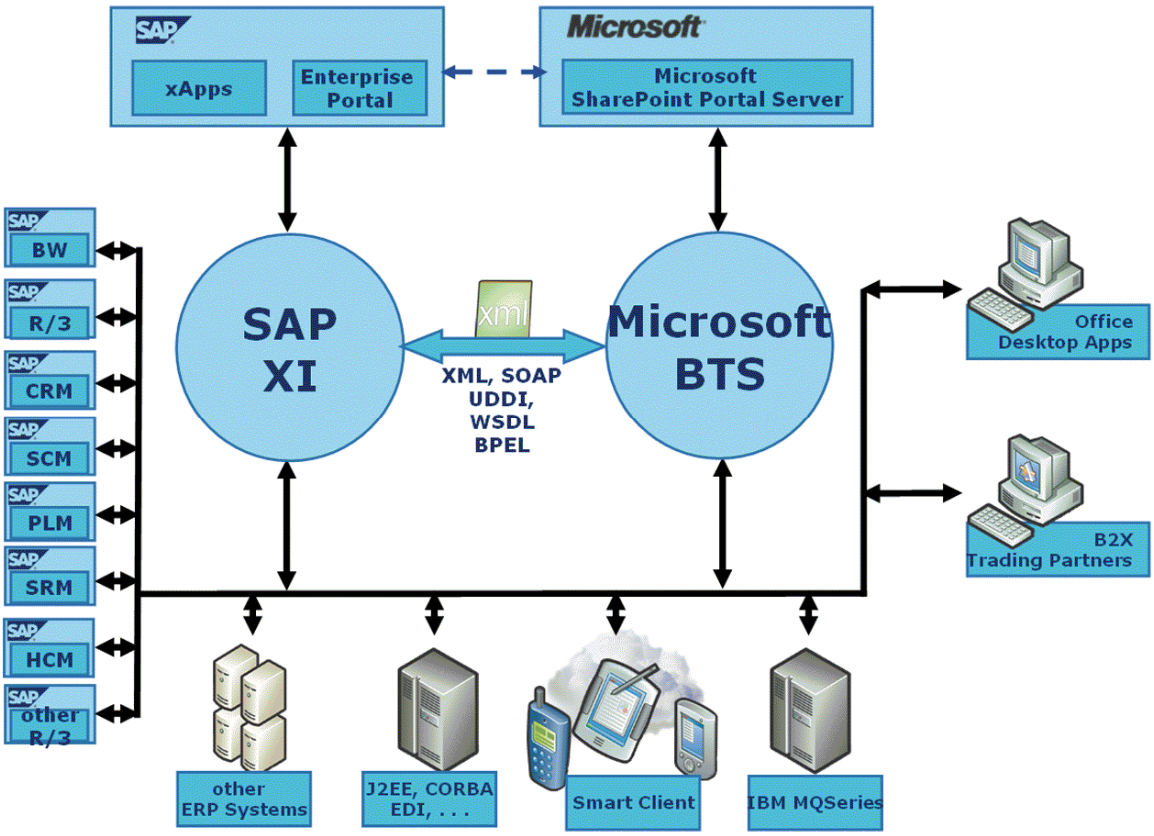

Figure 1 Coexistence of Microsoft BizTalk Server and SAP XI

To achieve a seamless and comprehensive connection between the two hubs several aspects regarding the different integration layers have to be considered. The white paper covers all relevant aspects required to establish integration at transport, messaging, process management and monitoring layer.

All solutions and architectural approaches within this paper are based on the current versions of Microsoft BizTalk Server (Version 2004 SP1) and SAP XI (Version 3.0). The main objective is to provide proven and tested integration mechanisms that are close to reality. Thus all future enhancements of the products especially in the relevant areas of Web Services will not be covered within this paper. Due to the fact that these enhancements will have significant impact on future integration scenarios in about 1-2 years the paper will not develop sophisticated techniques that require intensive implementation and investments respectively. Therefore all the presented approaches will stay abreast of changes and will focus on flexible and loosely coupled mechanisms.

Architecture

A double-hub integration architecture requires the mapping of both platforms' functionalities in order to establish a consistent and homogenous landscape. BizTalk and XI have slightly deviating architectures using different terms and definitions for the components. The white paper defines an abstract architecture with generic layers (Figure 2):

- **Transport Layer**: The transport layer comprises the protocol stacks (e.g. HTTP, SOAP, RFC, FTP, MQSeries), Bindings and APIs like COM, JMS or BAPI and the corresponding adapters.
- **Messaging Layer**: The messaging layer consists of engines and pipelines that handle the processing messages by parsing/serializing, routing, mapping and persisting them.
- **Process Management Layer**: The process management layer extends the messaging layer by adding stateful and stateless process handling.
- **Monitoring Layer**: The monitoring layer comprises monitoring mechanisms for all layers. It can be divided into technical monitoring targeting the infrastructure level, messaging monitoring and business process monitoring from a process user's viewpoint.

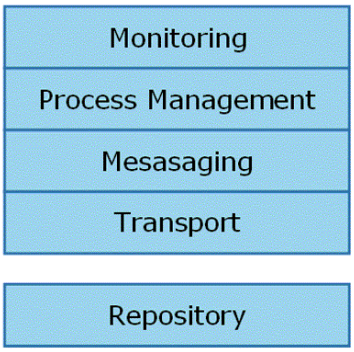

Figure 2 Integration Layers

Beside these functional layers the repository layer includes all mechanisms for storing metadata, development objects and configuration data. The double-hub architecture requires seamless and loss-free metadata sharing in order to create comprehensive integration scenarios with spanning processes and interfaces.

Microsoft BizTalk Server 2004 provides full support of all five integration layers. The core messaging engine uses pipelines to process inbound and outbound messages that are stored and forwarded using one or many SQL Server based MessageBox databases. Messages are routed inside of BizTalk using message context subscriptions that enable one-to-one or one-to-many distributions of source messages. Any physical connection between BizTalk and the outside world is established by adapters that have a one-to-one connection to a receive- or send pipeline (Figure 3). The orchestration engine is triggered by message context subscriptions at message engine level and executes process instances. In addition to the orchestration engine the Business Rule Engine enables dynamical and versioned changes of business processes at runtime. Message monitoring is done by the Health and Activity Tracking tool while business process monioring can be achieved using the Business Activity Monitoring Framework.

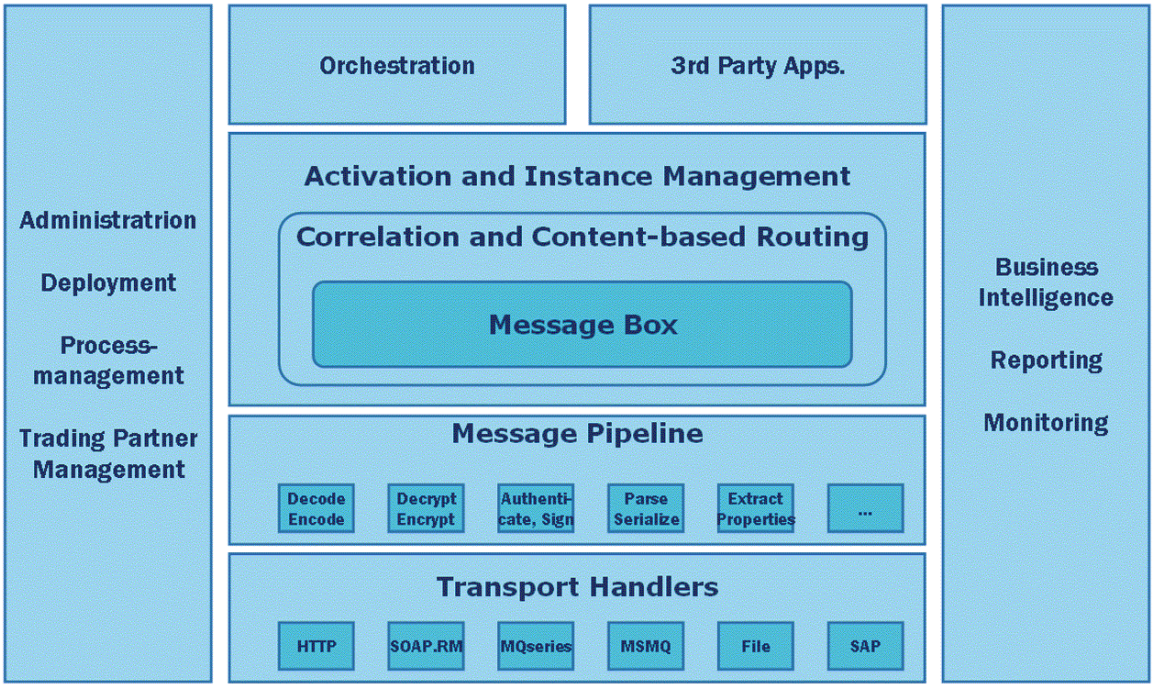

Figure 3 Microsoft BizTalk Server 2004 Architecture

SAP Exchange Infrastructure (XI) of SAP NetWeaver also covers all five integration layers. The Integration Server is the runtime component for communication, routing and process execution. It is running different engines separating the concerns:

- The Business Process Engine is responsible for executing and persisting integration processes.
- The Integration Engine is responsible for central Integration Server services for example routing and mapping.
- The Adapter Engine runs all communication with connected systems through communication type specific adapters.

Beyond it a central monitoring tracks all message and process activities. The Integration directory holds message and process definitions in combination with the Integration repository. The SAP System Landscape Directory (SLD) server contains component information, a landscape description, and a name reservation.

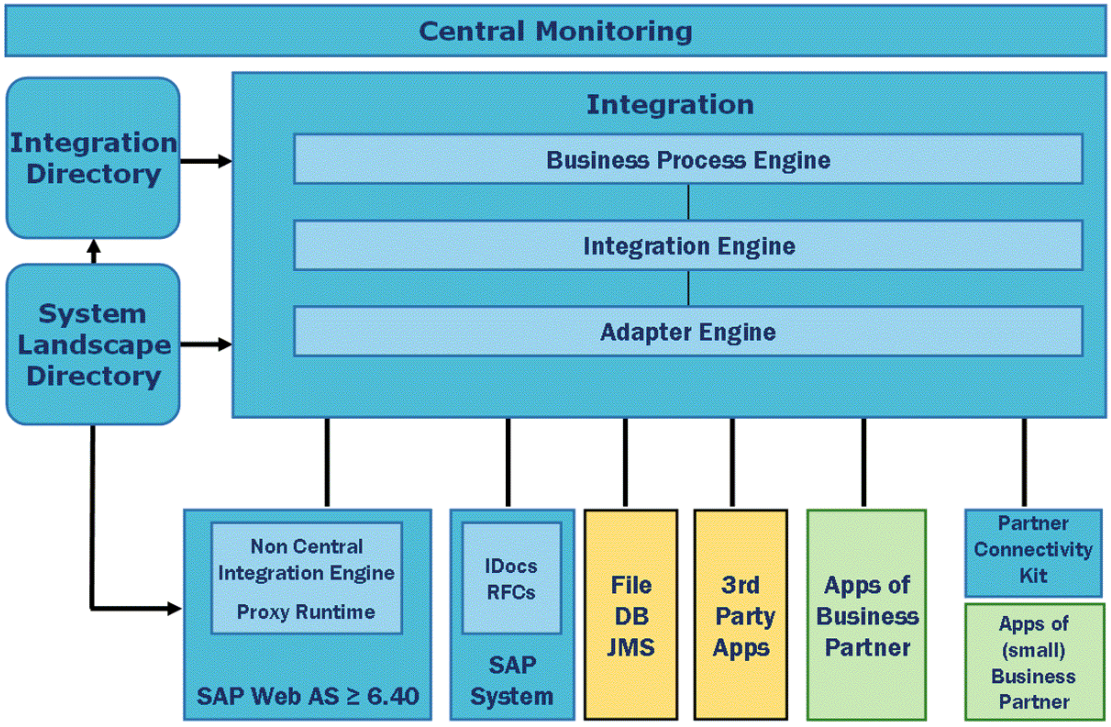

Figure 4 SAP XI 3.0 Architecture

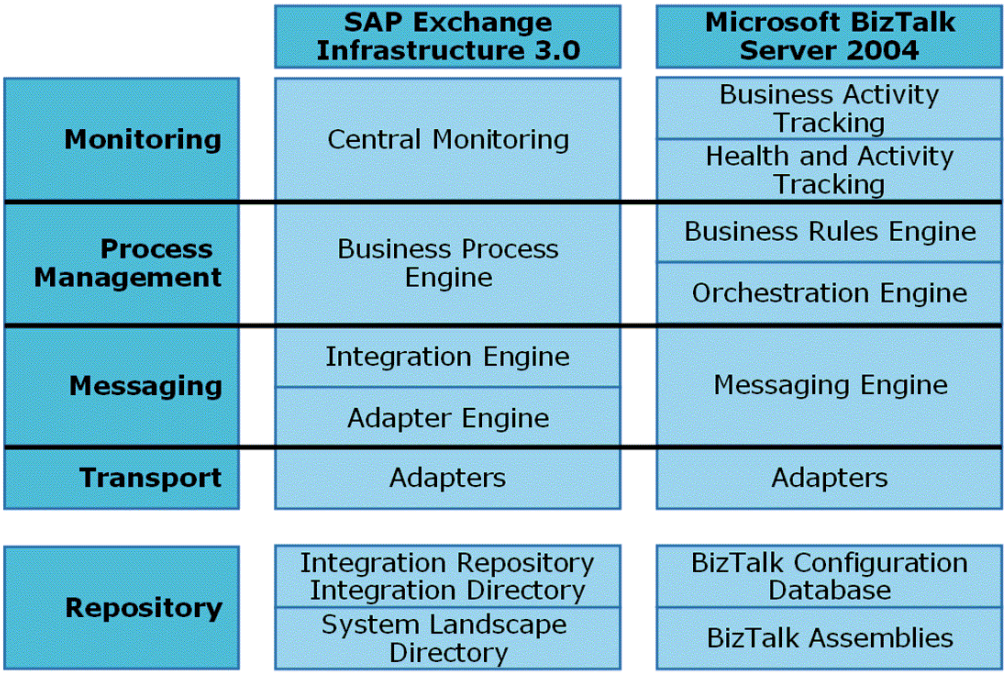

Figure 5 Functional Mapping of the Integration layers

Integration Scenarios

Integration scenarios differ in their demand for transactionality, throughput/turnover, reliability and communication style. Therefore the paper classifies the elementary scenarios and patterns in order to derive their impact for the integration mechanisms.

Synchronous vs. Asynchrounous Communication

Generally two types of communication mechanisms can be devided: asynchronous and synchronous.

Asynchronous communication moves information between one or many applications in an asynchronous way. In doing so it decouples source and target end points and makes them indenpedent from each other. The model is based on a queuing mechanism that enables parallel processing. The primary advantage of asynchronous communication is the non-blocking behavior that does not affect the sending process.

In contrast, a synchronous communication is based on tightly coupled applications. The calling process depends on the receiver's processing behavior, performance and the environmental issues. The calling process must halt in order to wait for the remote application's response. This results in a blocking type of behavior. Synchronous communication offers a direct request/response communication where the initiating process state depends on the result and state of the invoked application.

Transactions

A major requirement of enterprise applications are transactions. Transactional behavior represents an all or nothing execution of a group of logical operations.

A typical example of a transaction is a money transfer, moving an amount from one account to the other. Although this looks like a single operation it consists of debiting the one account and crediting the other. The whole operation will only succeed if both parts are successfully executed. If one part fails all changes must be rolled back.

Originally transactions have been the domain of database management systems, guaranteeing consistent concurrent multi-step operations. They have become a more general topic in software systems. In integration environments transactions are not limited to one database. They are processed over many distributed databases and application systems.

Scenarios

Based on the parameters mentioned above the following scenarios are in focus:

- Synchronous and non-transactional communication
- Synchronous and transactional communication
- Asynchronous and non-transactional communication
- Asynchronous and transactional communication

This leads to eight scenarios overall, as both hubs may act as the initiating starting point of the communication.

The choice of communication types for a certain integration process depends on business needs, technical needs and performance considerations.

Performance

Each scenario depends on the communication requirements of the involved applications and processes. Furthermore performance needs to be considered in communication scenarios. An important factor is the volume und and periodicity of the transferred data. At this point, only facing the scenarios, two factors are influencing the performance of communication: the overhead impact of transactionality and potential blocking of bi-directional communication.

Transactional communication is more expensive than a straight transmission without. The prevention of possible rollbacks of transactions is expensive. More over this request/response communication may slow down system availability. Synchronous and also asynchronous waiting for a response is allocating significant resources in mass data scenarios. A fire and forget strategy, whereby a message is just transferred to the communication partner without concerning the further processing on the receiver side offers a performing alternative. Both integration platforms handle their incoming messages based on queuing. This results in a decoupling of the internal message handling from the external communication and in a reduction of synchronous communication performance issues.

Transport and Messaging

For SAP-integration scenarios SAP XI uses proxies which are interfaces that are generated in application systems to communicate with the Integration Server. SAP systems based on Web AS 6.40 and higher are able to exchange data by using messages and HTTP. These systems can use proxies to connect to the Integration Server. SAP systems up to and including SAP Release 4.6 are not able to exchange data by using XML messages and http. The only way to connect such systems to the Integration Server directly is by using the IDoc adapter or the RFC adapter. This adapter can also connect non-SAP systems like Microsoft BizTalk Server as sub-systems.

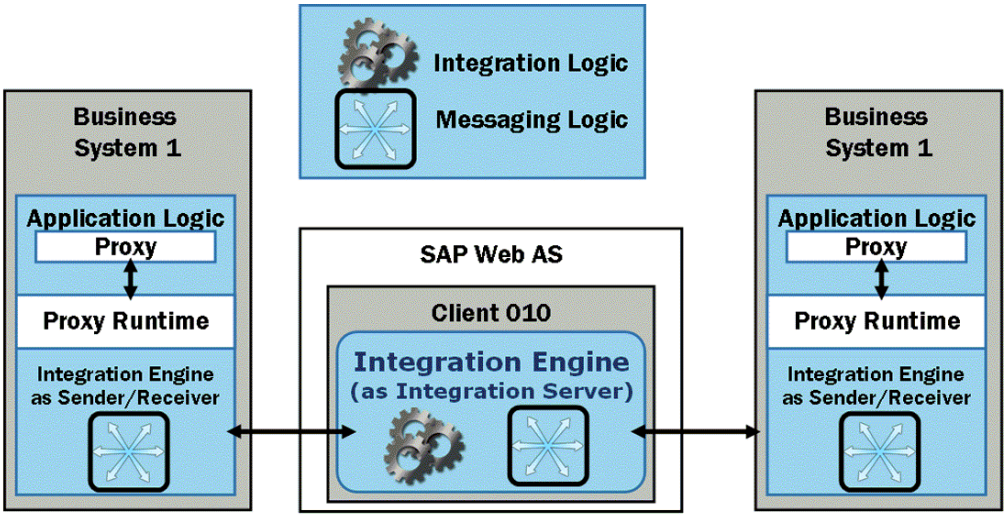

Figure 6 SAP Integration Engine and Proxy Runtime

As proxies require a Web AS they are not appropriate to integrate BizTalk Server and SAP XI. Thus a loosely-coupled and flexible connectivity between BizTalk and SAP XI at transport level involves the adapter layer on both sides. BizTalk and SAP XI include a set of ready to use adapters which will be the basis for the white paper. Beyond it third-party vendors offer additional adapters that require further licensing. Figure 7 shows the standard adapters available for SAP XI and BizTalk integration at transport and messaging level.

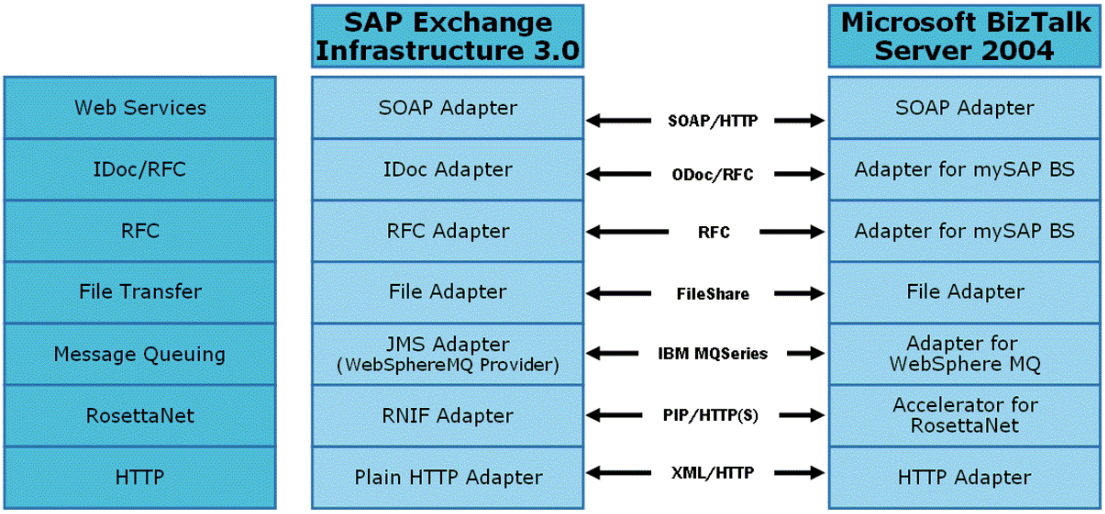

Figure 7 Adapter Architecture

The most obvious choice to connect both hubs is the SOAP adapter using Web Services technologies. The next chapters will point out advantages and several limitations of this approach. Alternative adapters, relevant especially for asynchronous, transactional and mass data integration scenarios, will be discussed.

Web Services

A common way of data exchange in integration environments are Web Services. Web Services provide message exchange between applications based on open standards and protocols. They are used as a platform and program language independent data communication over computer networks in a manner similar to inter-process communication on a single computer. Web Services standards are maintained by OASIS (Organization for the Advancement of Structured Information Standards, http://www.oasis-open.org/) and the W3C committee (http://www.w3c.org).

As a result of interoperability problems between implementations of the above standards the Web Services Interoperability Organization (WS-I) (http://www.ws-i.org/) was founded. It is an open industry consortium of over 170 software vendors, enterprise customers and system integrators focusing on comprehensive interoperability of Web Services. On of the most notable results of their work is the release of the 'Basic Profile 1.0', a documented profile defining a detailed specification for a Web Service based on the existing standards SOAP 1.1, WSDL 1.1, UDDI 2.0, XML 1.0, XML Schema and HTTP 1.1. They are provided together with hints on potential interoperability issues and usage recommendations. This profile is the basis of vendor independent Web Services interoperability.

SAP XI and BizTalk both offer a tight integration of Web Services into the engine and development tools. Using the SAP XI SOAP Adapter enables the consumption and publication of Web Services based on the SOAP 1.1 protocol according to the WS-I Basic Profile 1.0.

The BizTalk SOAP Adapter also offers Web Services support based on SOAP 1.1 support and Basic Profile 1.0 capabilities.

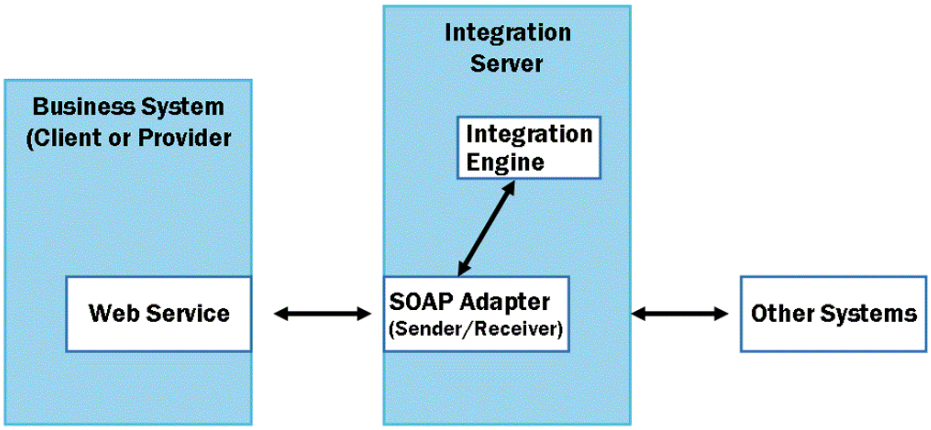

Figure 8 SOAP Adapter

Web Services according to the WS-I basic profile provide a synchronous-oriented transfer of data without additional Quality of Service (QoS) features.

Currently several new standardizations are in progress addressing different aspects of enterprise needs on data exchange, e.g. security, transactionality, reliability or

asynchronous messaging. A set of these new standards is defined in WS-* architecture containing Web Services Security (WSS), WS-ReliableMessaging and WS-Transaction.

The Web Services Enhancements (WSE) v 2.0 (http://msdn.microsoft.com/webservices/building/wse/) add WSS and other WS-* capabilities to Microsoft .NET and BizTalk. SAP XI itself implements vendor specific QoS features using special SOAP-headers.

The upcoming communication infrastructure of Microsoft called Indigo and the next version of SAP NetWeaver will support most of the WS-* standards.

In a real life scenario of an interoperation between BizTalk 2004 and SAP XI 3.0 these new technologies don't really matter due to limited support on the one hand and missing interoperability standards on the other.

All possible extensions to add session management for asynchronous communication, quality of service (e.g. delivery once) or transactional behavior must be done programmatically.

An example of implementing guaranteed delivery message transfer using Web Services and BizTalk Framework 2.0 SOAP extensions between XI and BizTalk is described in a future document of CTSC.

Limitations

Web Services features in data exchange between BizTalk and SAP XI have to be reduced to the most common level. Therefore transactions and asynchronous communication are not available out of the box.

Additionally performance is always important in communication scenarios especially using Web Services due to data overhead and parsing of XML. Mass data transfer through Web Services is still a complex topic.

IDoc/tRFC

The Remote Function Call (RFC) is the SAP standard interface for communication between systems (SAP or non-SAP). tRFC (Transactional RFC) is an asynchronous RFC supporting a transaction protocol with two-phase-commit.

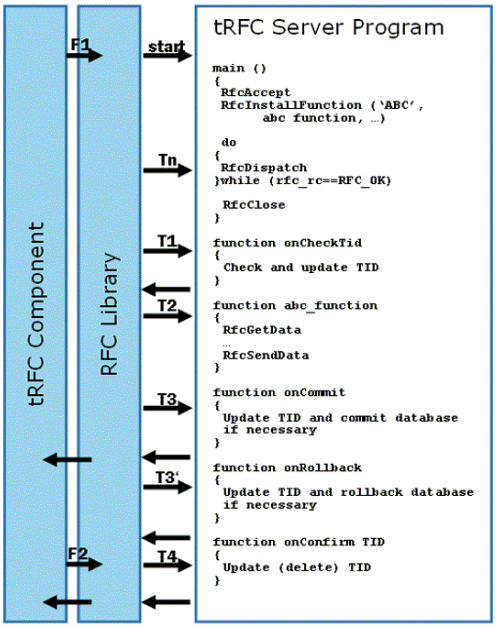

Figure 9 tRFC call

The Intermediate Document (IDoc) format is a SAP message format of data exchange with other systems. Each IDoc generated exists as a self-contained text file that can then be transmitted to the requesting workstation without connecting to the central database. IDoc-communication with tRFC is the standard transport- and message format for asynchronous communication between SAP-systems and SAP and non-SAP systems. IDoc types define the message format for different business scenarios. Beyond standard IDoc types (e.g. for invoicing (INVOICE), ordering (ORDERS), etc.) customized types are common.

Although IDoc communication is discussed as legacy, the installation base of SAP systems today requires the usage of this technology. For SAP systems up to and including SAP Release 4.6 IDoc communication is the recommended way of integration with SAP XI (see SAP XI Documentation).

BizTalk 2004 and SAP XI 3.0 both use adapters for a transparent communication over IDoc/tRFC. The internal format of the integration systems is based on XML and XSD. SAP XI uses the IDoc adapter, BizTalk the Adapter v2.0 for mySAP Business Suite, both supporting the sending and receiving IDocs.

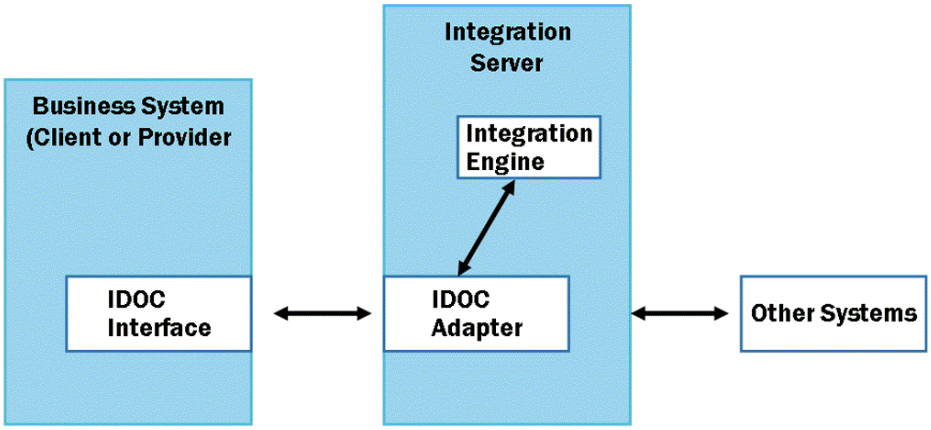

Figure 10 IDoc Adapter

The SAP XI IDoc Adapter maintains its own meta data repository with IDoc types. As a result the IDoc communication is decoupled from the backend systems.

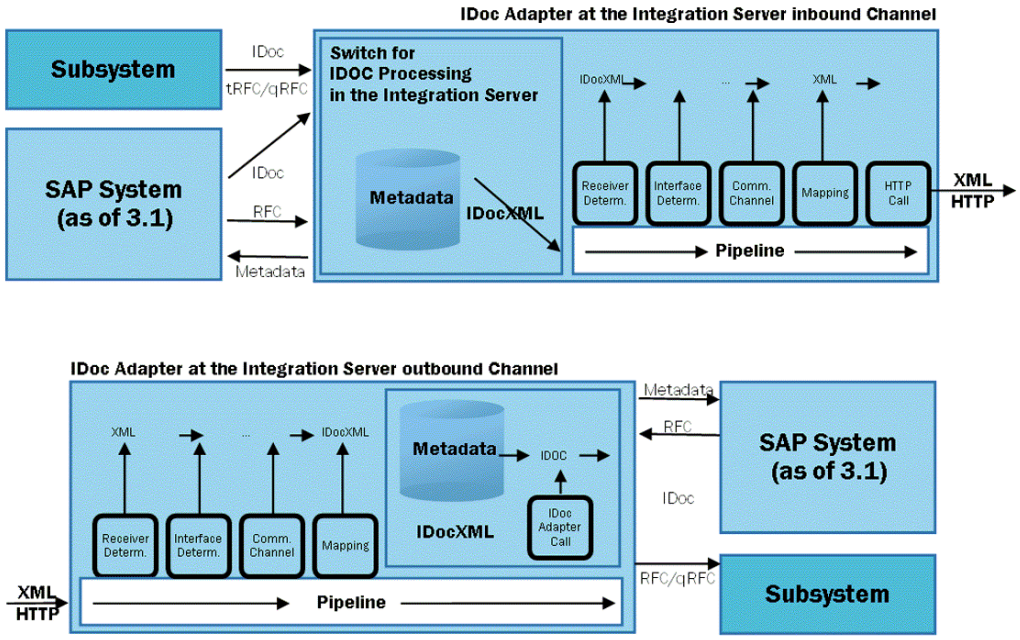

Figure 11 SAP XI IDoc Adapter

The BizTalk Adapter v2.0 for mySAP Business Suite enables IDoc exchange between SAP systems and BizTalk. It includes meta data access by a schema generator.

Beyond integrating native SAP systems both IDoc adapters can be used for the communication between the two integration systems.

IDoc communication offers a robust, well proven way of transactional asynchronous communication. With the background processing option within SAP mass data inbound scenarios with high scalability are possible.

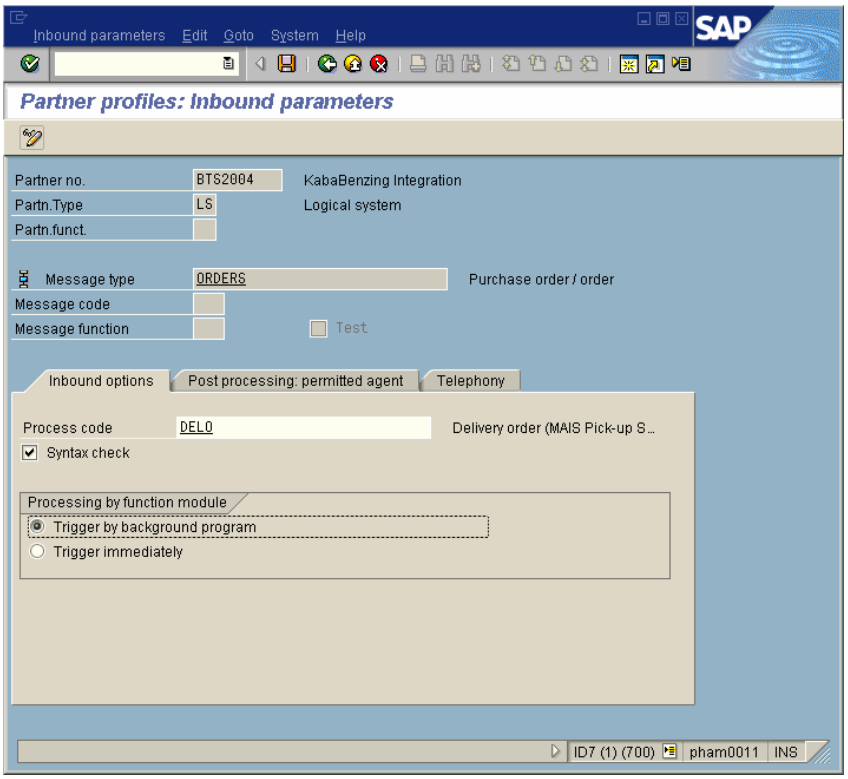

Figure 12 Background processing of IDocs

RFC

The Remote Function Call is an SAP API driven way to integrate systems with SAP function components. In contrast to transactional RFC (see chapter IDoc/tRFC) plain RFC provides a synchronous and non-transactional communication.

SAP XI supports RFC through the adapter engine with the RFC Adapter. It converts RFC messages to the internal XML format and vice versa.

BizTalk server using the BizTalk Adapter v2.0 for mySAP Business Suite is able to act as RFC-sender. It includes meta data access to a BOR (SAP Business object repository) and schema generation.

Limitations

SAP XI does not provide a meta data repository for RFC. The only way to transfer this metadata is a direct connection to the underlying mySAP/R3 system.

Message Queuing

A Message queue is a software component providing an asynchronous communication infrastructure.

Common message queuing systems function as a middleware for inter-process and inter-application communication and implement transactions and several QoS-features (e.g. reliability/guaranteed delivery, ordered delivery). Beyond point-to-point message transfer publish/subscribe communication is common. This model defines two roles, a publisher as the producer of messages for a context, and a subscriber as the consumer of these messages subscribing to this context. One or more publishers may send messages to a context. With more than one registered subscribers this results in a many-to-many communication.

SAP XI supports the usage of message queuing by the JMS Adapter. JMS (Java Message System) is an abstraction layer for messaging systems. Common commercial messaging systems are integrated through vendor specific JMS providers.

BizTalk uses vendor specific adapters to connect to message queuing systems. The BizTalk MSMQ Adapter and the Microsoft BizTalk Adapter for MQSeries v2.0 are available free of charge. Furthermore third-party adapters exist.

A possible integration scenario of Microsoft BizTalk 2004 and SAP XI 3.0 using message queuing is shown below.

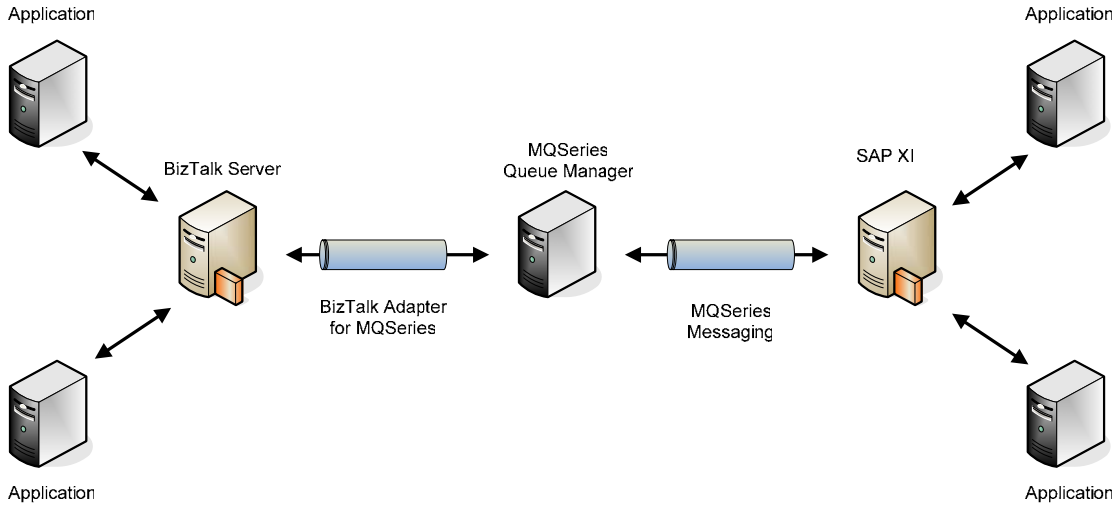

Figure 13 Microsoft BizTalk Adapter for MQSeries

The messaging infrastructure is based on MQSeries with the according queue manager. BizTalk connects to queue managers using the MQSeries adapter. The SAP XI system uses the JMS Adapter configured with the MQSeries JMS provider. This scenario allows asynchronous communication with or without transactions.

Limitations

Integrating both hubs using a common message queuing product adds an additional communication layer to the infrastructure. It is based on a vendor specific engine and

message format. Therefore a mapping to this format is required as well as a dedicated operating, monitoring and maintenance approach. Furthermore a common way of meta data exchange is not established.

Although adapters and/or providers for SAP XI and BizTalk Server may be free of charge additional licenses for the messaging product are required. This scenario should be considered in environments with an established message queuing infrastructure.

XML over HTTP

Besides Web Services a possible communication scenario is the exchange of XML payload over HTTP without an enclosing SOAP envelope. The sender acts as an HTTP-client transferring the payload as MIME-type text/xml via HTTP-request. The receiver is a web- (namely HTTP-) server, reading and consuming the data and if necessary responding also with XML payload.

This way of communication covers a synchronous message transfer without transaction support avoiding the overhead of SOAP data.

SAP XI provides the Plain-HTTP Adapter for this XML-data exchange. It covers inbound and outbound scenarios. BizTalk with the accompanied HTTP adapter enables data exchange with XML payload in both directions.

As SAP XI does not support chunked encoding with HTTP it is necessary to disable this feature in BizTalk. Microsoft knowledge base article 839663 describes this issue (http://support.microsoft.com/kb/839663/en-us).

Limitations

This scenario is pretty similar to plain Web Services communication over the HTTP protocol. Anyhow this communication using XML over HTTP is not standardized at messaging level. Furthermore there is no standardized way for schema exchange between the systems analog to WSDL.

Others

File transfer

The transfer of files between both hubs over a file system represents the most straight forward way of inter-application communication. It allows asynchronous fire-and-forget communication scenarios based on various message formats.

BizTalk and SAP XI both offer a FILE adapter to send and receive files.

The free choice of the underlying message format implies a lack of meta data exchange. File transfer additionally requires a dedicated communication control and monitoring to guarantee message delivery and reliability (e.g. clustered shares) preventing the loss of messages.

B2B industry standards

Beyond open standards different established ways of data communication or “industry standards” like RosettaNet and EDIFACT could be a possible solution to integrate BizTalk and SAP XI. The communication mechanisms and QoS-support depend on the certain standard. Also compatible adapters with the same support set have to be available for SAP XI and BizTalk (e.g. BizTalk Accelerator for RosettaNet and SAP XI RNIF Adapter).

The common limitation of all these potential standards is their proprietary. All data transferred using such a standard has to be converted into the propriety format and transformed to none-standard messages. Additional licensing on both sides for the specialized adapters is necessary in most cases (e.g. BizTalk Accelerator for RosettaNet).

Walkthrough

The following sections describe a technical walkthrough on how to establish the integration of BizTalk and XI from a technical perspective. All scenarios are based upon the demo examples SAP delivers with XI 3.0. The demo examples configuration is described in a document available at the SAP marketplace (https://service.sap.com/xi, menu path SAP Exchange Infrastructure\Media Library\Documentation).

The scenarios require the “General configuration steps” according to the document’s descriptions. These configuration steps define the system landscape in SLD, perform the client copy to the model agency and the airlines and configure the airline systems enabling a connection to XI.

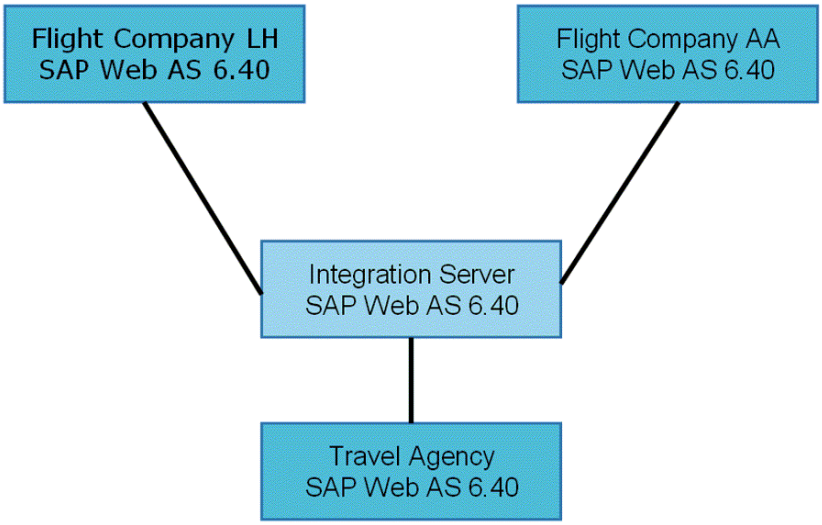

Figure 14 SAP XI integration scenario demo

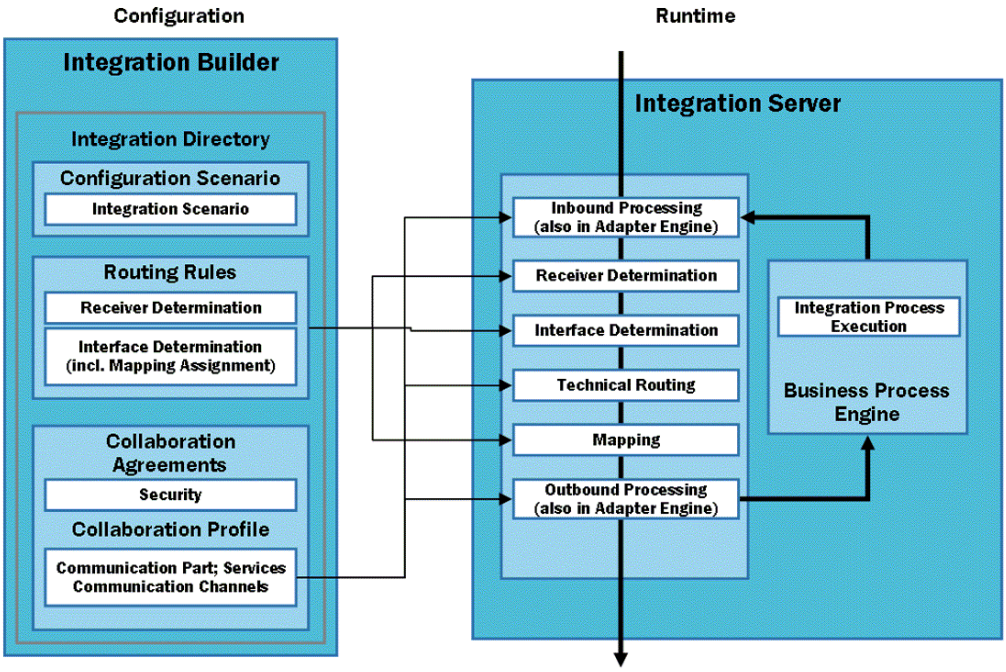

Figure 15 Scenario architecture

The walkthrough scenarios do not make use of the business process engine. They are based on the simple approach where a message is received by XI, mapped to a destination schema and forwarded to the endpoint.

Scenario 1: Synchronous Integration – SAP XI Web Service Provider

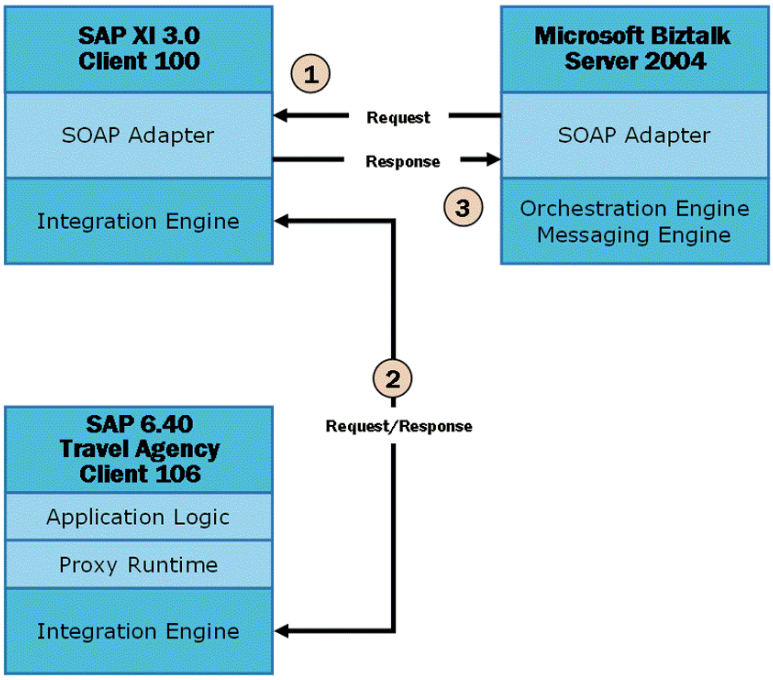

Figure 16 SAP XI as Web Service Provider

Introduction

The underlying business case of this scenario describes a travel agency checking the flight availability in an airline system. It is based on the CheckFlightSeatAvailability setting of the SAP XI examples.

In this case SAP XI provides a Web Service that connects to the airline system (1). The connection to the airline system is done by a RFC (2). The XI WebService is consumed by BizTalk which simulates the agency's system (3).

SAP XI - Integration Builder (Design)

XI defines internal scenarios as high level description of system interaction. These scenarios describe the data flow between systems and will be used for a semi automatic generation of configuration objects in the Integration Builder (Configuration). This walkthrough focuses on the CheckFlightSeatAvailability scenario as shown in the figure below. No further development of objects is required in the Integration Builder (Design).

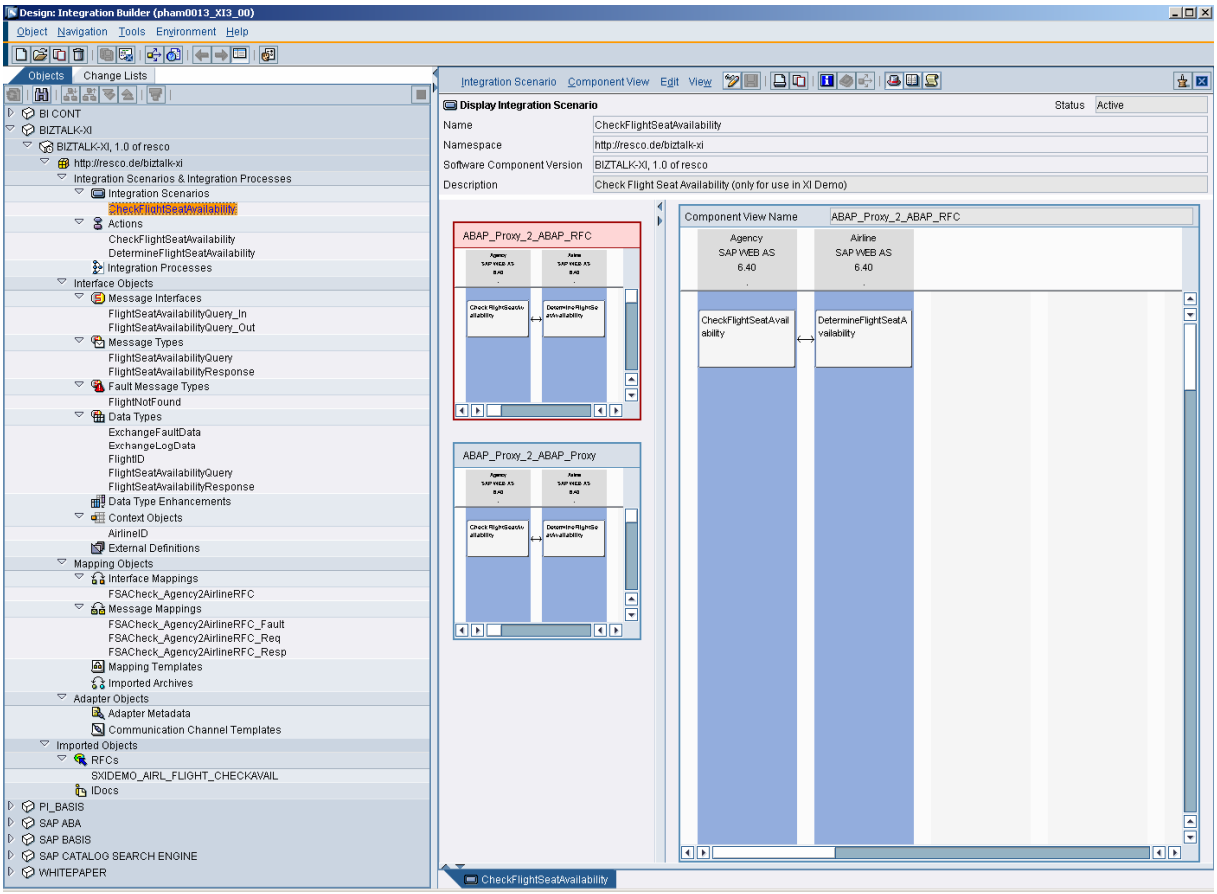

Figure 17 Integration scenario CheckFlightSeatAvailability

BizTalk calls an XI Web Service based on the message interface FlightSeatAvailabilityQuery_Out. The interface is a synchronous outbound message interface based on a query defining several input values (airline identifier, connection and flight date) and a response returning the available seats from the airline.

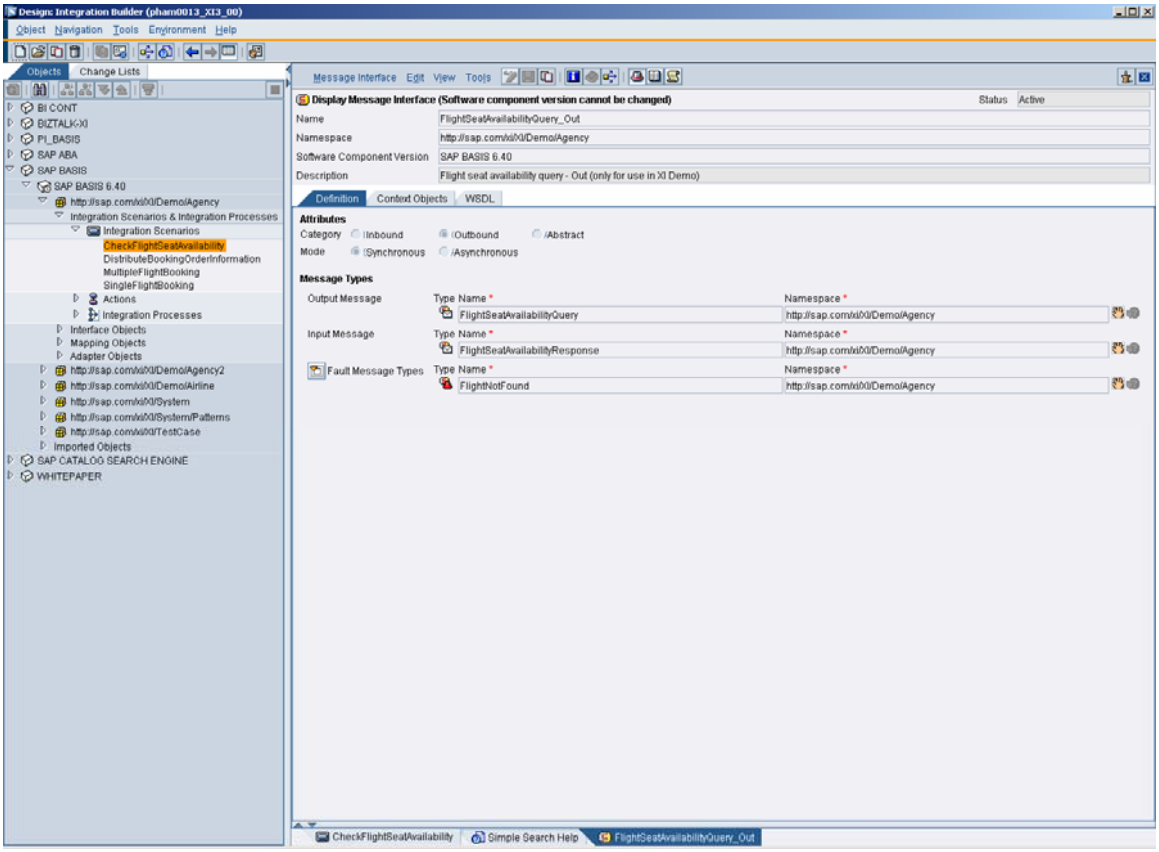

Figure 18: Message interface FlightSeatAvailabilityQuery_Out

- Figure 18 shows the XI inbound message used by the agency (simulated by BizTalk Server) to submit an availability request.

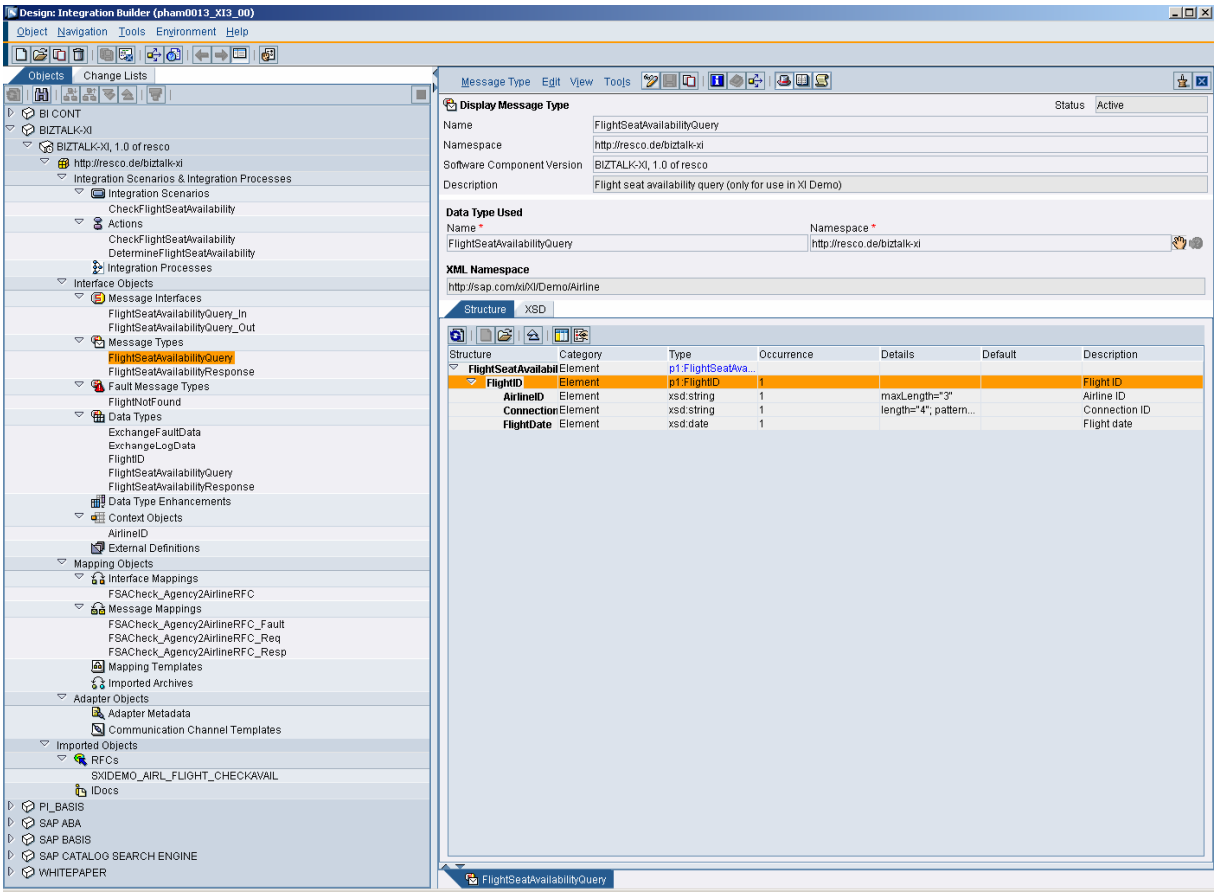

Figure 19 Message type FlightSeatAvailabilityQuery

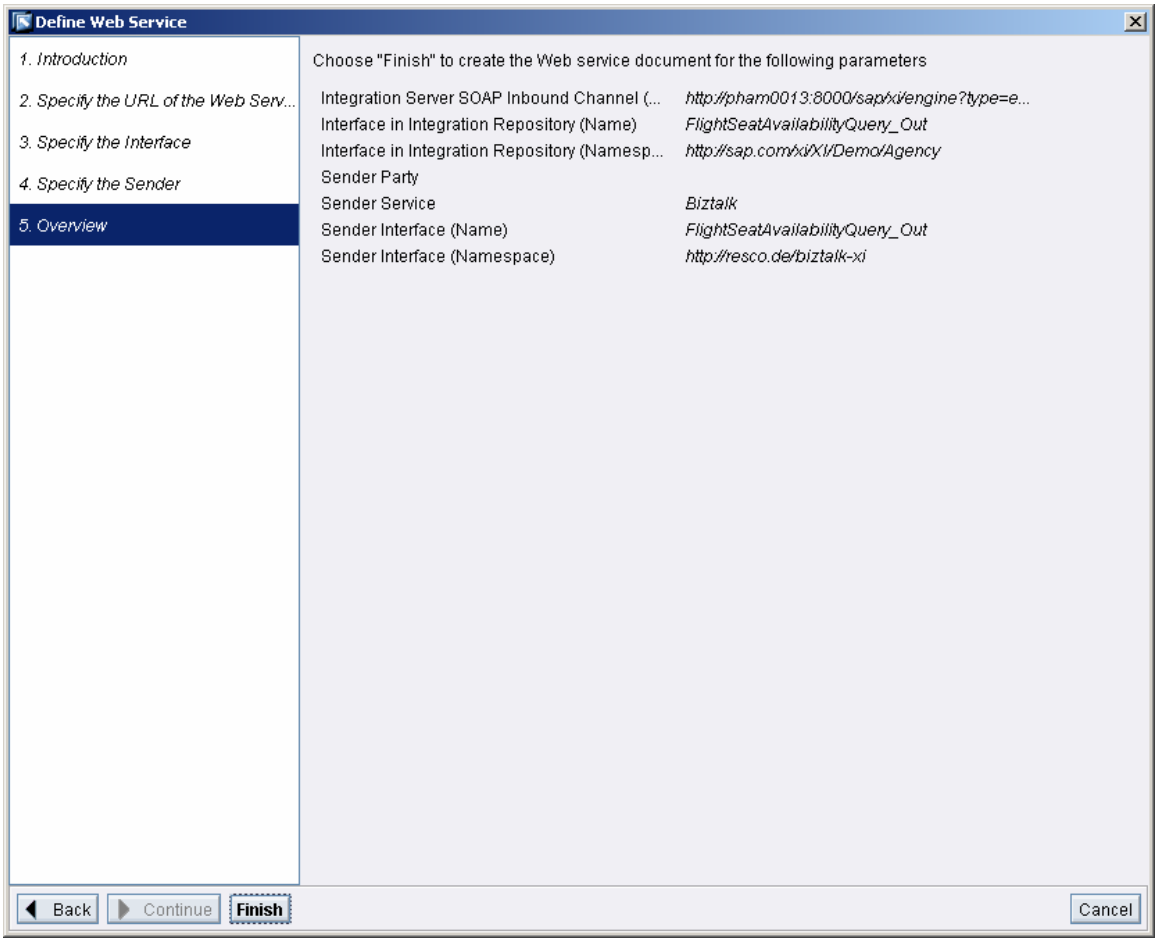

Figure 20 WSDL generation for message interface

- The WSDL is generated using the WebService generation wizard for the message interface FlightSeatAvailabilityQuery_Out. The generated file is used within BizTalk Server / Visual Studio .NET as the source of the Web Service consumption.

SAP XI - Integration Builder (Configuration)

All specific port configurations are handled using the Integration Builder (Configuration).

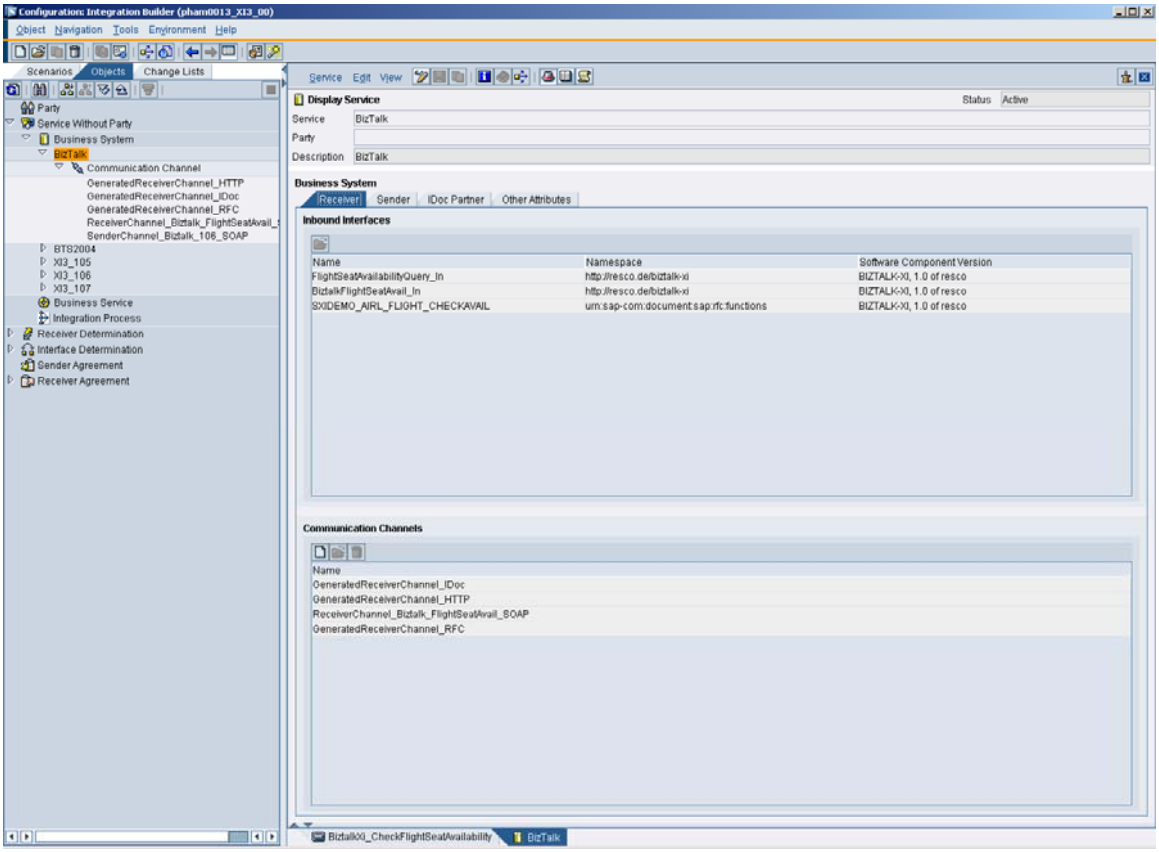

Figure 21 Definition of the business service BizTalk

- Figure 21 shows the definition of a new business system/service BizTalk.

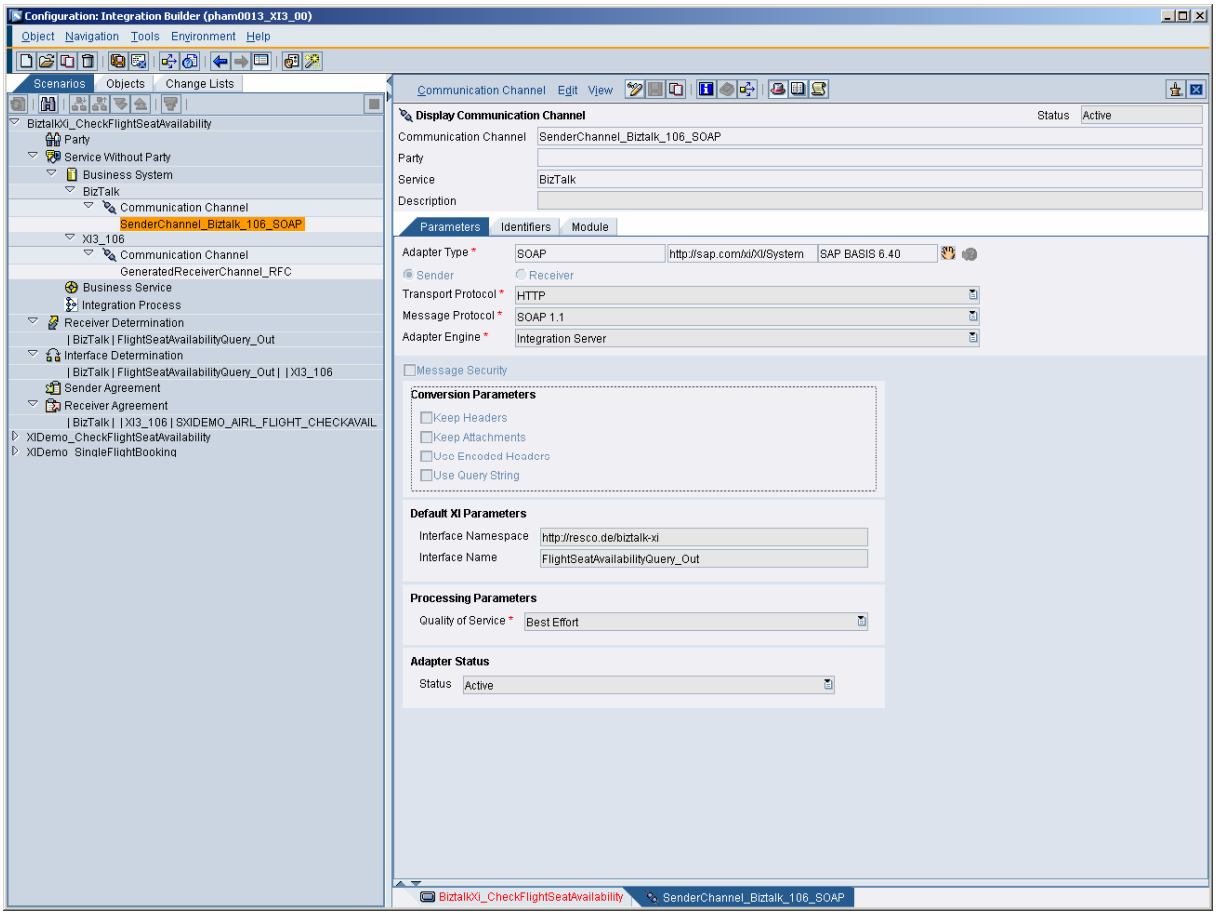

Figure 22 Configuration of the sender SOAP channel

- For incoming Web Services requests a communication channel of type SOAP Sender is configured. SOAP sender defines the incoming message type as well as the processing parameters. The BizTalk Web Services client uses this parameters (service and communication channel). This enables XI to assign messages to the corresponding integration scenario.

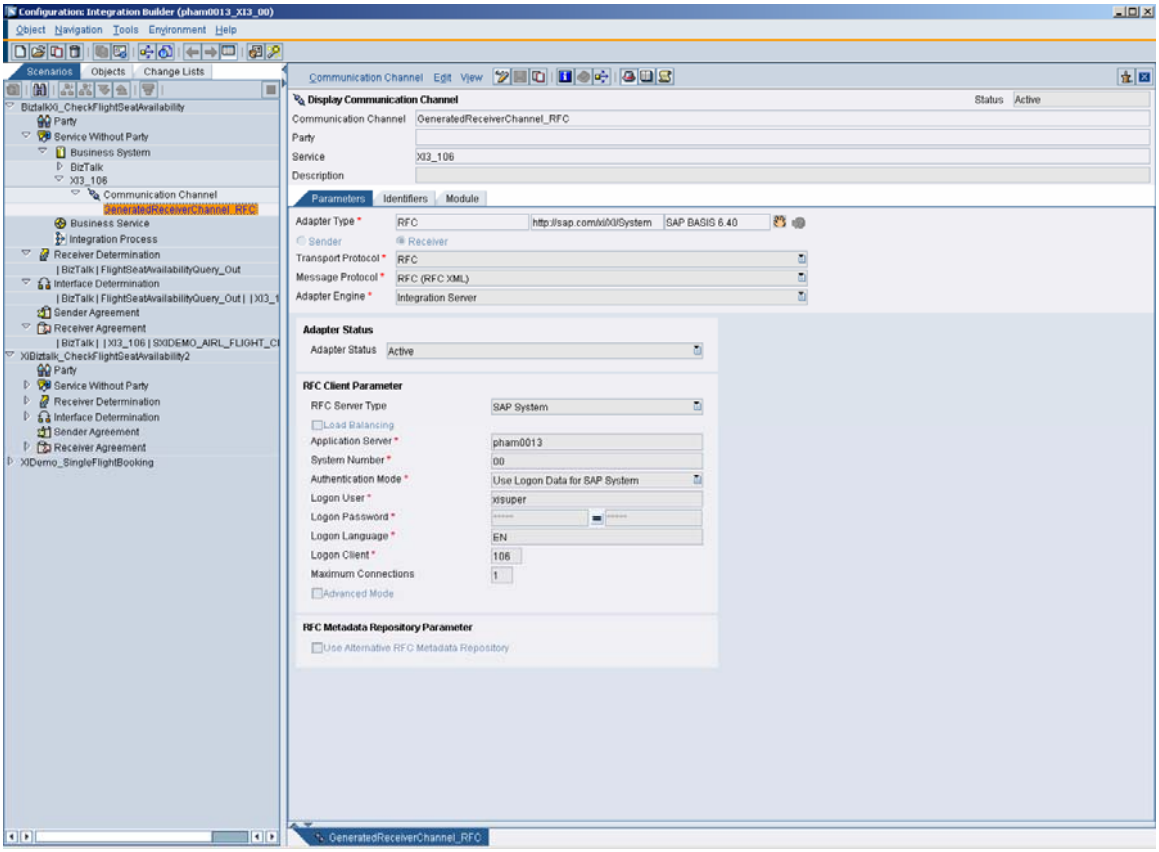

Figure 23 Configuration of the reveiver RFC channel

- The connectivity to the SAP airlines system is configured using a communication channel of type RFC receiver. The base settings of this configuration are generated during the creation of the service XI3_106 based on the settings in the SLD. The communication channel contains the logon and password information.

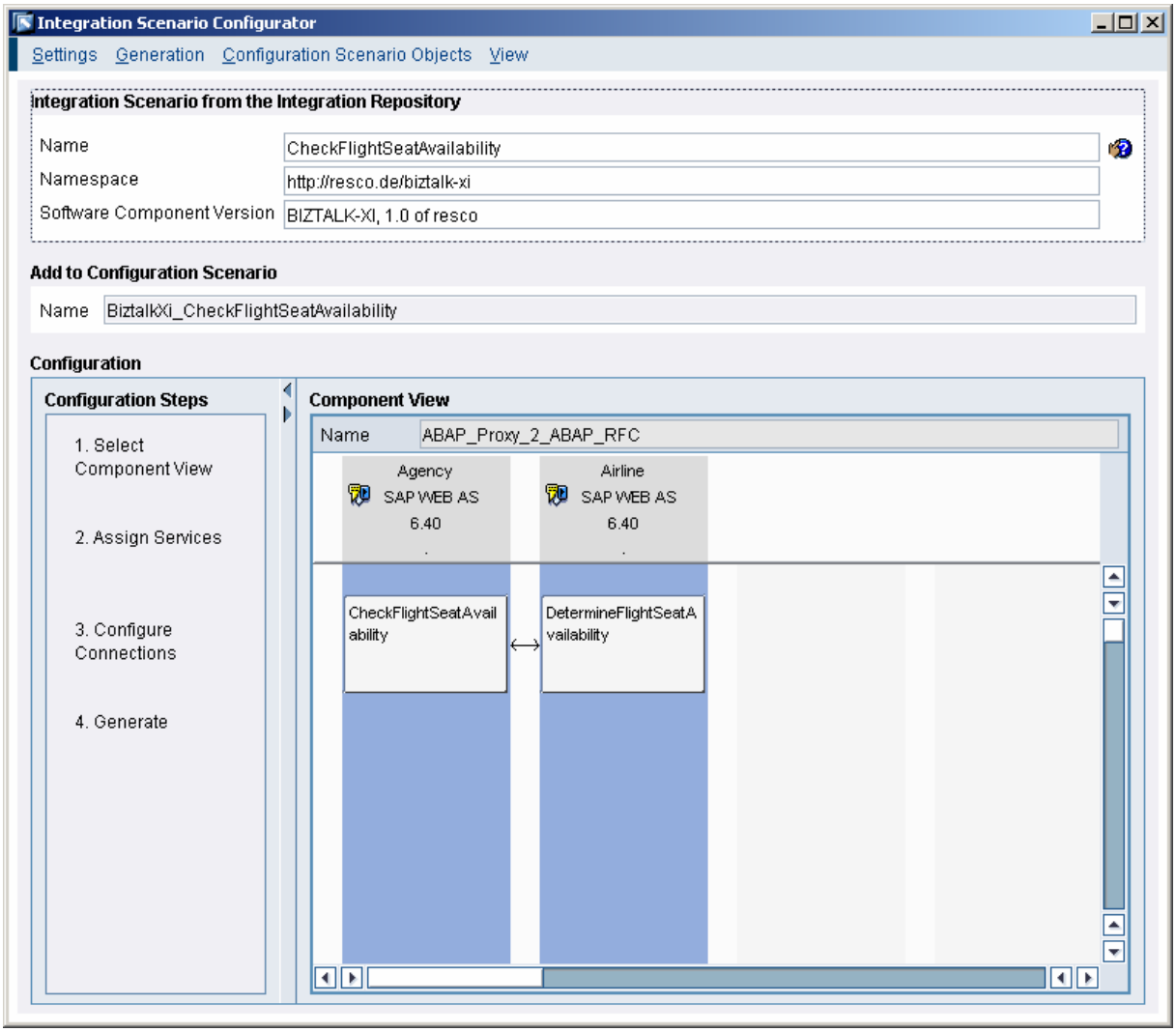

Figure 24 Configuration of the XI scenario

- The scenario CheckFlightSeatAvailability is imported into the Integration Builder (Configuration). The configuration is done with assistance of the Integration scenario configurator. The configuration requires three steps:
  - **Selection of the component view:**
    The scenario offers two alternatives to contact the airline system (connectivity via proxy or RFC). The ABAP_Proxy_2_RFC is used in here.
  - **Assignment of services:**
    In this scenario BizTalk acts as the travel agency and XI3_106 as the airline.
  - **Configuration of the connections:**
    The connection to the airline system is defined using RFC by selecting the communication channel GeneratedReceiverChannel_RFC.

- The generation is started based on the settings above. This results in the following objects, which are executed successively at runtime according to the XI pipeline approach:

- **Receiver determination:**
  Sender service: BizTalk
  Interface: FlightSeatAvailabilityQuery_Out
  Configured receiver: XI3_106
- **Interface determination** triggers the mapping of the incoming message to the parameters of the RFC function module in the airline system and vice versa
  Sender service: BizTalk
  Interface: FlightSeatAvailabilityQuery_Out
  Receiver: XI3_106
  Inbound interface: SXIDEMO_AIRL_FLIGHT_CHECKAVAIL
  Interface mapping: FSACheck_Agency2AirlineRFC
- **Receiver agreement** binds an inbound interface to a receiver communication channel
  Sender service: BizTalk
  Receiver: XI3_106
  Interface: SXIDEMO_AIRL_FLIGHT_CHECKAVAIL
  Receiver communication channel: GeneratedReceiverChannel_RFC

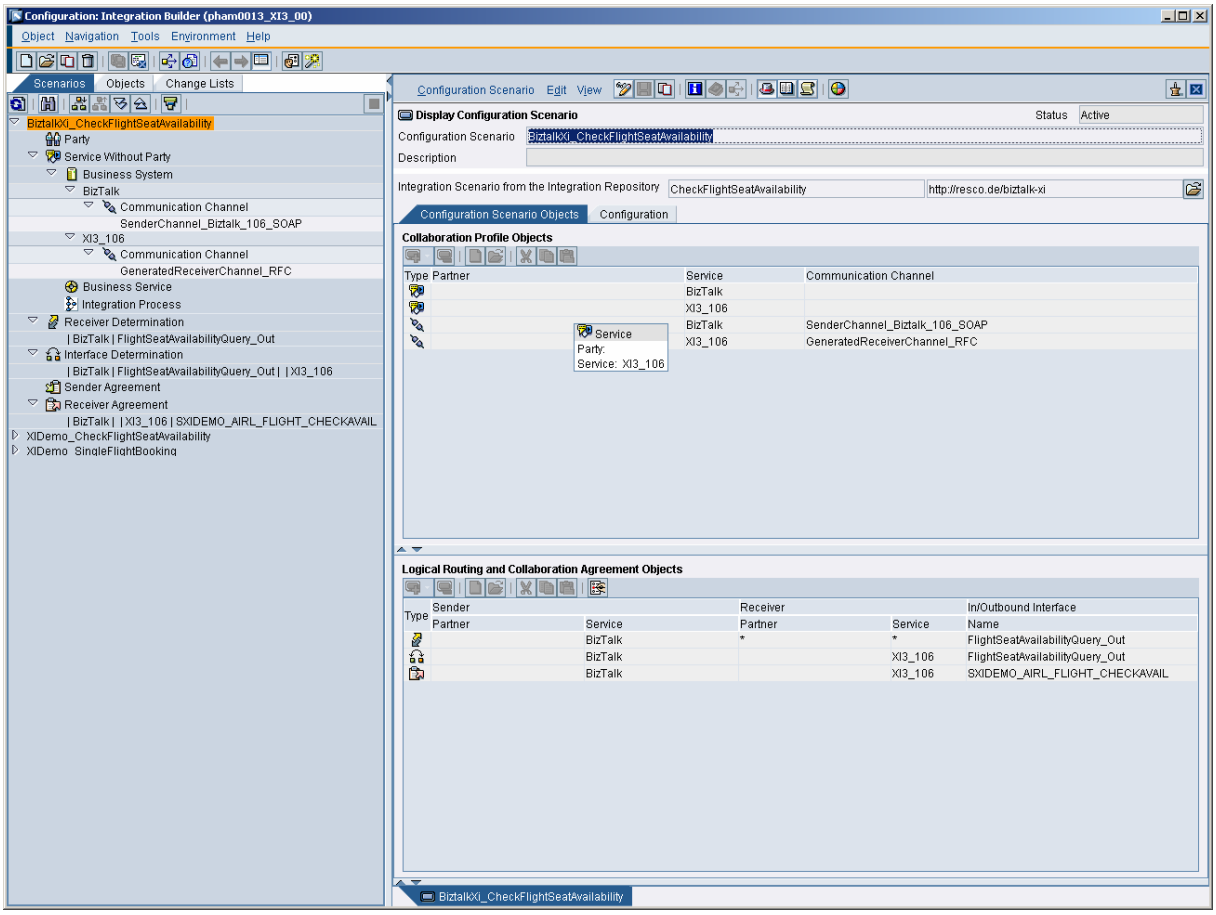

Figure 25 Configured integration scenario

- Using SAP XI scenarios is a good approach to get a quick overview which configuration objects belong to a integration scenario.

Microsoft BizTalk 2004

The BizTalk Server process is based on an orchestration. The orchestration receives the request from the agency, calls the XI Web Service and responds to the agency.

This scenario focuses on the interoperability of both integration systems. Therefore the data exchange with the agency is based on XML files using the Web Service's payload schema. The data is directly transferred to the Web Service without any mapping.

At first a web reference is added to a BizTalk project (named BizTalk_SAP_XI).

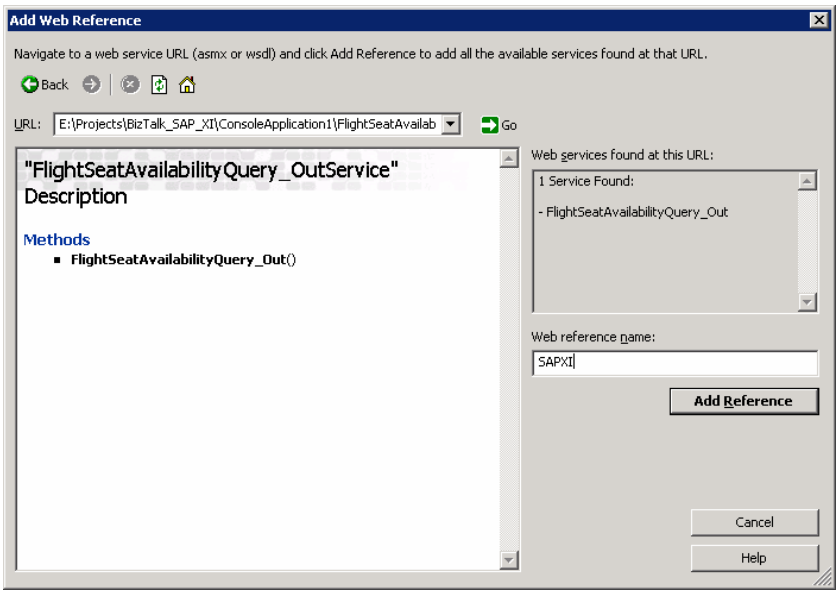

Figure 26 A web reference is added to the BizTalk project

- The location of the generated WSDL of XI on the file system is entered as URL. SAP does not provide the WSDL over network connections.

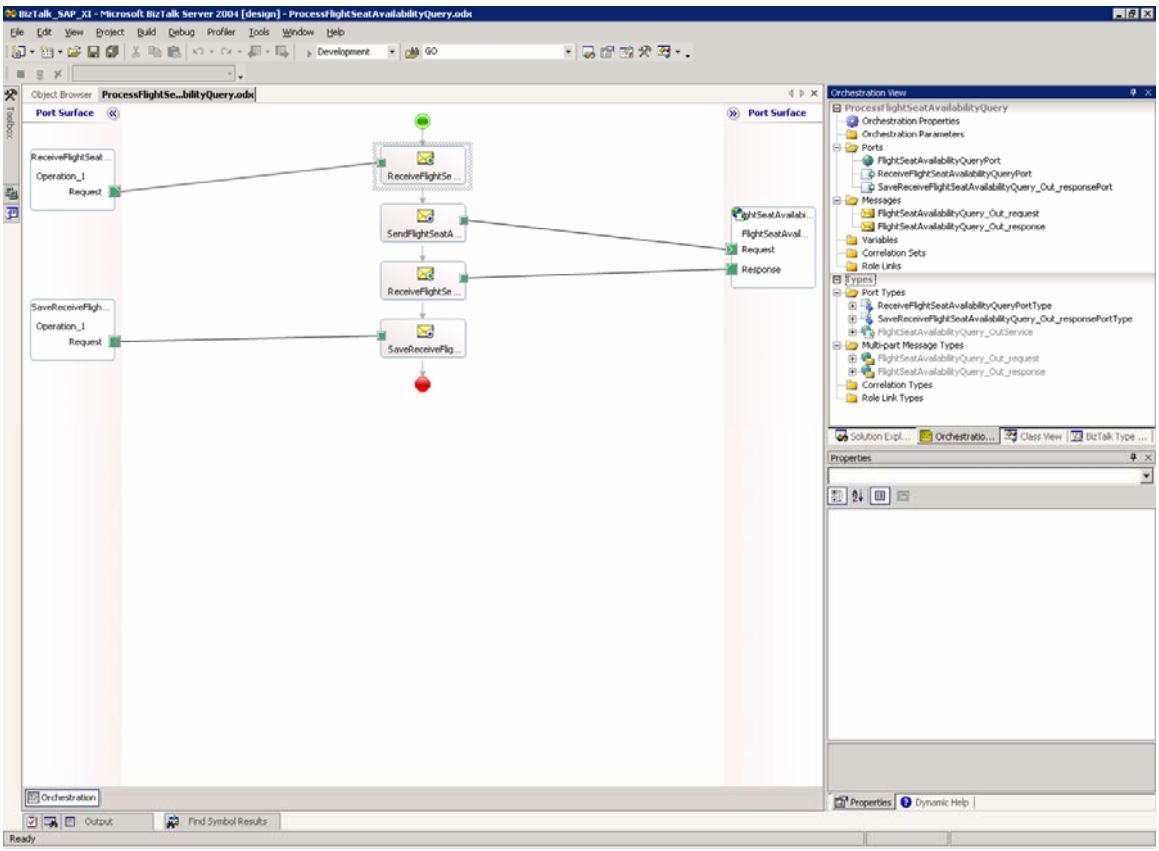

Figure 27 BizTalk orchestration consuming an XI Web Service

- Creation of an orchestration with the following types:

Messages

- FlightAvailabilityQuery_out_request of type BizTalk_SAP_XI.SAPXI.FlightSeatAvailabilityQuery_OutService_.FlightSeatAvailabilityQuery_Out_request, representing the availability query request
- FlightAvailabilityQuery_out_response of type BizTalk_SAP_XI.SAPXI.FlightSeatAvailabilityQuery_OutService_.FlightSeatAvailabilityQuery_Out_response, representing the availability query request

Message Receiver

- ReceiveFlightSeatAvailabilityQuery with message FlightAvailabilityQuery_out_request, receiving the request from the agency
- ReceiveFlightSeatAvailabilityQuery_Out_response with message FlightAvailabilityQuery_out_response, receiving the query response from the webservice

Message Sender

- FlightSeatAvailabilityQuery_Out_request with message FlightAvailabilityQuery_out_request, sending th query message to the Web Service
- SaveReceiveFlightSeatAvailabilityQuery_Out_response with message FlightAvailabilityQuery_out_response sending the response to the agency

Port types

- ReceiveFlightSeatAvailabilityQueryPortType with an operation using the web message type BizTalk_SAP_XI.SAPXI.FlightSeatAvailabilityQuery_OutService_.FlightSeatAvailabilityQuery_Out_request as request
- SaveReceiveFlightSeatAvailabilityQuery_Out_responsePortType with an operation using the web message type BizTalk_SAP_XI.SAPXI.FlightSeatAvailabilityQuery_OutService_.FlightSeatAvailabilityQuery_Out_response as request
- FlightSeatAvailabilityQuery_Out_response_PortType automatically generated with the Web Service reference
- SaveReceiveFlightSeatAvailabilityQuery_Out_responsePort with an operation using message type BizTalk_SAP_XI.SAPXI.FlightSeatAvailabilityQuery_OutService_.FlightSeatAvailabilityQuery_Out_response as response

Ports

- ReceiveFlightSeatAvailabilityQueryPort with port type BizTalk_SAP_XI.ReceiveFlightSeatAvailabilityQueryPortType
- FlightSeatAvailabilityQueryPort with port type BizTalk_SAP_XI.FlightSeatAvailabilityQueryPortType
- SaveReceiveFlightSeatAvailabilityQuery_Out_responsePort with port type BizTalk_SAP_XI.SaveReceiveFlightSeatAvailabilityQuery_Out_responsePortType

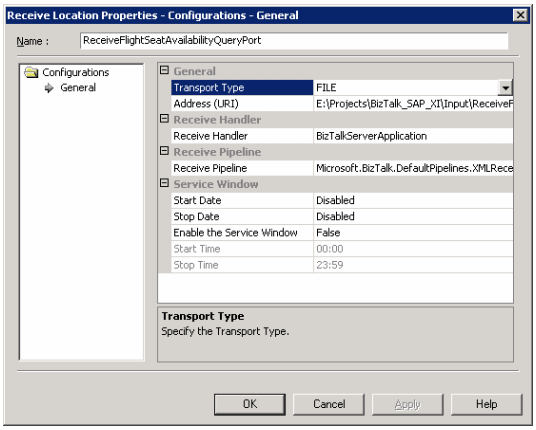

Figure 28 Receive Location for agency availability request files

- Configuration of the receive port and location ReceiveFleightSeatAvailabilityQueryPort for the request files from the travel agency

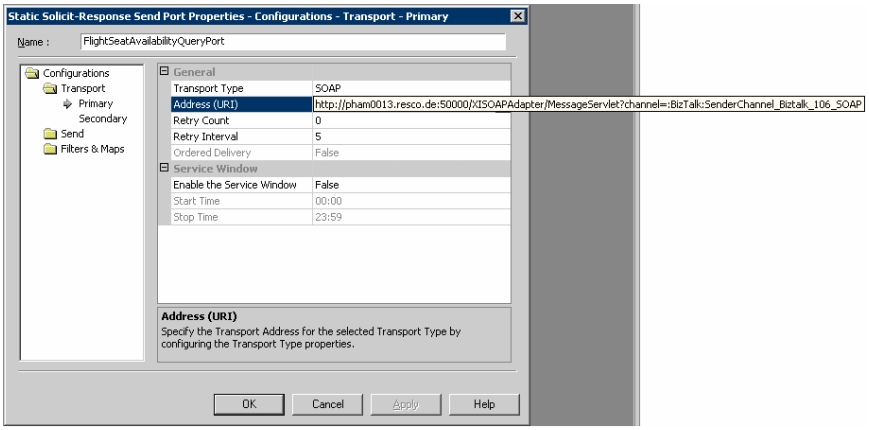

Figure 29 Solicit response send port for Web Services communication

- Figure 29 shows the configuration of a solicit response port FleightSeatAvailabilityQueryPort calling the XI Web Service.

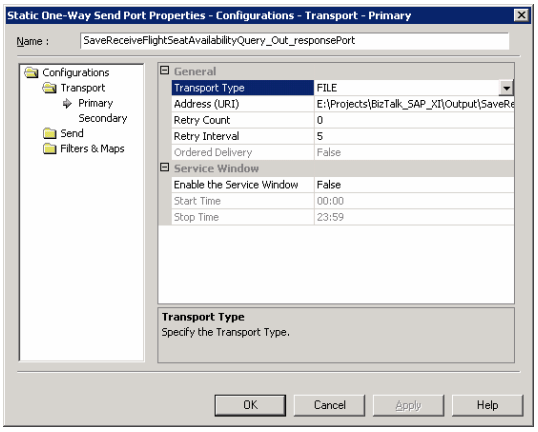

Figure 30 File send port for Web Service response messages

- Configuration of the file send port SaveReceiveFleightSeatAvailabilityQuery_Out_responsePort for the response files containing the availability data.

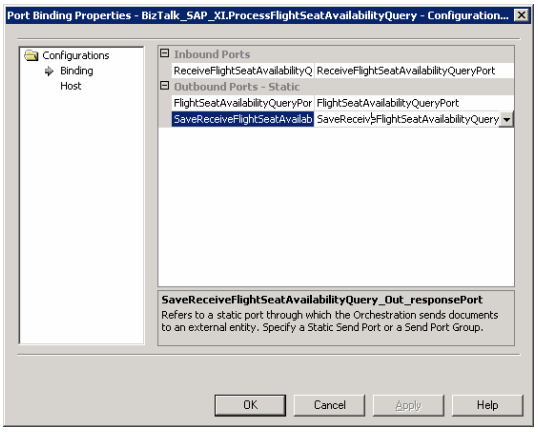

Figure 31 Orchestration binding

- Figure 31 shows the configuration of the port bindings containing the binding of the logical orchestration ports to the physical ports and locations.

Initiating and Monitoring

In order to start of the scenario an XML-file matching the schema FlightSeatAvailabilityQuery_Out_request of BizTalk is required. Using the “generate instance” of the schema’s context menu in Visual Studio creates the appropriate file. The fields AirlineID, ConnectionID and FlightDate must be filled with appropriate business data (see Figure 32).

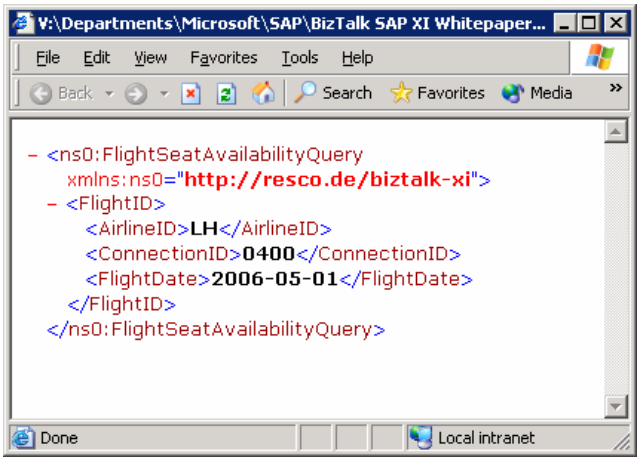

Figure 32 Flight availability request XML file

The scenario is started by dropping this XML file into the configured path of the BizTalk Receive Location ReceiveFlightSeatAvailabilityQueryPort (e.g. E:\Projects\BizTalk_SAP_XI\Input\ReceiveFlightSeatAvailabilityQueryPort).

As result of a successful call BizTalk creates an output XML-file in the directory of the File Send Port SaveReceiveFleightSeatAvailabilityQuery_Out_responsePort. This file contains the availability information (see Figure 33).

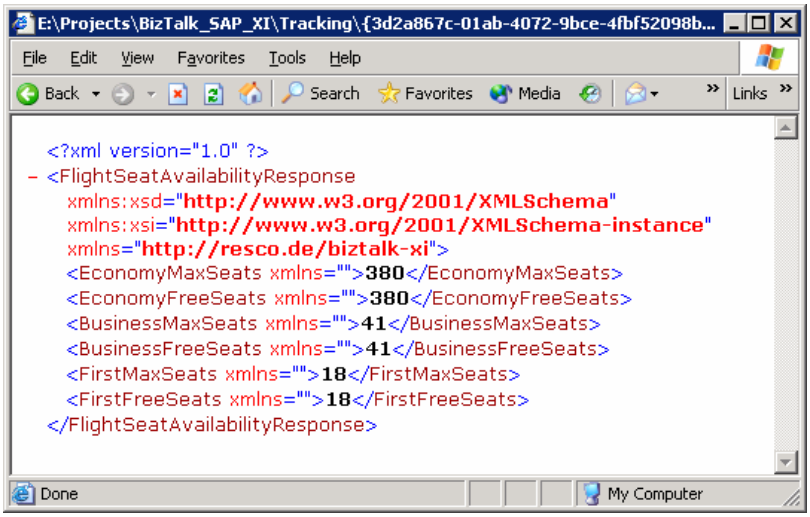

Figure 33 Flight availability response XML file

Scenario 2: Synchronous Integration – BizTalk Web Service Provider

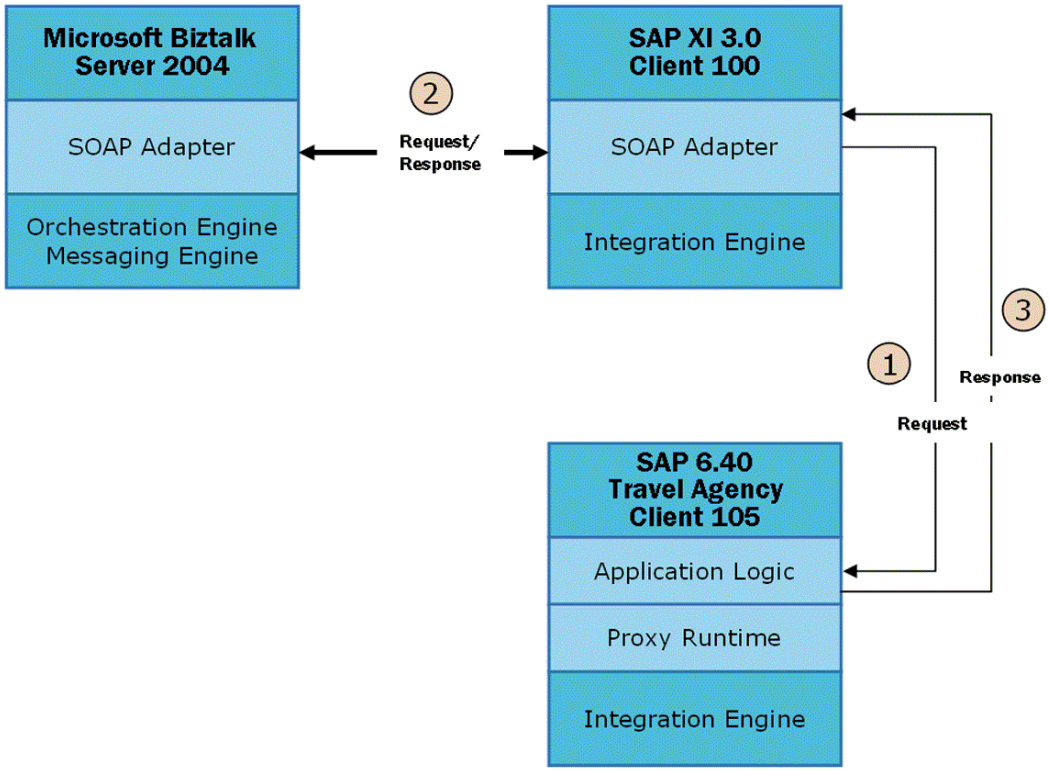

Figure 34 BizTalk Server as Web Service Provider

Introduction

In this scenario CheckFlightSeatAvailability is used once again but the roles have been switched. The agency SAP XI3_105 requests the flight seat availability from XI (1). XI forwards the request to BizTalk using the SOAP adapter (2). BizTalk simulates the airline and returns the SOAP response that is dispatched from XI to the agency client (3). This service is accessible via the WebGui of the demo examples (transaction SXIDEMO). The agency XI3_105 is not aware of underlying implementation changes.

SAP XI - Integration Builder (Design)

BizTalk provides a WSDL describing the service CheckFlightSeatAvailability of the airline. This WSDL-file is imported into XI called “external definition”. This external definition can be used like any other message interface within XI.

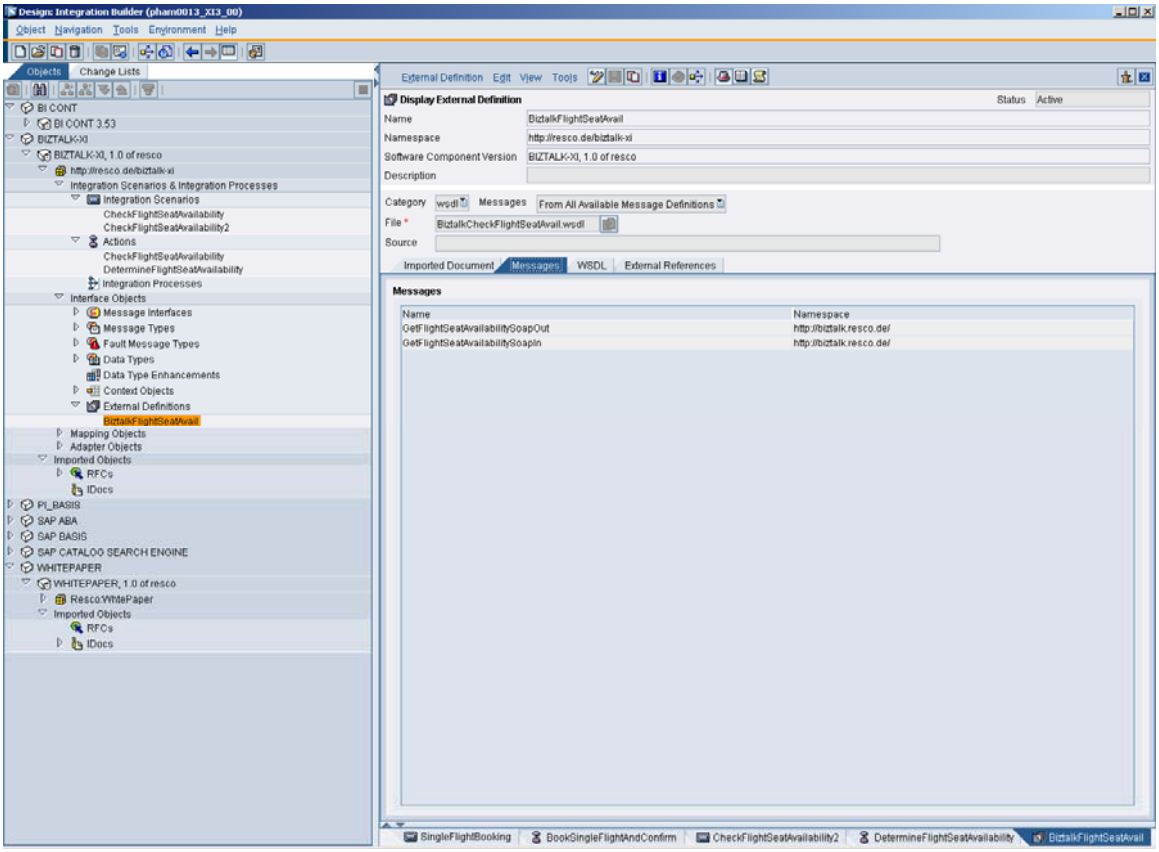

Figure 35 External definition of the imported BizTalk Web Service

- The SAP integration scenario CheckFlightSeatAvailability is copied into a separate namespace. The action DetermineFlightSeatAvailability is extended with the message interface BiztalkFlightSeatAvail_In received via WSDL.

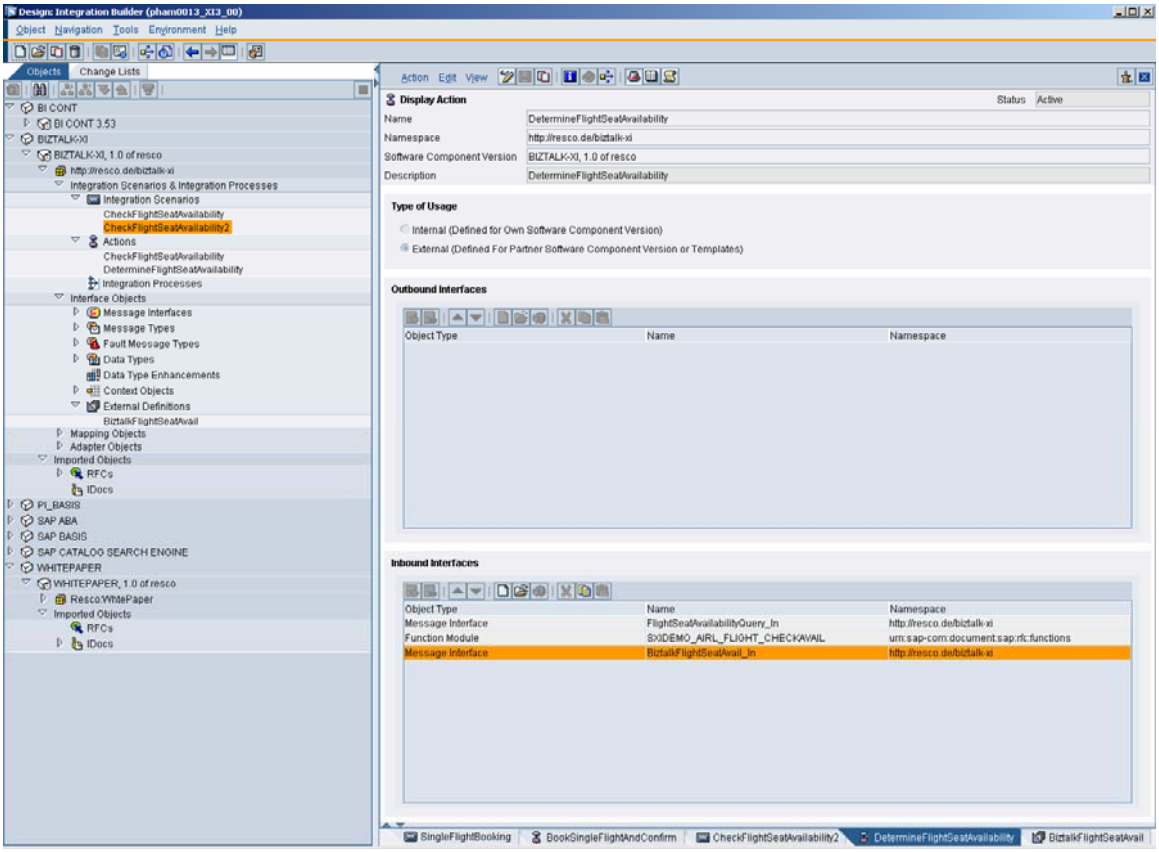

Figure 36 Design of the action within Integration Builder

- Figure 36 shows configuration of the link between the two actions in the scenario. The BizTalk based message is used.

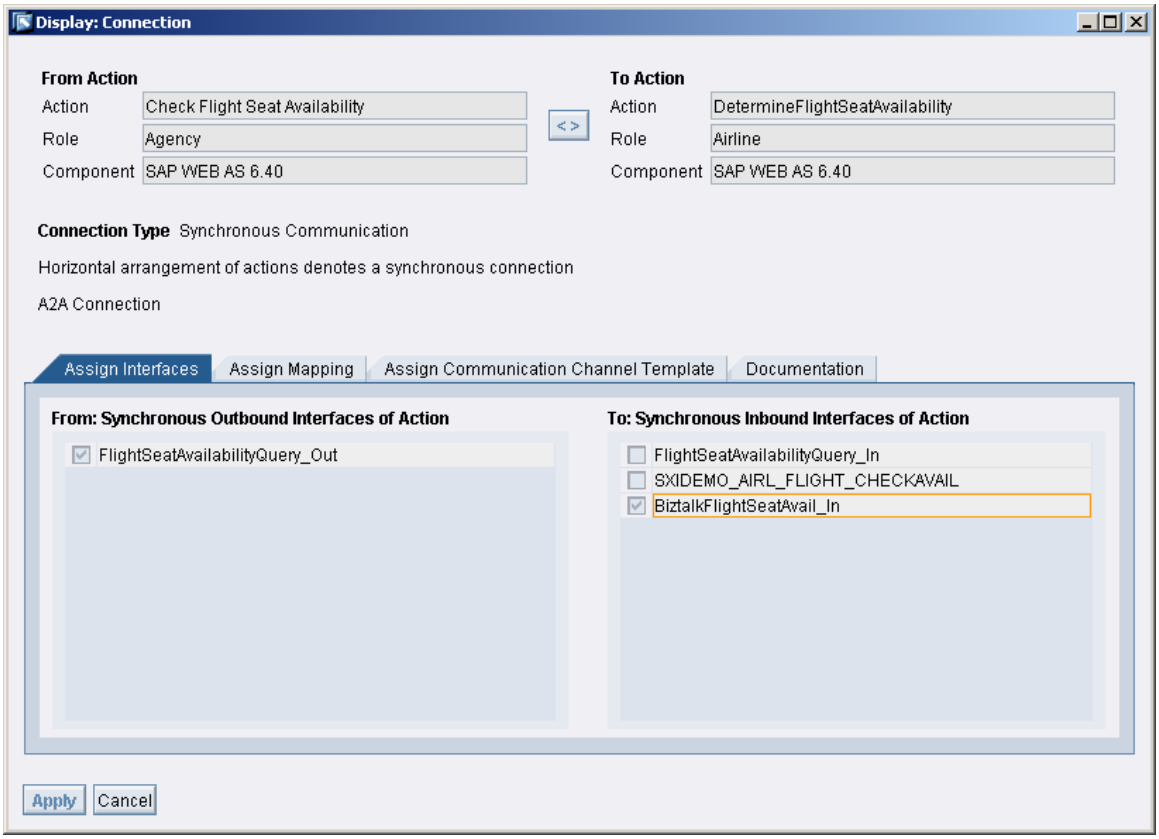

Figure 37 Interface mapping

- An interface mapping is defined. The source interface FlightSeatAvailabilityQuery_Out as sent outbound by the agency XI3_105 to the Airline interface BiztalkFlightSeatAvail_In.
- Two maps are defined mapping requests and responses:
  - FlightSeatAvailabilityQuery_GetFlightSeatAvailabilitySoapIn maps the request from XI3_105 to the WebService message type.
  - GetFlightSeatAvailabilitySoapOut FlightSeatAvailabilityResponse maps the response from the BizTalk WebService to the response format as expected by the agency XI3105.

SAP XI - Integration Builder (Configuration)

The scenario is imported into the Integration Builder (Configuration). The configuration is done using the Integration scenario configurator.

- **Receiver determination**
  Sender service: XI3_105
  Interface: FlightSeatAvailabilityQuery_Out
  Configured receiver: BizTalk
- **Interface determination**
  Sender service: XI3_105
  Interface: FlightSeatAvailabilityQuery_Out
  Receiver: BizTalk

Inbound interface: BiztalkFlightSeatAvail_In
Interface mapping: FlightSeatAvailability_BiztalkFlightSeatAvail

- **Receiver agreement**
  Sender service: BizTalk
  Receiver: XI3_106
  Interface: SXIDEMO_AIRL_FLIGHT_CHECKAVAIL
  Receiver communication channel: GeneratedReceiverChannel_RFC

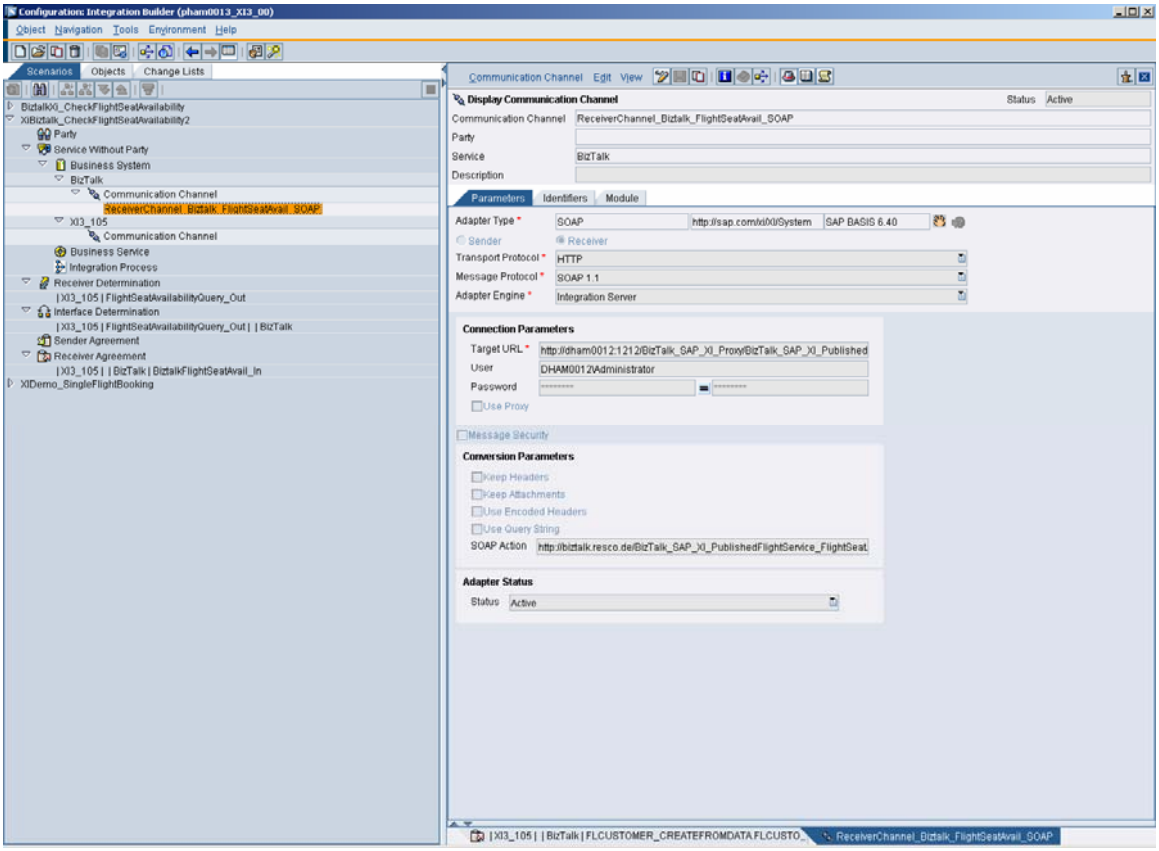

Figure 38 Configuration of the SOAP receiver channel for the BizTalk WebService

The field SOAP-Action is mandatory for Web Services interoperability with BizTalk. The username of the connection parameters must be supplied with a prefixed definition of the used domain or machine (e.g. machine\username).

Microsoft BizTalk 2004

BizTalk Server provides a mechanism to publish orchestrations as Web Services. Therefore a new orchestration PublishedFlightService is added to the BizTalk project.

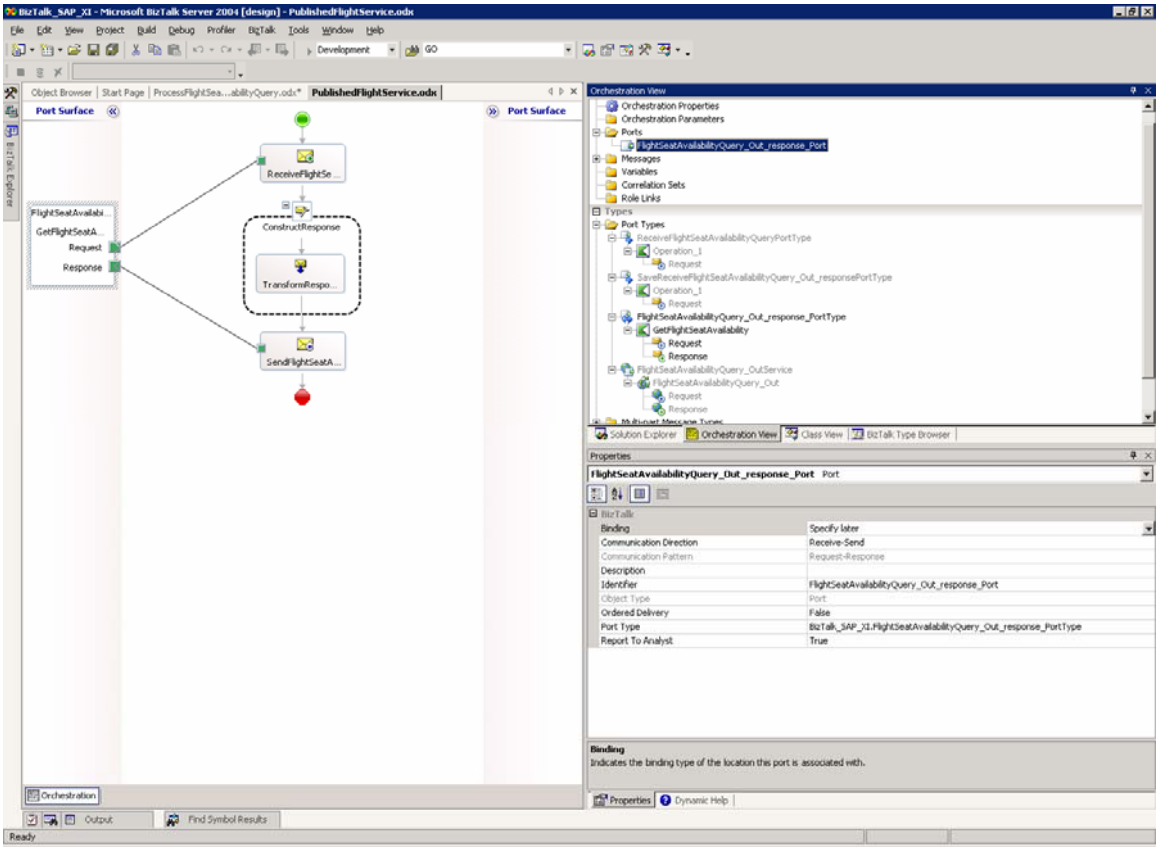

Figure 39 Orchestration being published as a Web Service

The orchestration uses the following types:

- Messages
  - FlightSeatAvailabilityQuery_Out_request of type BizTalk_SAP_XI.SAPXI.FlightSeatAvailabilityQuery_OutService_.FlightSeatAvailabilityQuery_Out_request
  - FlightSeatAvailabilityQuery_Out_response of type BizTalk_SAP_XI.SAPXI.FlightSeatAvailabilityQuery_OutService_.FlightSeatAvailabilityQuery_Out_response

- PortType
  - FlightSeatAvailabilityQuery_Out_response_PortType with a operation with message type BizTalk_SAP_XI.SAPXI.FlightSeatAvailabilityQuery_OutService_.FlightSeatAvailabilityQuery_Out_request as request and message type BizTalk_SAP_XI.SAPXI.FlightSeatAvailabilityQuery_OutService_.FlightSeatAvailabilityQuery_Out_response as response

- Port
  - FlightSeatAvailabilityQuery_Out_response_Port of type FlightSeatAvailabilityQuery_Out_response_PortType

- Message Receiver
  - ReceiveFlightSeatAvailabilityQuery_Out_request with message FlightSeatAvailabilityQuery_Out_request bound to FlightSeatAvailabilityQuery_Out_response_Port request

- Message Sender
  - SendFlightSeatAvailabilityQuery_Out_response with message FlightSeatAvailabilityQuery_Out_response bound to FlightSeatAvailabilityQuery_Out_response_Port response

- Transform
  - TransformResponse using the map FlightSeatAvailabilityQuery_Out_request_response and FlightSeatAvailabilityQuery_Out_request as input message and FlightSeatAvailabilityQuery_Out_response as output message.
    The map simply creats the destination message without any value mapping (Figure 40). All required attributes are set to a constant value of 5.

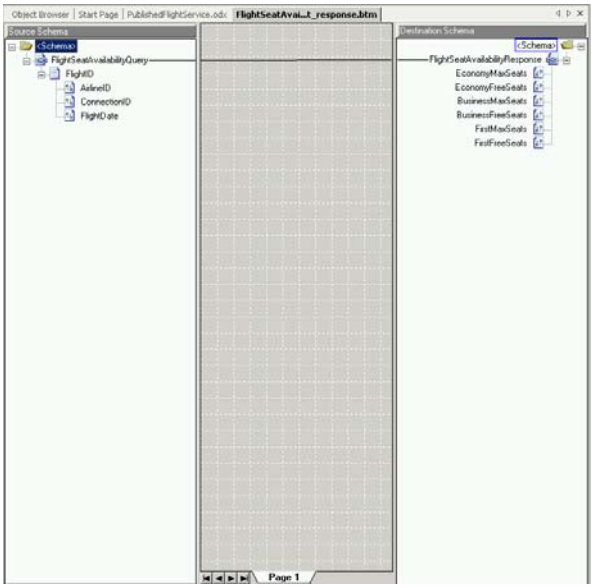

Figure 40 Map creating the response message

- Message constructor
  - Construct response creating message FlightSeatAvailabilityQuery_Out_response containing the transform shape TransformResponse

- The orchestration is published as a Web Service using the BizTalk Web Services publishing wizard. The identity of the application pool of the virtual directory the Web Service is published to must be a member of the BizTalk administration group.

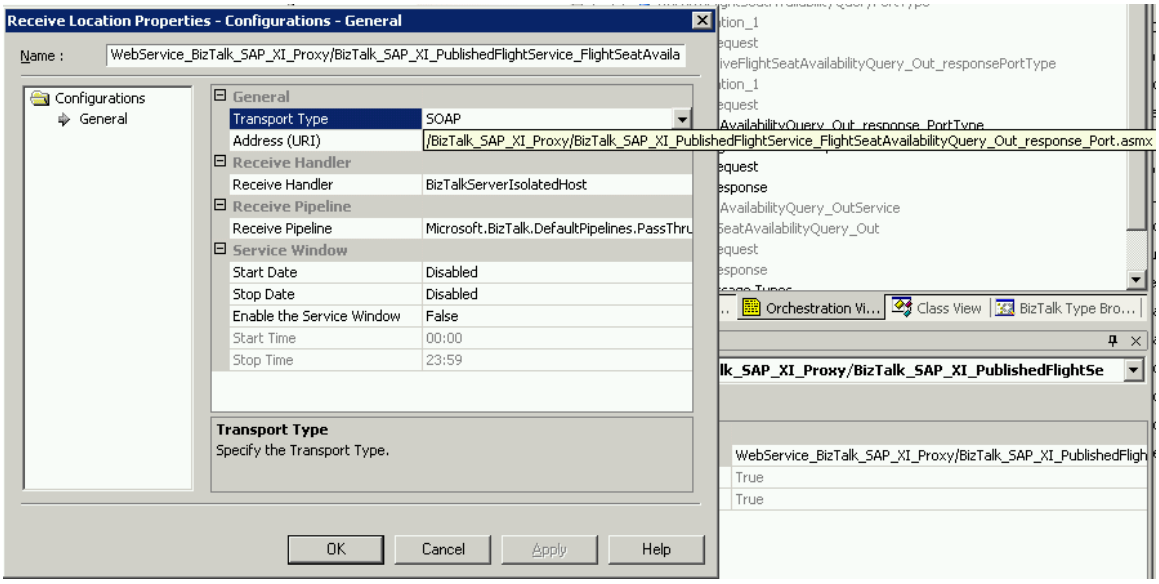

Figure 41 Web Service receive location

- Figure 41 shows the configured Receive Location with the URL of the published Web Service

Initiating and Monitoring

The scenario is started using the web interface of the SAP XI Demo examples.

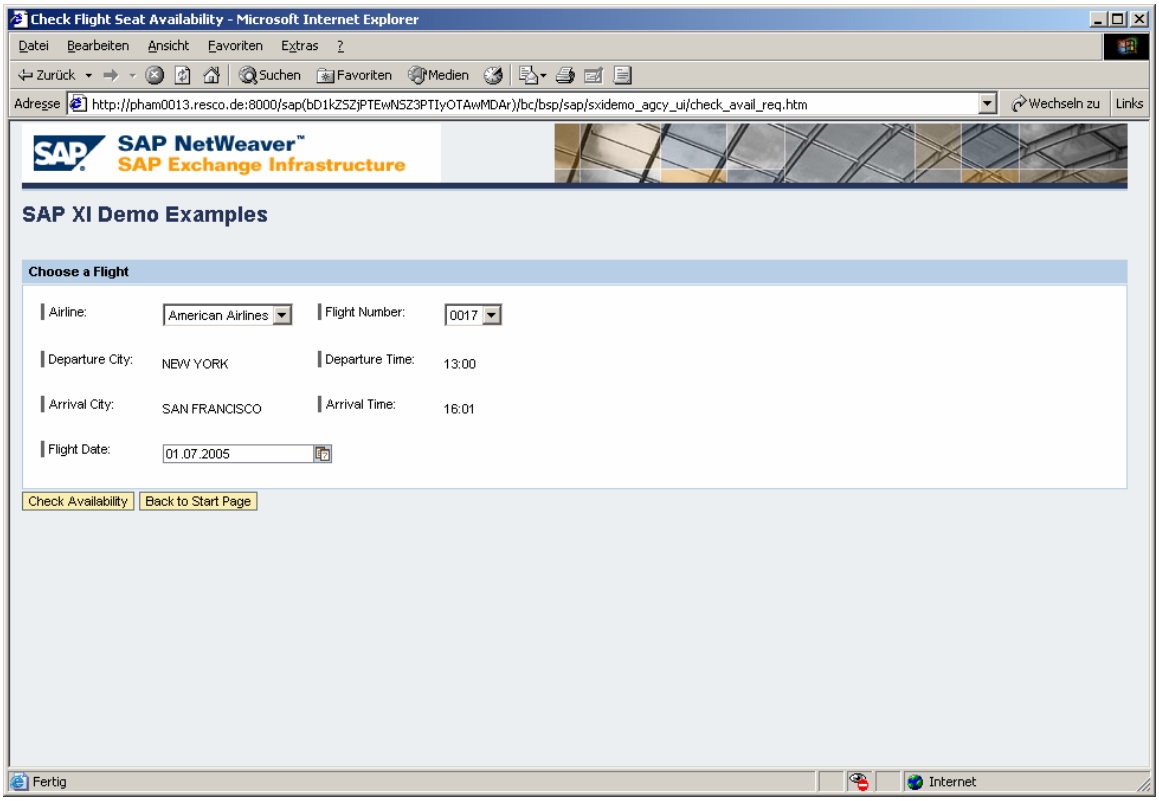

Figure 42 Web GUI SAP XI Demo Examples – Check Flight Seat Availability

Choosing “Check Availability” initiates the request and returns a result page. According to the mapping in BizTalk 5 seats are available in all categories (see Figure 43).

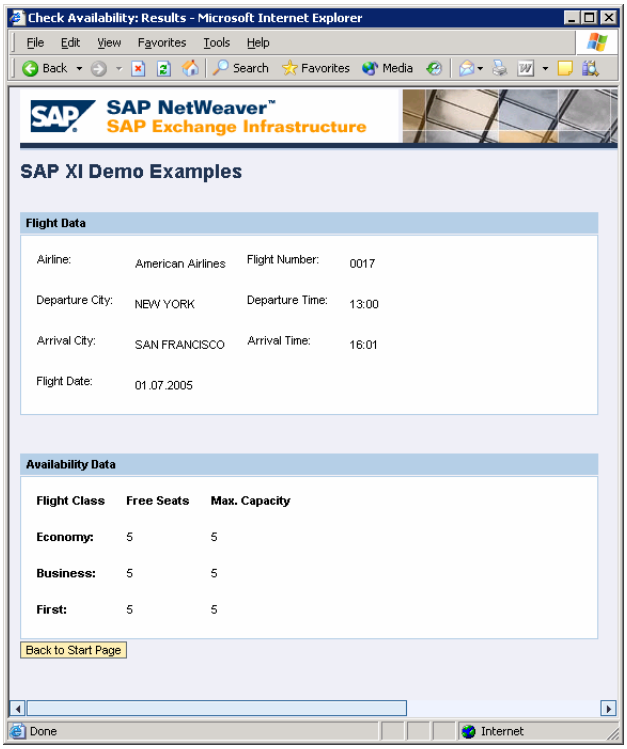

Figure 43 Web GUI SAP XI Demo Examples – Check Flight Seat Availability result

Scenario 3: Asynchronous, transac tional Integration – SAP XI Outbound

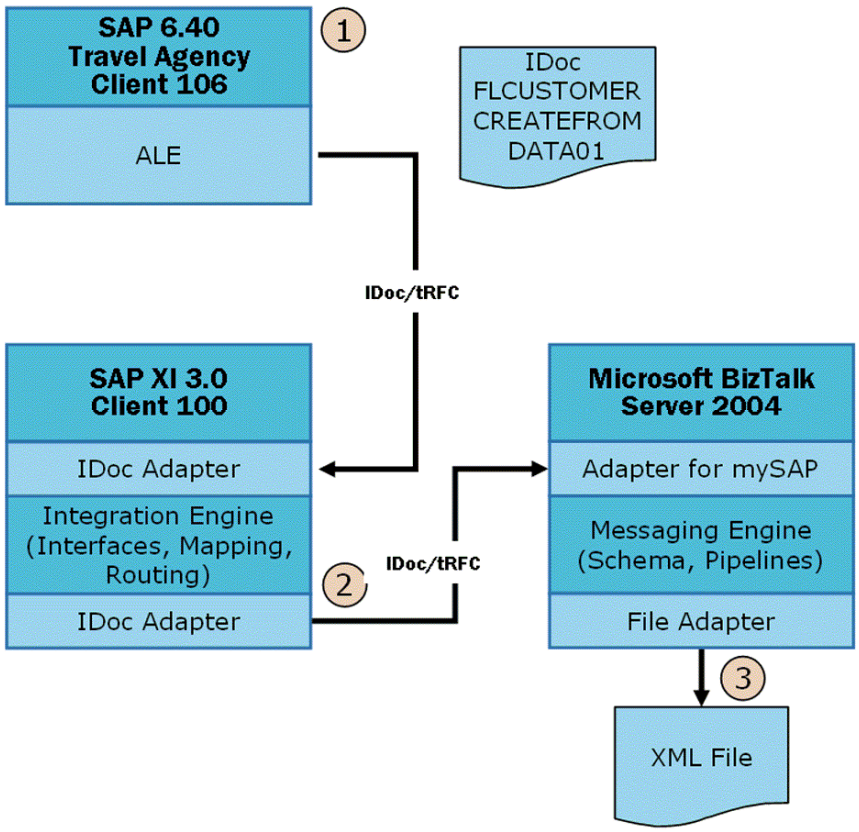

Figure 44 SAP XI IDoc Outbound Scenario

Introduction

In this scenario the agency (XI3_105) sends customer master data to BizTalk via XI (1). BizTalk simply writes the XML representation of the transferred data into a directory on the file system (3). For the integration between XI and BizTalk an IDoc/tRFC is used (2).

SAP XI - Integration Builder (Design)

The IDoc adapter is part of the ABAP stack (in contrast to the SOAP adapter). The configuration is done using SAP transactions within the SAP-GUI.

The IDoc adapter needs corresponding metadata for the conversion of IDoc data streams from or to XML. This metadata is imported into the IDoc adapter configuration at design time. At first the destination of type R/3 connection is defined using the transaction SM59. This connects the XI system with the Airline SAP system (XI3_105) hosting the IDoc definitions.

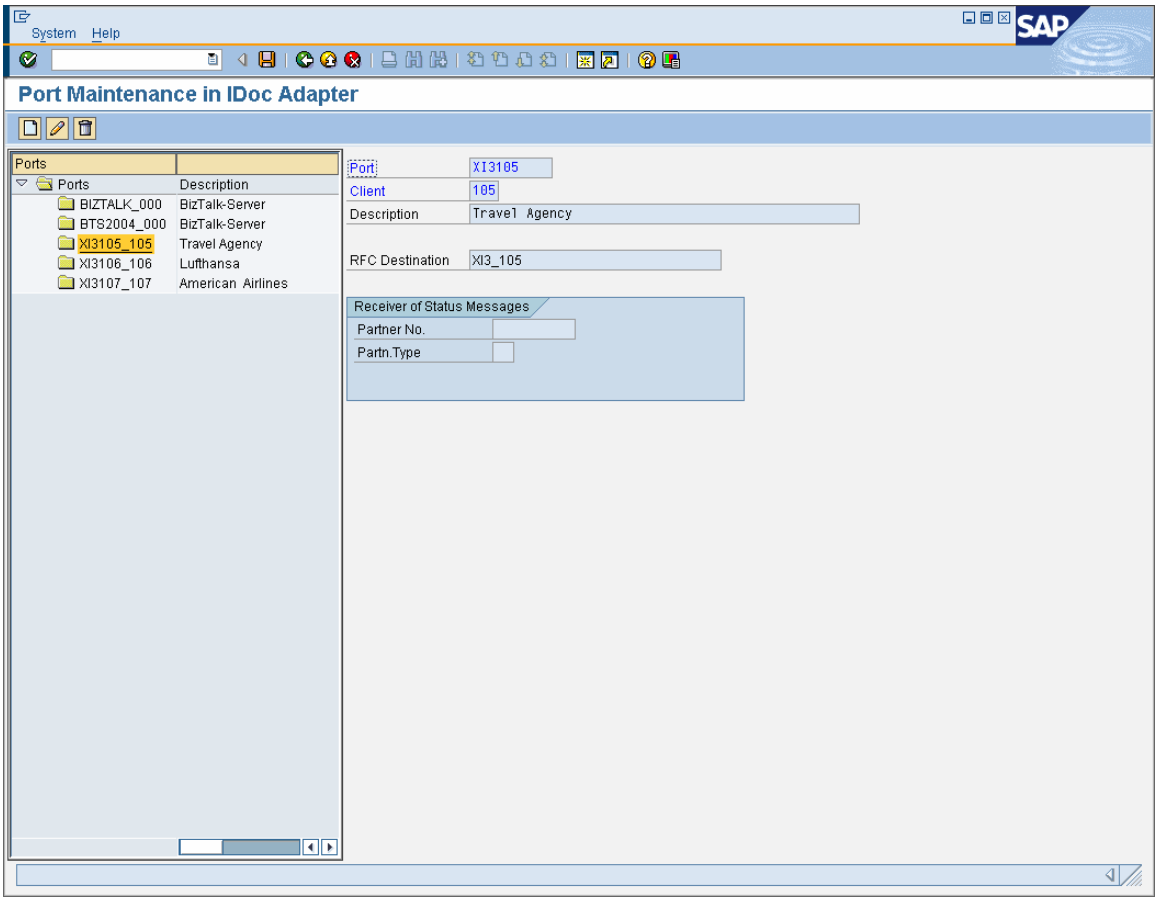

Figure 45 Maintenance of the IDoc adapter port

- Figure 45 shows the registration of the new system in the IDoc adapter (transaction IDX1) as a port.

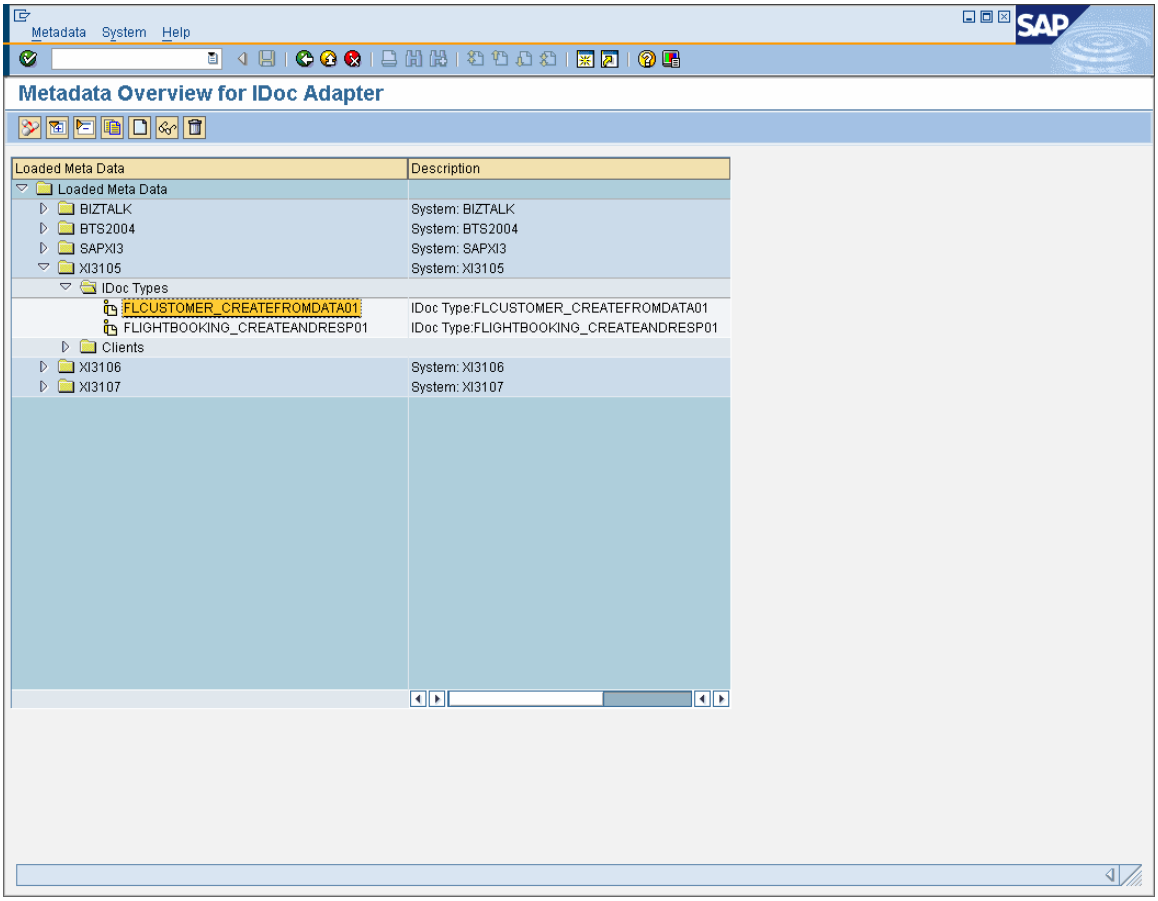

Figure 46 Meta data binding

- Transaction IDX2 allows binding IDoc metadata to a port defined in the port maintenance. In this example the IDocs of type FLCUSTOMER_CREATEFROMDATA01 is submitted to the external system.

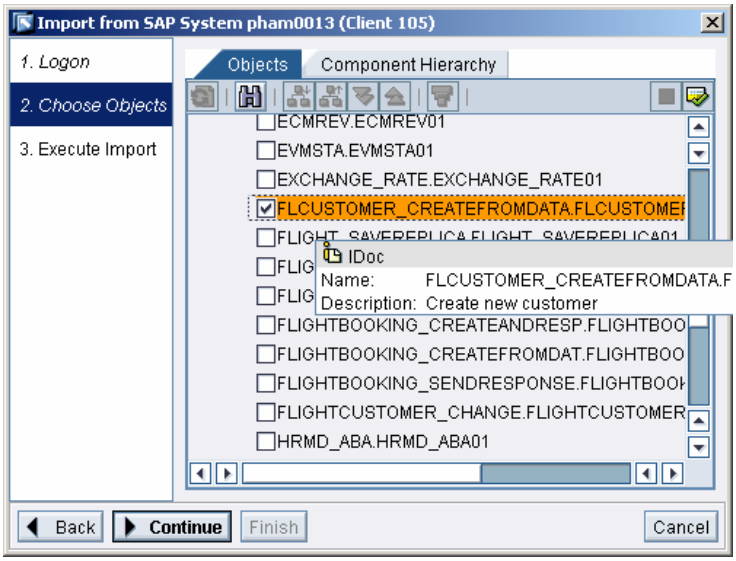

Figure 47 Import of IDoc type

- The IDoc type FLCUSTOMER_CREATEFROMDATA01 is imported into the repository using Integration Builder (Design). Further objects like message mappings are not defined as BizTalk receives the IDocs without any transformation

SAP XI - Integration Builder (Configuration)

The connectivity to the BizTalk IDoc receiver is defined using a communication channel of adapter type IDoc. It is configured with a RFC destination defining a gateway server and gateway host. The BizTalk IDoc receiver and the XI IDoc sender use this destination for interaction. In contrast to the RFC destination mentioned during IDoc adapter port maintenance this destination is outbound (destination type is T).

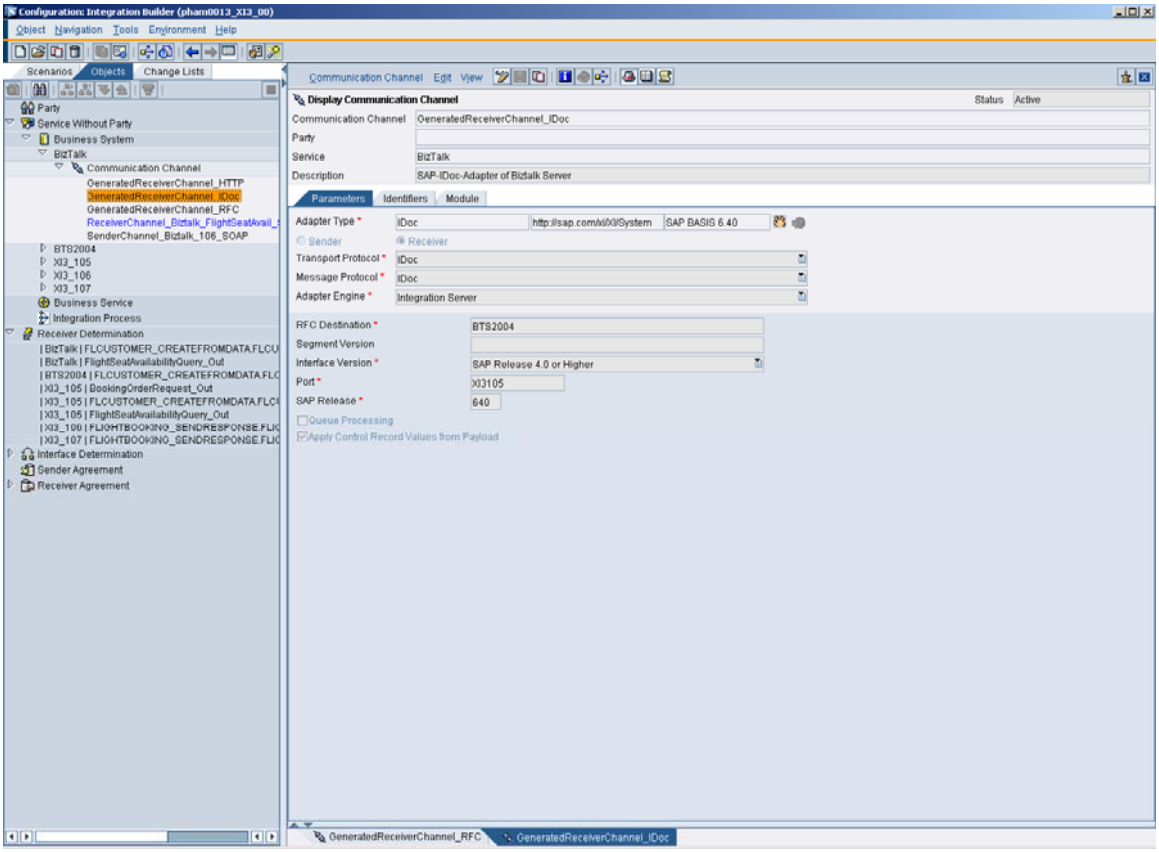

Figure 48 Configuration of the communication channel

The pipeline is configured using the following objects:

- **Receiver determination**
  Sender service: XI3_105
  Interface:
  FLCUSTOMER_CREATEFROMDATA.FLCUSTOMER_CREATEFROMDATA01
  Configured receiver: BizTalk
- **Interface determination**
  Sender service: XI3_105
  Interface:
  FLCUSTOMER_CREATEFROMDATA.FLCUSTOMER_CREATEFROMDATA01
  Inbound interface:
  FLCUSTOMER_CREATEFROMDATA.FLCUSTOMER_CREATEFROMDATA01 (no mapping is performed)
- **Receiver agreement**
  Sender service: XI3_105
  Receiver: BizTalk
  Interface:
  FLCUSTOMER_CREATEFROMDATA.FLCUSTOMER_CREATEFROMDATA01
  Receiver communication channel: GeneratedReceiverChannel_IDoc

Microsoft BizTalk 2004

The BizTalk Server process consists of a receive port for the incoming IDoc and a send port saving the data to the file system as an XML-file.

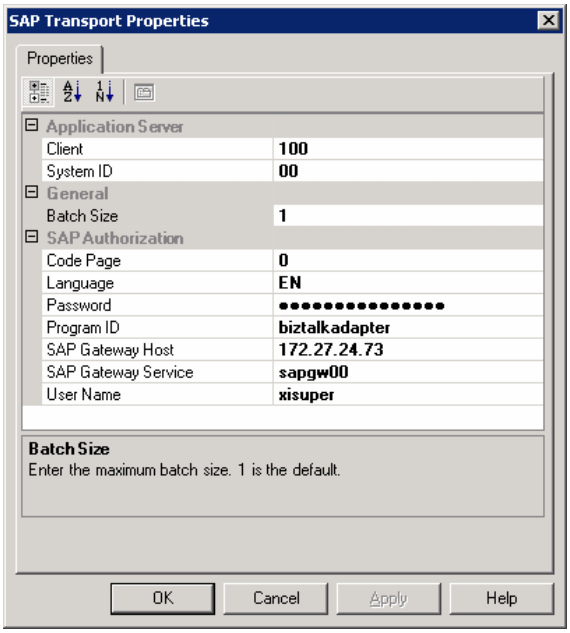

Figure 49 SAP Transport Properties for IDoc Receive Location

- Figure 49 shows the configuration of the receive port location SAP_IDoc_Handler using transport type SAP

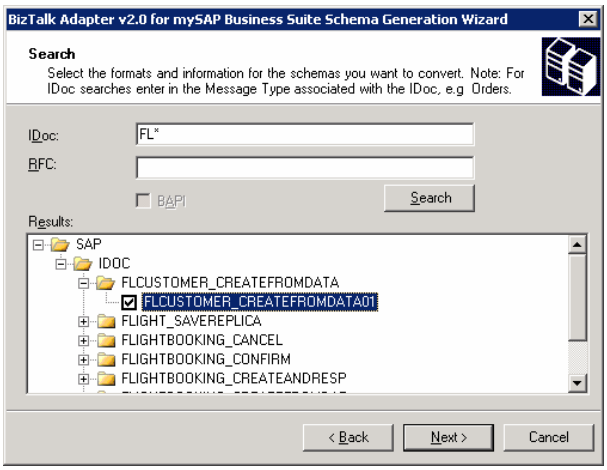

Figure 50 Schema Generation Wizard within Visual Studio .NET

- Figure 50 shows the schema generation wizard within Visual Studio .NET. The schema BizTalk_SAP_XI.FLCUSTOMER_CREATEFROMDATA01 is imported from the underlying SAP BOR.

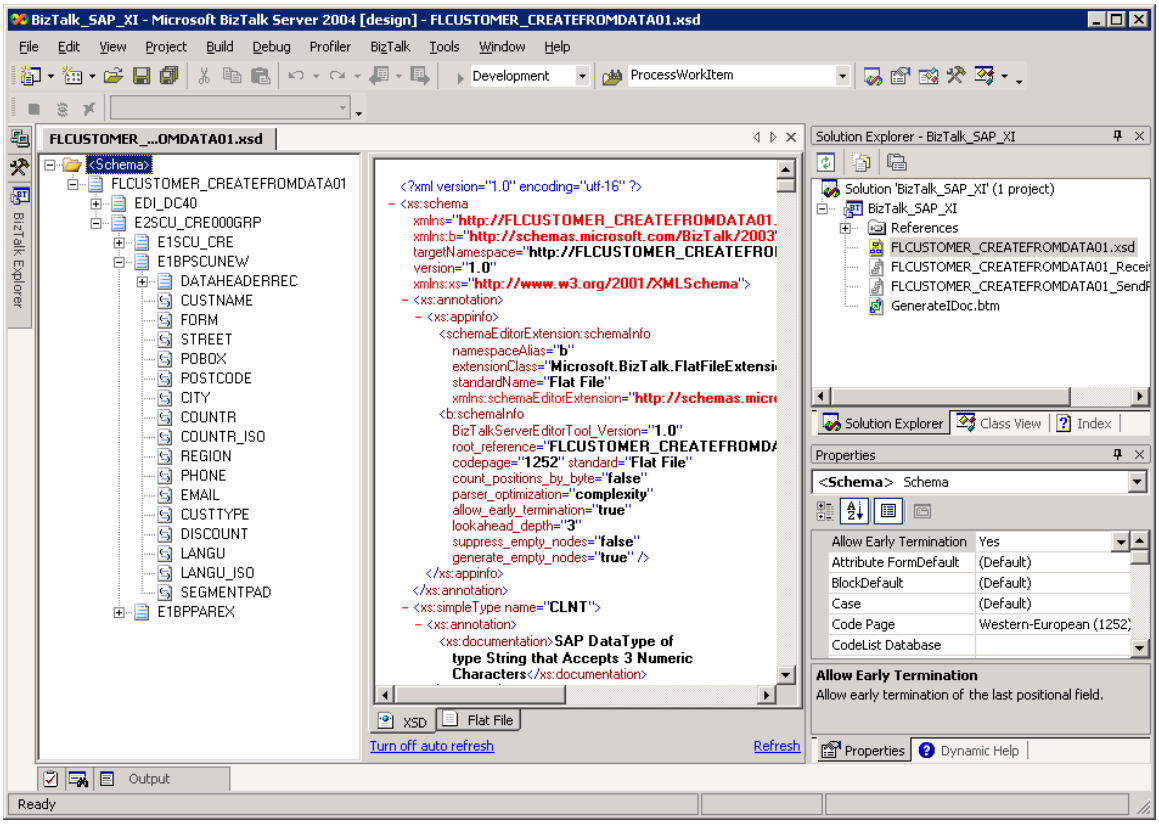

Figure 51 Imported IDoc Schema within Visual Studio .NET BizTalk Project

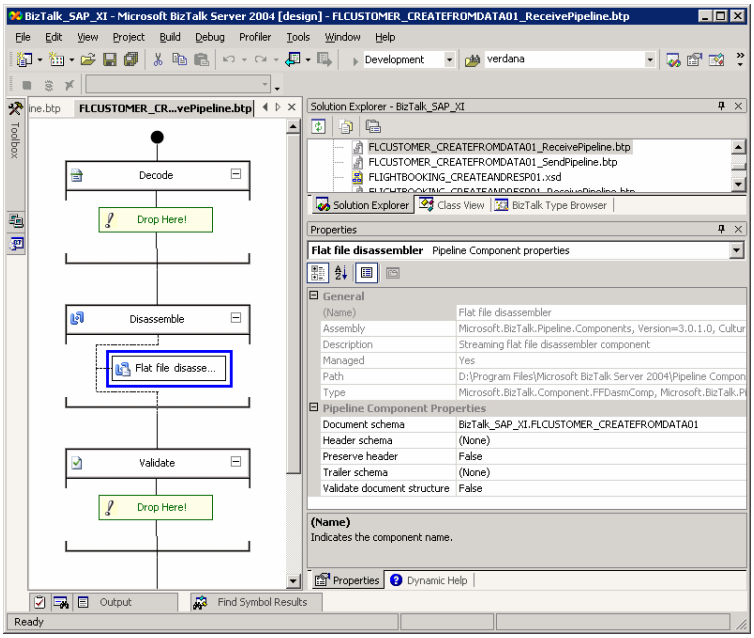

Figure 52 Flat file receive pipeline

- A receive pipeline with a Flatfile-Disassembler using the generated schema is created and applied it to the receive port SAP_IDoc_Handler created above (Figure 49).

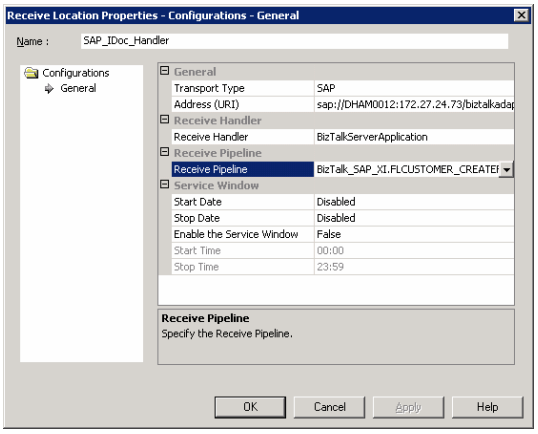

Figure 53 Configuration of the IDoc receive port with custom pipeline

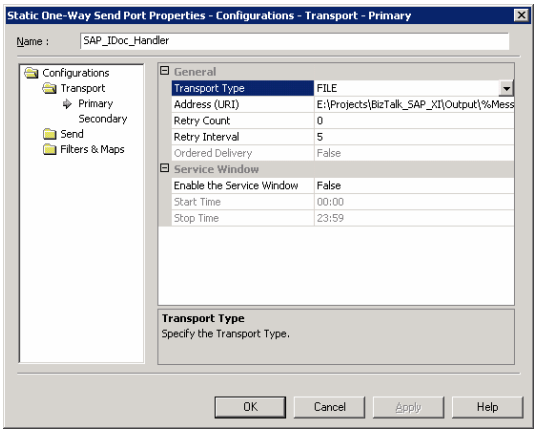

Figure 54 Configuration of the File send port forwarding the IDoc

- A file send port SAP_IDoc_Handler with a message subscription to the incoming message is added. The subscription is defined setting the filter criteria BTS.ReceivePortName to the port name of the receive port (“SAP_IDoc_Handler”).

Initiating and Monitoring

This scenario is initiated within SAP using transaction WE19 that allows the creation of an IDoc FLCUSTOMER_CREATEFROMDATA01 from the scratch.

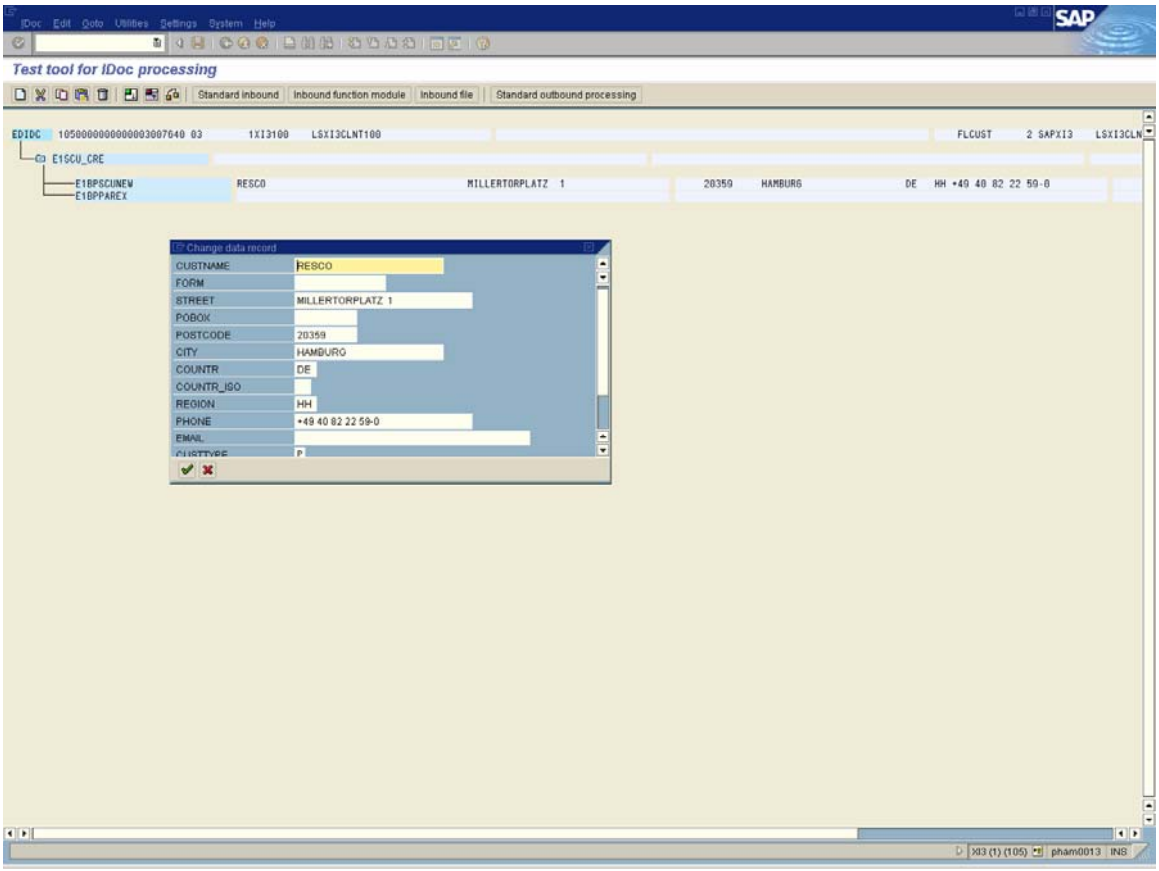

Figure 55 Manual IDoc creation of type FLCUSTOMER_CREATEFROMDATA01

Choosing ‘Standard outbound processing’ passes the customer data IDoc via the XI IDoc Adapter and the XI integration engine to BizTalk. This requires the maintenance of the partner profiles of the agency system(XI3_105) allowing outbound IDoc transfer to XI.

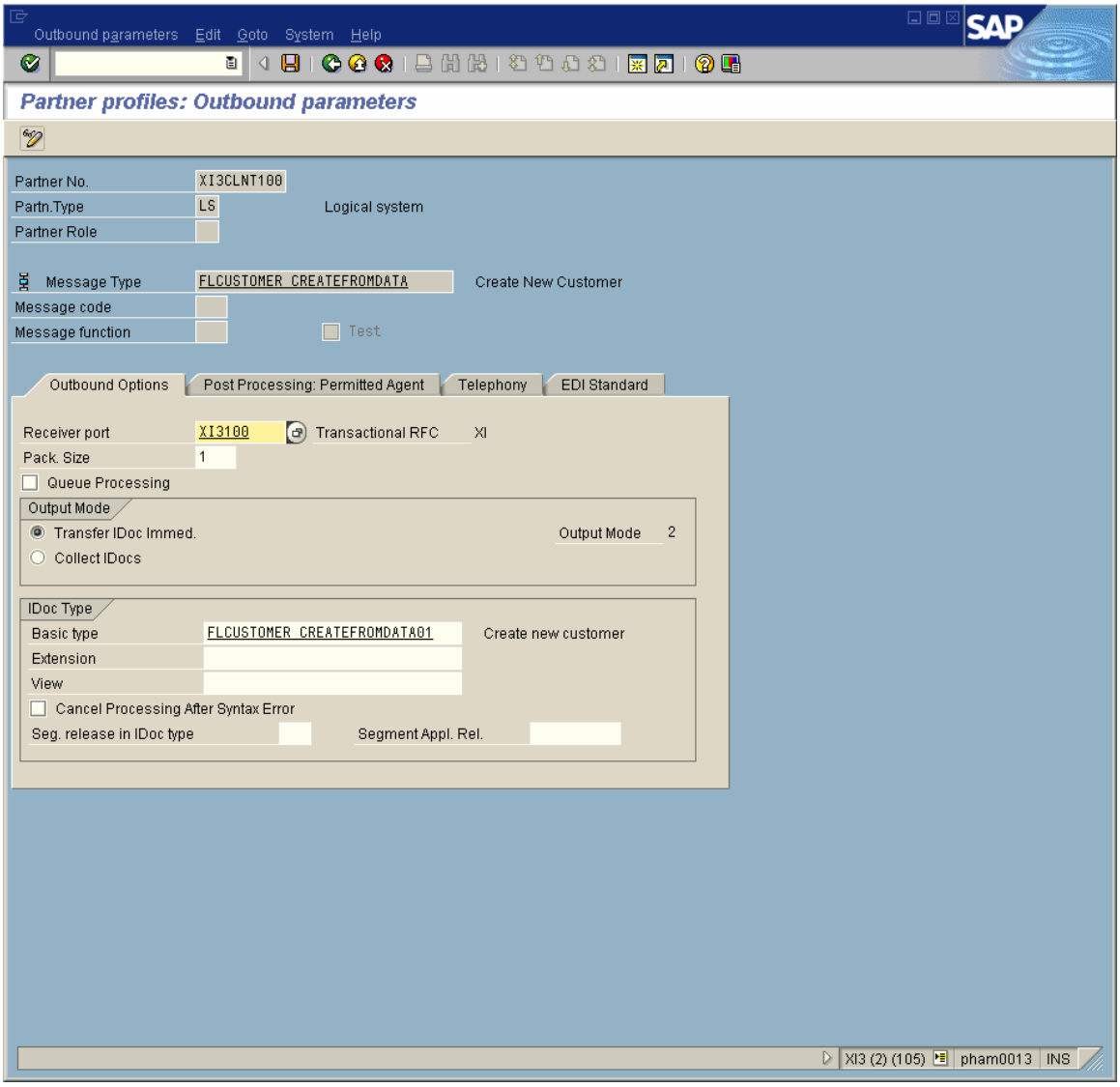

Figure 56 Partner profile for IDoc outbound processing

Because IDocs are processed asynchronously further monitoring has to be done using the XI monitoring (transaction SXMB_MONI).

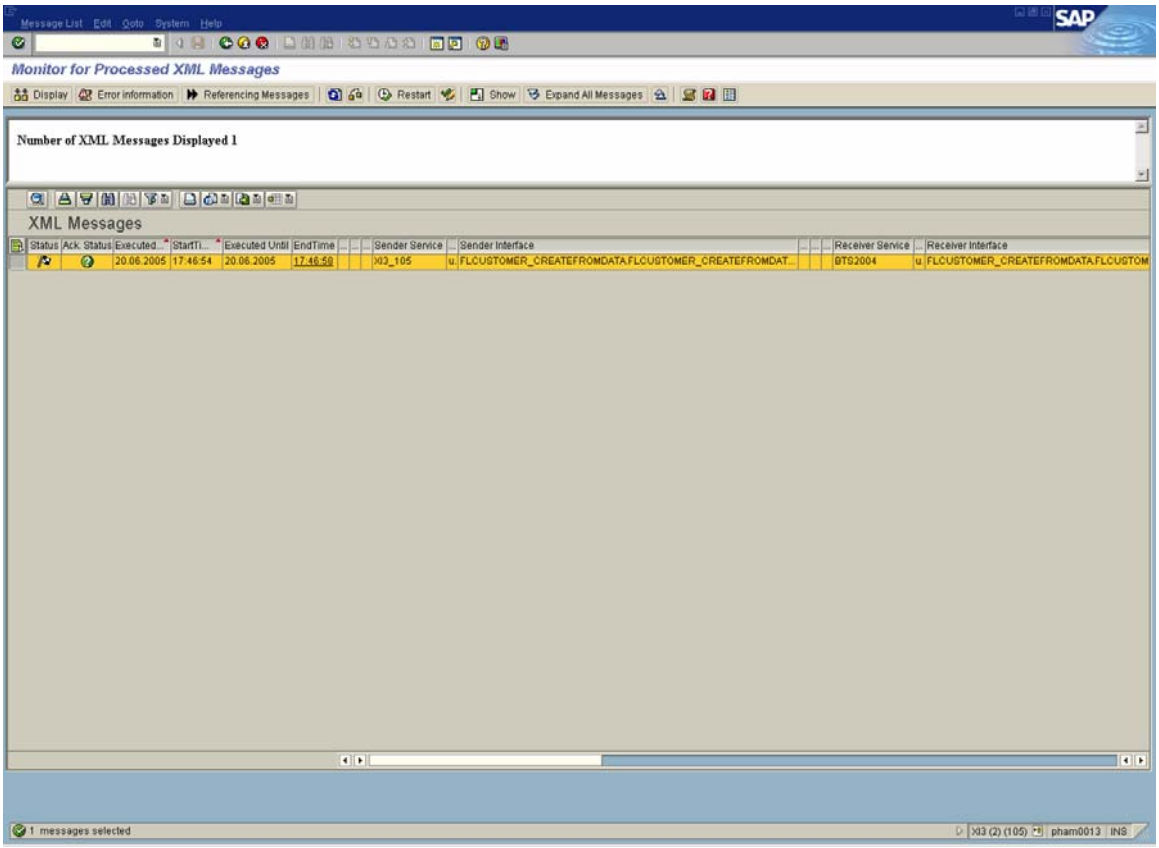

Figure 57 IDoc FlightCustomer passed to Biztalk

Figure 58 shows a file processed by BizTalk containing the IDoc content as XML.

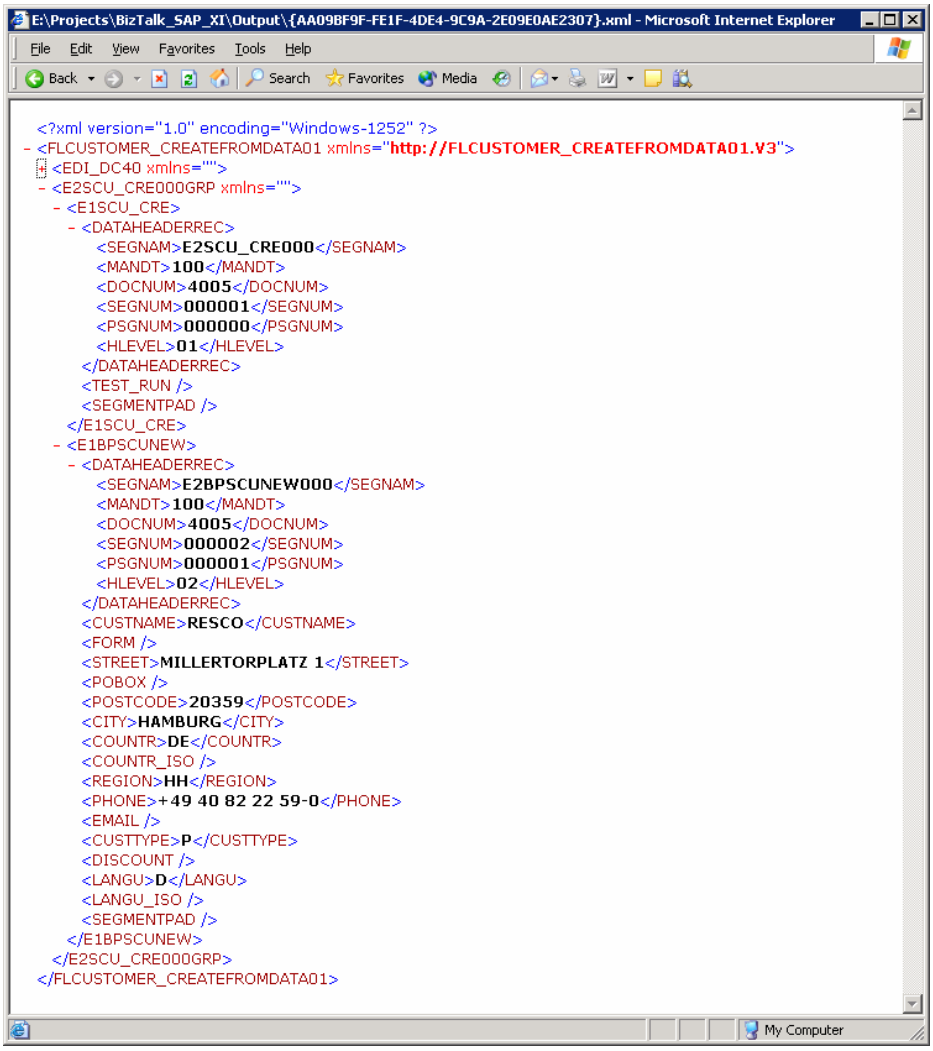

Figure 58 IDoc output as XML

Scenario 4: Asynchronous, transactional Integration – SAP XI Inbound

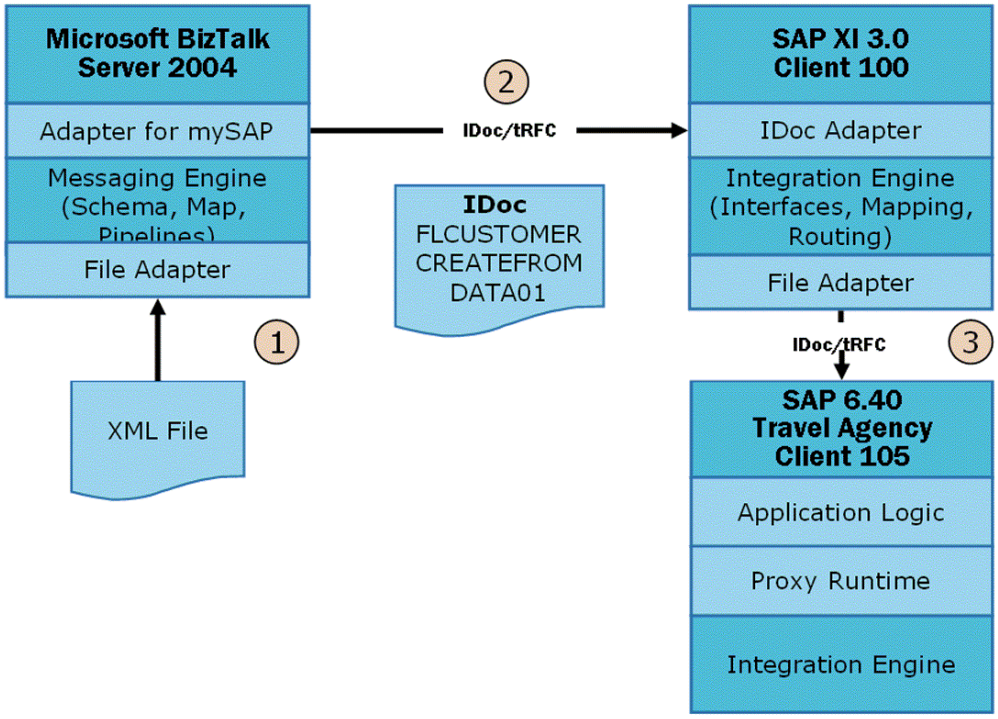

Figure 59 SAP XI IDoc Inbound Scenario

Introduction

In this scenario an external system delivers customer master data via BizTalk to the agency (XI3_105). The external system provides an XML file that is consumed (1) and transformed to an IDoc by BizTalk. After the transformation it is sent to XI using tRFC (2). XI routes the IDoc without any transformations to the agency system (3).

SAP XI - Integration Builder (Design)

BizTalk uses the IDoc type FLCUSTOMER_CREATEFROMDATA01. The metadata for this IDoc type was already imported during the last scenario. A new scenario FlightCustomerCreate is created defining the outbound and inbound interface of this integration process. Both interfaces are based on FLCUSTOMER_CREATEFROMDATA01 as BizTalk already delivers the right format for the agency system using the IDoc Adapter.

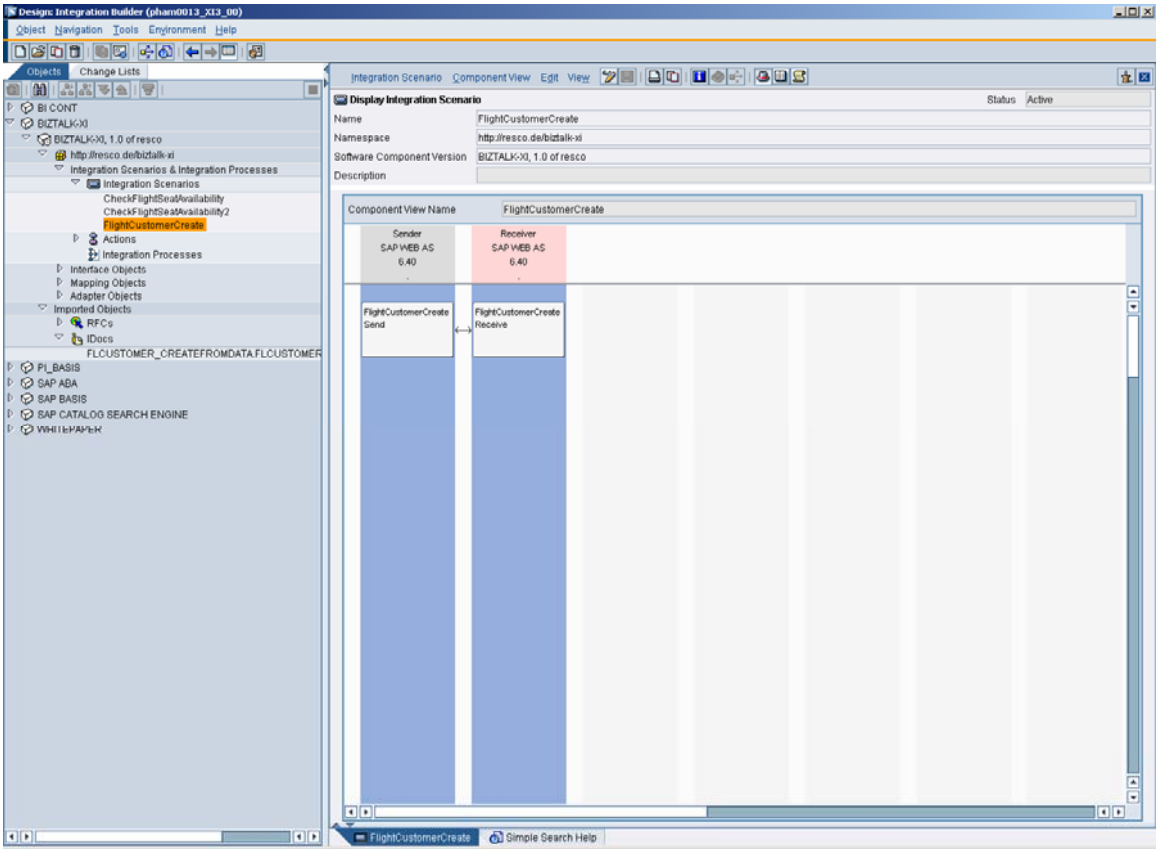

Figure 60 Integration scenario FlightCustomerCreate

SAP XI - Integration Builder (Configuration)

The integration scenario FlightCustomerCreate is imported and configured using the integration scenario editor. The following objects are generated.

- **Receiver determination**
  Sender service: BTS2004
  Interface:
  FLCUSTOMER_CREATEFROMDATA.FLCUSTOMER_CREATEFROMDATA01
  Configured receiver: XI3_105
- **Interface determination**
  Sender service: BTS2004
  Interface:
  FLCUSTOMER_CREATEFROMDATA.FLCUSTOMER_CREATEFROMDATA01
  Inbound interface:
  FLCUSTOMER_CREATEFROMDATA.FLCUSTOMER_CREATEFROMDATA01 (no mapping is performed)
- **Receiver agreement**
  Sender service: BTS2004
  Receiver: XI3_105
  Interface:
  FLCUSTOMER_CREATEFROMDATA.FLCUSTOMER_CREATEFROMDATA01
  Receiver communication channel: GeneratedReceiverChannel_IDoc

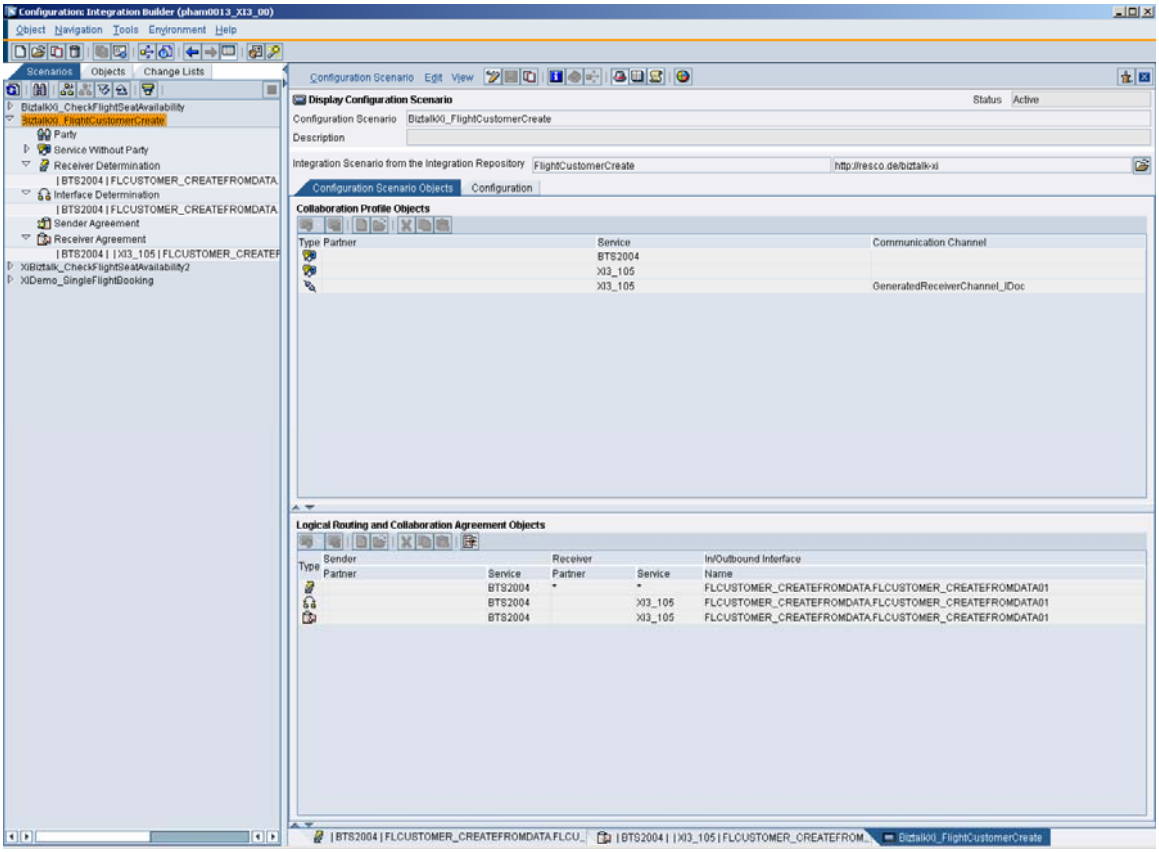

Figure 61 Configuration of the scenario FlightCustomerCreate

- The IDoc adapter is configured using the sender port (in this case BTS2004) to enable metadata lookup. The port BTS2004 is created in transaction IDX1 (port maintenance). The IDoc metadata is bound to this port in transaction IDX2. The metadata is retrieved from the agency system and copied to this port.

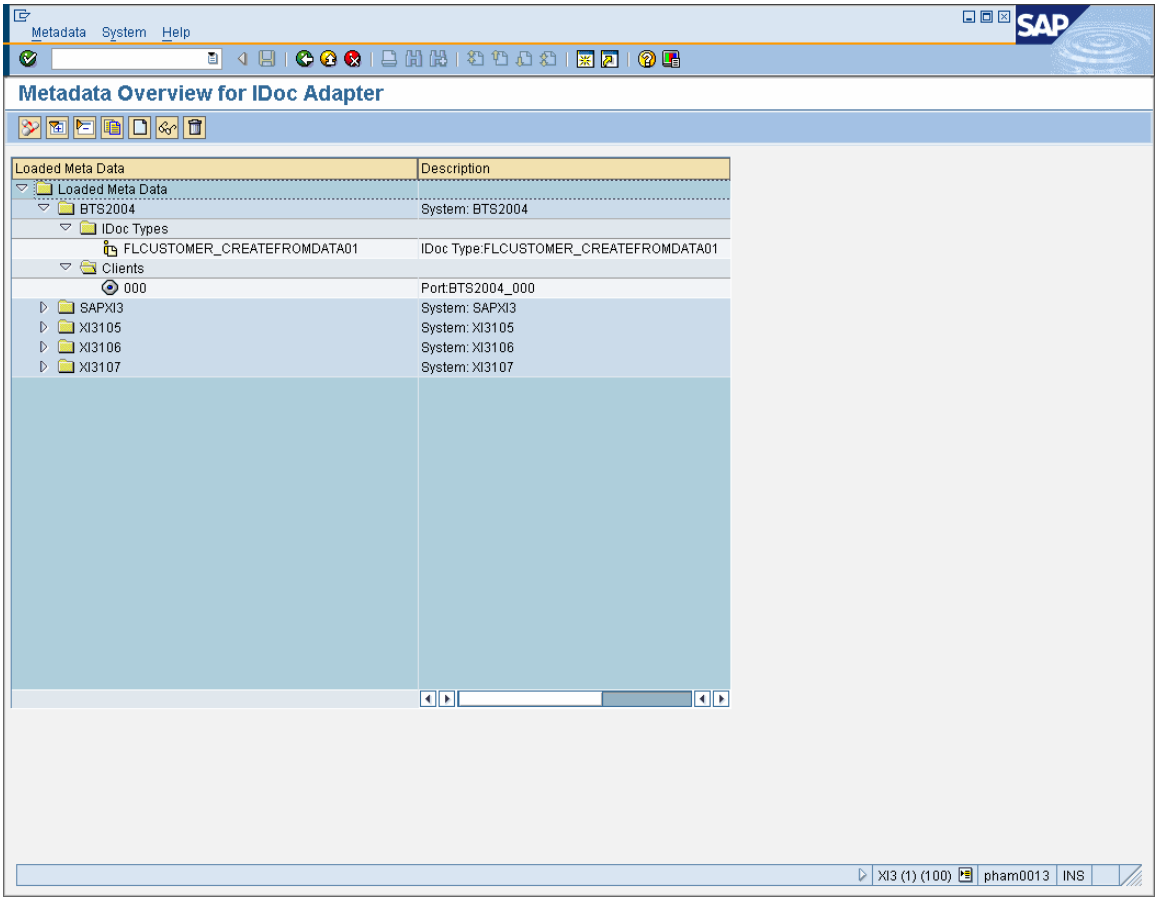

Figure 62 Configuration of IDoc metadata

- The IDoc type FLCUSTOMER_CREATEFROMDATA01 is maintained in the partner profile of the agency system (XI3_105) in order to process the incoming IDoc. The logical system BTS2004 is created. The settings regarding port maintenance and partner profiles can be done with the header mapping functionality in the XI receiver agreement.

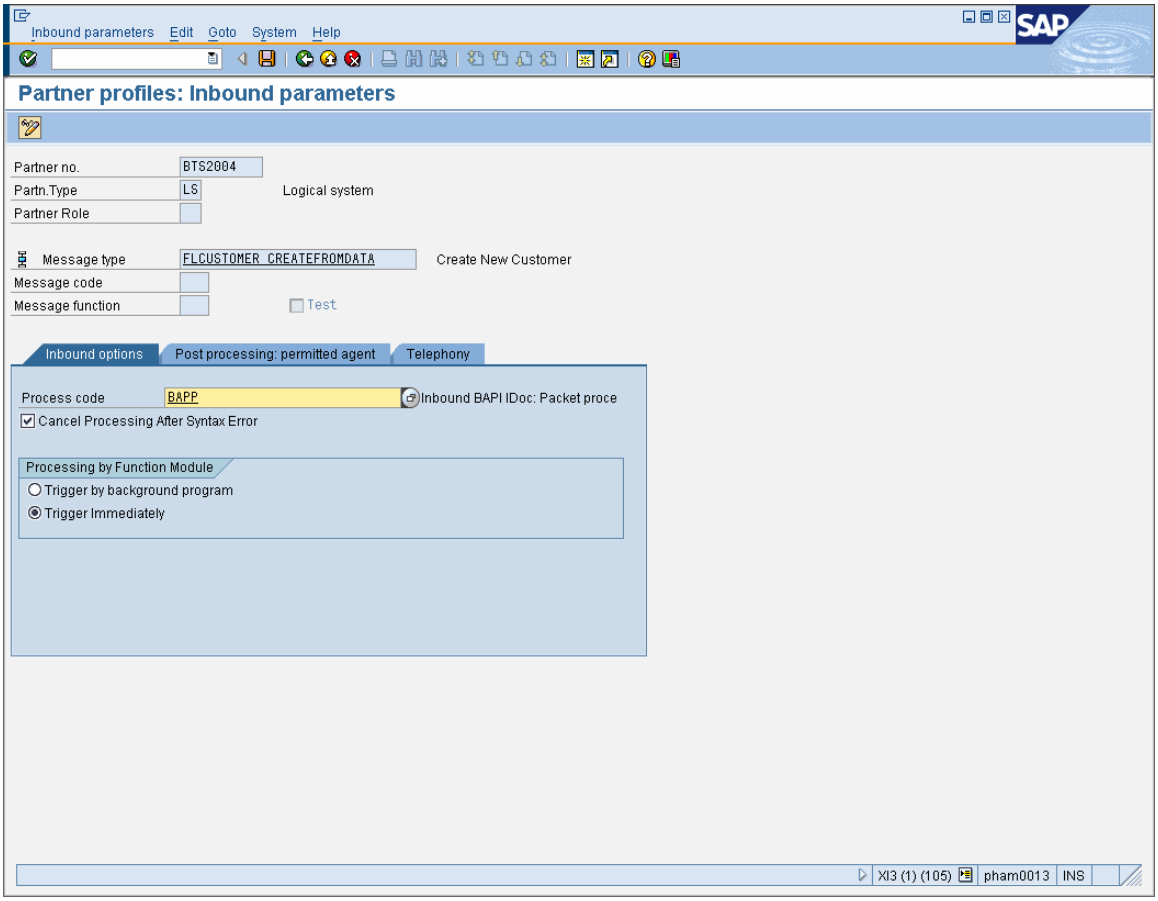

Figure 63 Partner Profile parameters

Microsoft BizTalk 2004

The BizTalk process consists of a receive port for the file to transmit and a send port delivering the IDoc to SAP XI.

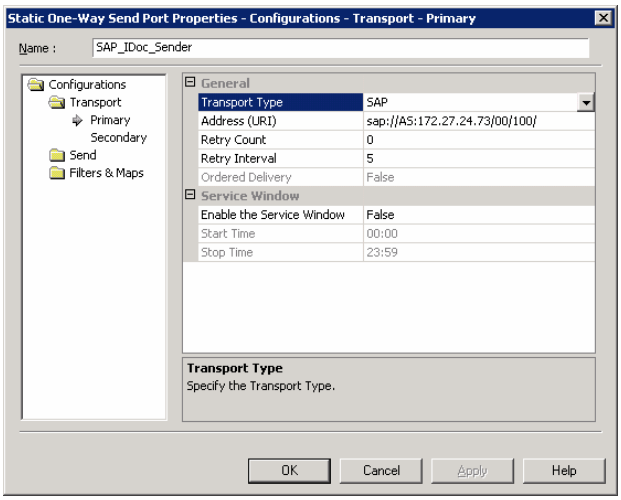

Figure 64 IDoc send port

- Figure 64 shows the configuration of the send port SAP_IDoc_Sender with transport type SAP

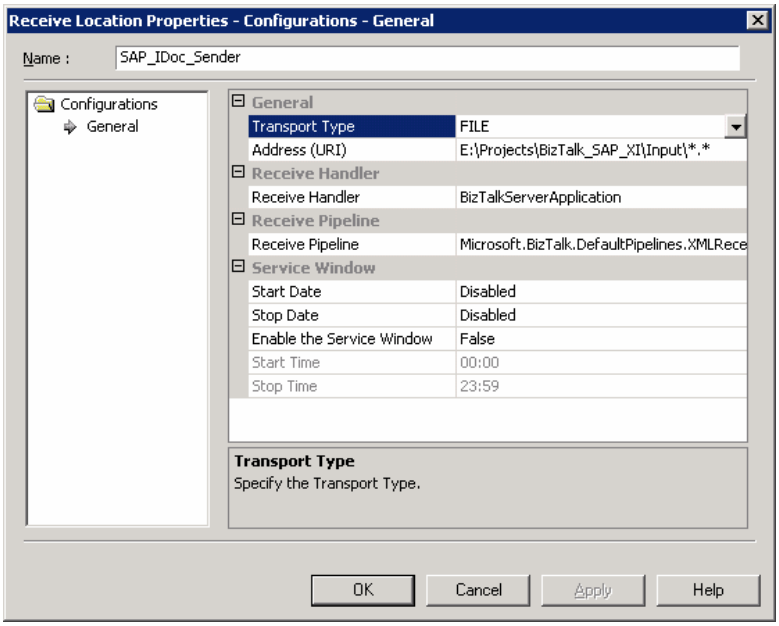

Figure 65 XML file receive location

- Figure 65 shows the configuration of the file receive port and location SAP_Idoc_Sender with the XMLReceivePipeline.

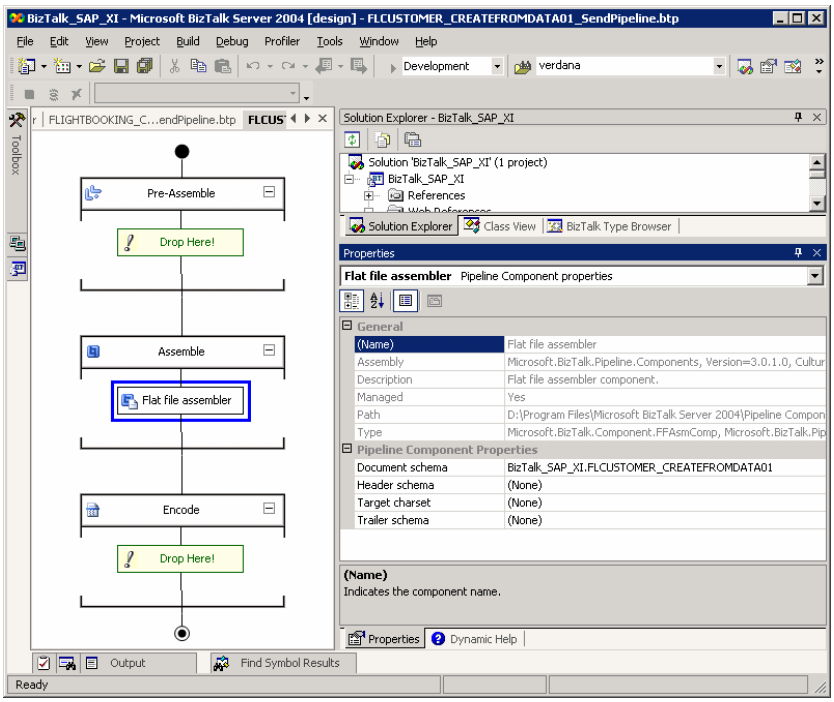

Figure 66 Flat file send pipeline

- A flatfile send pipeline using the schema BizTalk_SAP_XI.FLCUSTOMER_CREATEFROMDATA01 is created and applied to the send port SAP_Idoc_Sender. A subscription of the port is defined to process incoming messages of the receive port by setting the filter property BTS.ReceivePortName to SAP_Idoc_Sender.

Initiating and Monitoring

Similar to scenario 1 the process is started by a file drop to the polled directory of the BizTalk Receive Location SAP_Idoc_Sender. An XML instance taken from scenario 3 can be taken. It has to be customized due to technical and business needs. See Figure 67 for an example.

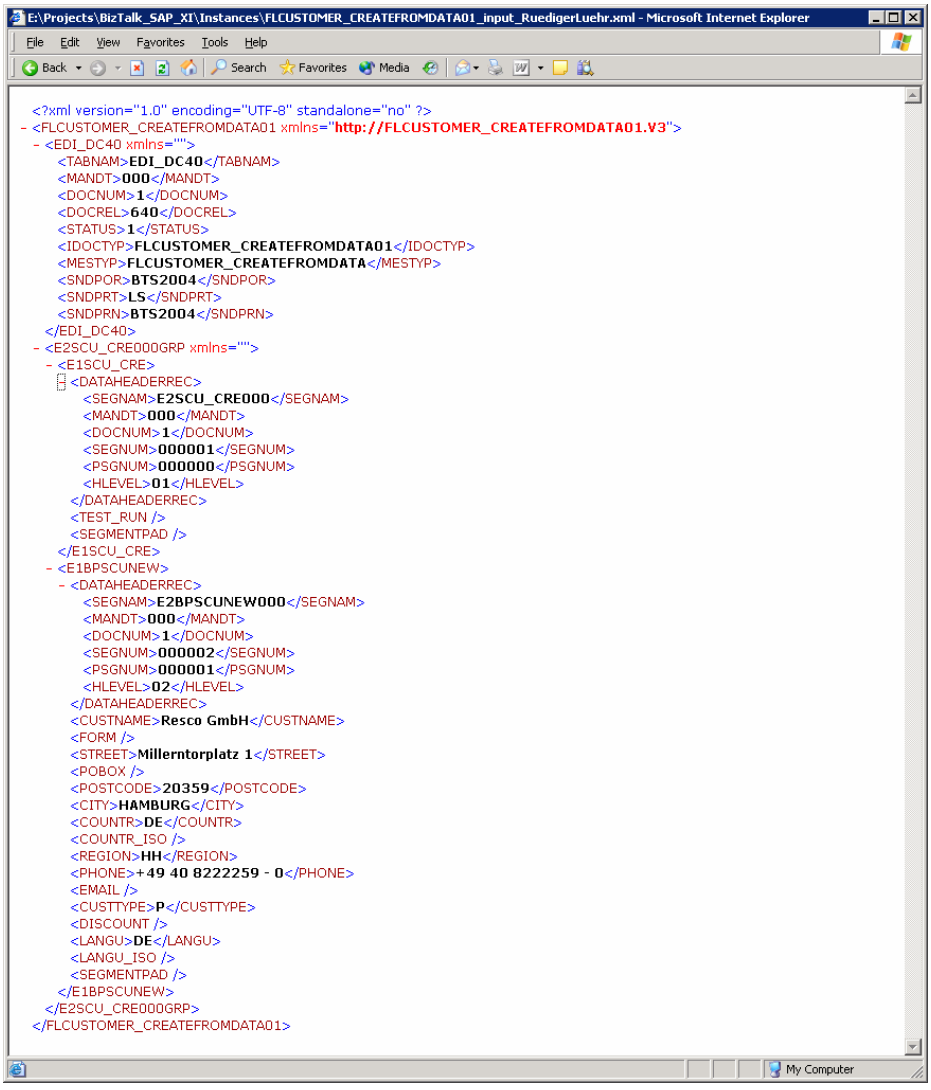

Figure 67 BizTalk IDoc XML message

In the agency system (XI3_105) the incoming IDoc can be displayed using transaction WE02.

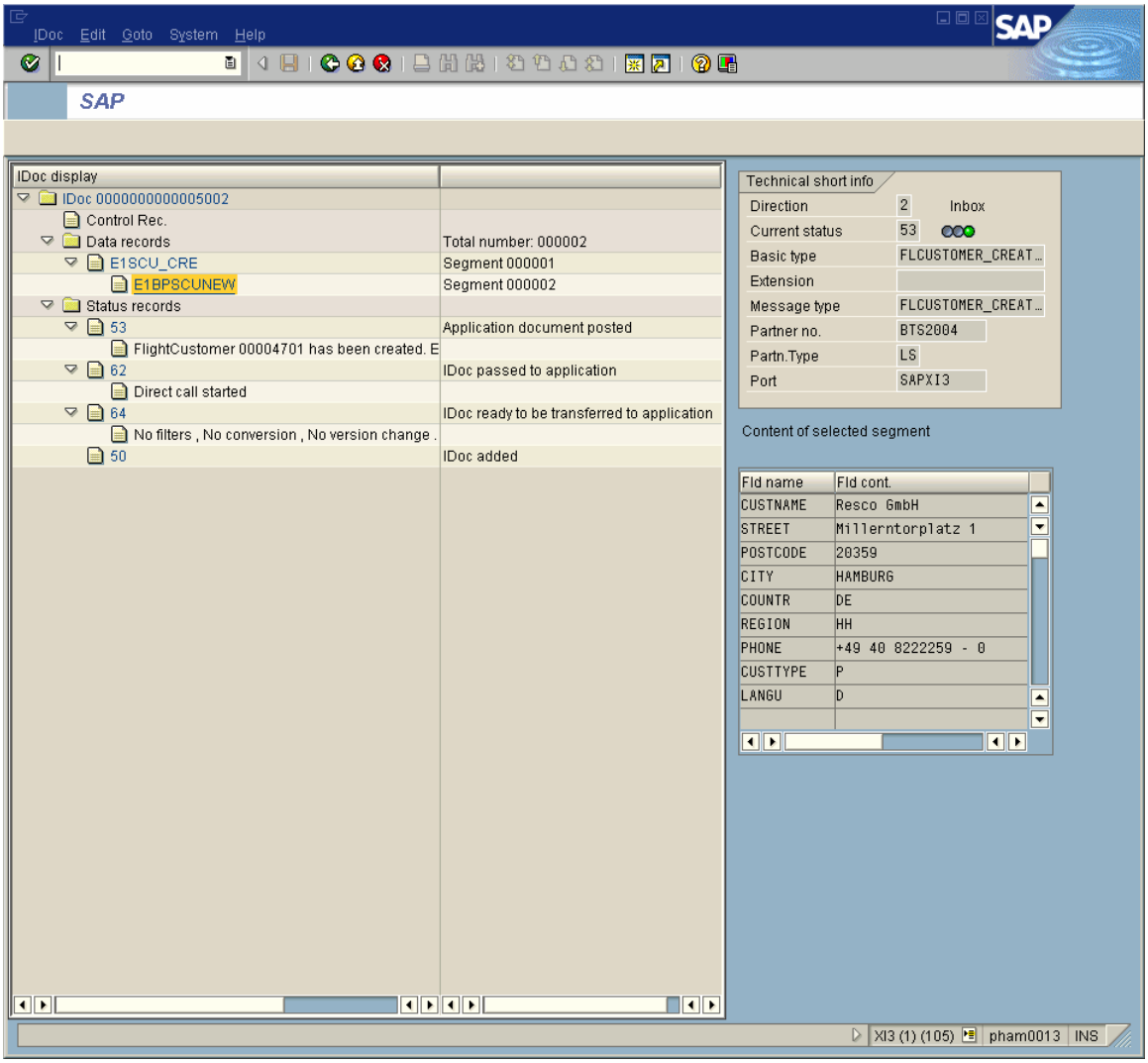

Figure 68 IDoc monitoring using transaction WE02

Process Management

Process management includes the design, modeling and orchestration of processes. It can be approached from two perspectives: The business perspective on the one hand and the technical perspective on the other.

Both integration systems offer tool support for both perspectives.

The SAP XI Workbench containing the process editor is the modeling environment for the SAP Exchange infrastructure. It is used to define scenarios, processes and message structures.

Microsoft BizTalk enables the modeling of business processes from the business perspective using the BizTalk Server 2004 Orchestration Designer for Business Analysts Visio Add-In. The implementation and physical binding of processes is done within Visual Studio .NET.

In a double hub scenario both processing engines get connected whereby the process calling another hub integrates the other's service transparently as a sub-process. This results in a leading process deployed to one hub calling other decoupled processes being deployed to the other hub.

An instance spanning business process design and management is the ultimative goal to establish an integrated process management. In this case an external modeling tool that supports the design the whole process as well as the deployment of the needed parts to both participants is required.

Microsoft BizTalk and SAP XI both offer a standardized interface for the import of business process definitions based on BPEL (Business Process Executing Language).

BPEL or BPEL4WS is a descriptive language standard based on XML describing business processes in a technology independent way. The standardization accepted by OASIS is based on the work and agreement of a number of major Web Services key players including Microsoft and SAP.

The common support of BPEL4WS 1.1 of both integration platforms enables any modeling tool with BPEL support (e.g. ARIS) to define and deploy spanning processes. A process is designed with a central tool and its relevant parts are deployed to the target systems using the BPEL interfaces.

Monitoring

Monitoring includes different aspects of tracking and monitoring integration environments: a technical view on the infrastructure and the tracking of business centric information about processes and messages.

Technical monitoring addresses the observation of the physical machines running the applications, the software infrastructure (operation system, executing runtime, data bases) and the message processing.

Infrastructure monitoring is covered by operation management software like MOM 2005 (Microsoft Operations Manager), OpenView and SAP CCMS. MOM 2005 offers capabilities to monitor Windows, SQL Server, BizTalk and also SAP environments. The SAP support is available through a third party component, the Horizon for SAP MOM Edition from Tidal software.

BizTalk's Health and Activity Tracking and SAP XI's Runtime Workbench offer the tracking of message flow and process activities of the integration system.

Business monitoring uses a content orientated view. It tracks which message is processed in which way at which time and the status of processes. Although the process may be performed beyond application borders its status can be monitored at one point. The tracking system is able to react to events based on given rules (e.g. write an email to an information worker if an incoming order has not been processed within a given time).

Business Activity Monitoring (BAM) of BizTalk Server provides a framework to track and monitor process key performance indicators using SQL Server 2000 database tables and Analysis Services. SAP XI does not offer process monitoring out-of-the-box. Instead SAP BW (business warehouse) can be used to store tracked process data.

Monitoring Level	**Microsoft BizTalk**	**SAP XI**
Business Activity Tracking	Business Activity Monitoring (BAM), SQL-Server Monitoring tools, MOM	SAP-BW
Integration Engine	Health And Activity Tracking	Runtime Workbench
Machine/Infrastructure	MOM, OpenView, CCMS, Tivoli …	

The current version of MOM 2005 expands its monitoring features to the business level. The MOM 2005 Resource Kit allows the observation of BAM Key Performance Indicators (KPIs). Alerts can be raised when these KPIs deviate from normal operating conditions. Therefore MOM 2005 offers comprehensive monitoring functionalities targeting all relevant levels.

Another way to realize instance spanning business monitoring is the definition of a central data repository (e.g. SAP BW or the BizTalk BAM database) collecting necessary data at one point. The data collection may be done by customized orchestration components ("functoids") providing the necessary information at process stages of interest.

Conclusion

The co-existence of SAP XI and Microsoft BizTalk Server within double-hub integration architecture is possible and practicable. Using common messaging and communication technologies all relevant integration scenarios can be implemented incorporating environmental and performance issues.

Web Services, promoted as the universal integration technology, are an important factor to establish the connection between the double-hub. For synchronous and non-transactional calls Web Services based on the WS-I basic profile are the best choice. Whenever asynchronicity or transactions are required the limits of current Web Service technology are obvious. In high volume and/or transactional scenarios IDoc technology still fulfils the demand for performance and reliability better than Web Services. This is because the implementation of important WS-* specification like atomic transactions and reliable messaging will not be available until future versions of the products.

Limitations

The walkthroughs described apply to a SAP XI 3.0 installation including the demo integration scenario. It demands the three clients 100 (SAP XI), 105 (travel agency) and 106 (airline).

BizTalk scenarios apply to a BizTalk 2004 SP1 installation including BizTalk Adapter v2.0 for mySAP Business Suite and SAP Connector for Microsoft .NET 1.0.3. Web Services scenarios additionally rely on Internet Information Services (IIS) 6.0.

References

- Microsoft BizTalk Online Help http://www.microsoft.com/biztalk/techinfo/productdoc/
- SAP Online Help http://help.sap.com
- Understanding Microsoft's Integration Technologies http://www.msnusers.com/biztalkserverstuff/Documents/Understanding%20MS%20Integration%20Technologies%2C%201.0.doc
- Microsoft Web Services Enhancements (WSE) http://msdn.microsoft.com/webservices/building/wse/
- Web Services Interoperability Organization (WS-I) http://www.ws-i.org/
- Microsoft BizTalk Adapter v2.0 for mySAP Business Suite http://www.microsoft.com/biztalk/evaluation/adapter/adapters/sap/2004/default.asp
- Microsoft Indigo http://msdn.microsoft.com/Longhorn/understanding/pillars/Indigo/default.aspx

Table of Figures