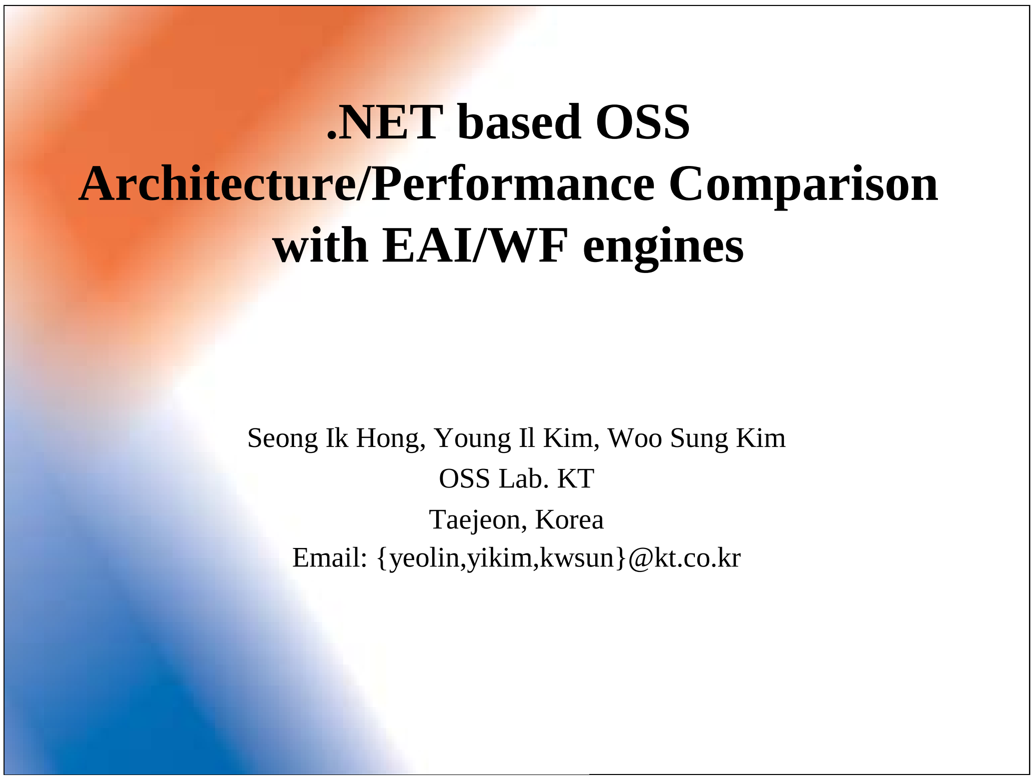


.NET based OSS Architecture/Performance Comparison with EAI/WF engines

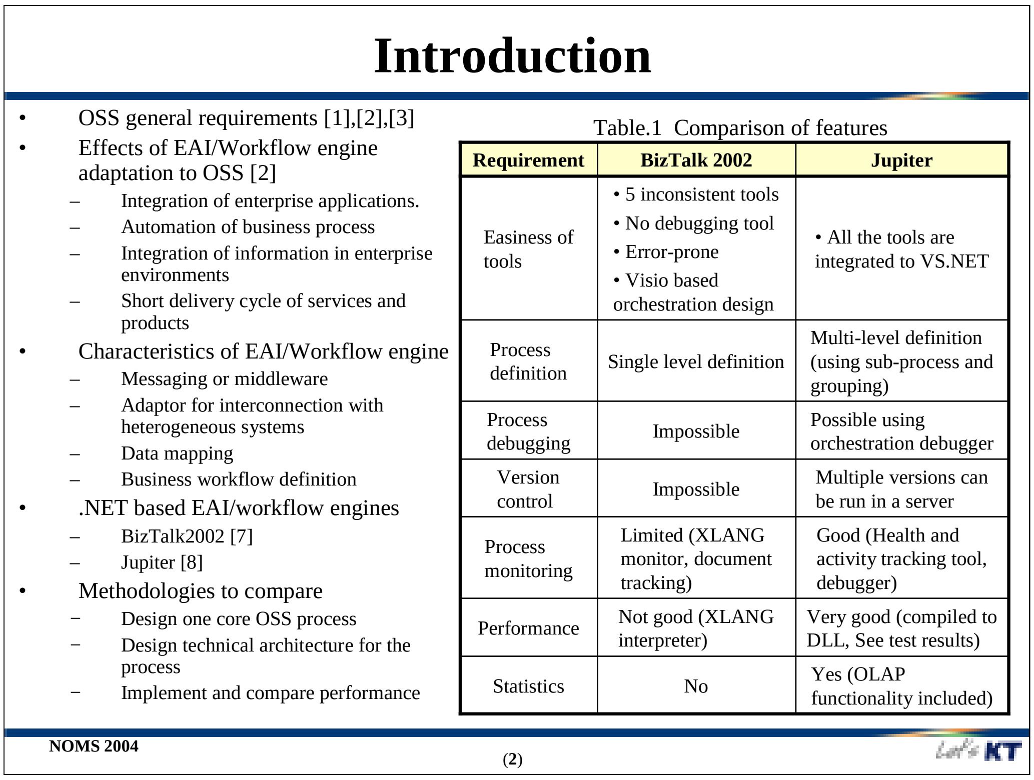
Seong Ik Hong, Young Il Kim, Woo Sung Kim
OSS Lab. KT
Taejeon, Korea
Email: {yeolin,yikim,kwsun}@kt.co.kr

Abstract

As network technologies have tremendously evolved and the customer demand for telecommunications services has increased so fast, the need for OSS to process various service orders and manage a lot of equipments has become bigger and bigger. To meet these requirements, OSS must have some special characteristics such as time-to-market quick deployment, extensible architecture and simple user interface, etc. Most OSS processes are well set-up series of tasks or actions, the order in which they must be performed, so we can use commercial workflow engine to achieve time-to-market OSS deployment.

In this paper, we describe and compare the architecture and framework of two .NET EAI/Workflow solution based OSS and test the performance. The Microsoft BizTalk server is a EAI/Workflow engine that can be used to enforce the OSS workflow definitions. Current version of BizTalk server is 2002, but the totally renew next version, Jupiter will be published in 2004. We will compare the architecture, performance and functionality of OSS using Jupiter and BizTalk 2002.

Key words : OSS implementation, EAI/Workflow engine, .NET platform



Introduction

- OSS general requirements [1],[2],[3]
- Effects of EAI/Workflow engine adaptation to OSS [2]
  - Integration of enterprise applications.
  - Automation of business process
  - Integration of information in enterprise environments
  - Short delivery cycle of services and products
- Characteristics of EAI/Workflow engine
  - Messaging or middleware
  - Adaptor for interconnection with heterogeneous systems
  - Data mapping
  - Business workflow definition
- .NET based EAI/workflow engines
  - BizTalk2002 [7]
  - Jupiter [8]
- Methodologies to compare
  - Design one core OSS process
  - Design technical architecture for the process
  - Implement and compare performance

Table.1 Comparison of features

Requirement	BizTalk 2002	Jupiter
Easiness of tools	• 5 inconsistent tools • No debugging tool • Error-prone • Visio based orchestration design	• All the tools are integrated to VS.NET
Process definition	Single level definition	Multi-level definition (using sub-process and grouping)
Process debugging	Impossible	Possible using orchestration debugger
Version control	Impossible	Multiple versions can be run in a server
Process monitoring	Limited (XLANG monitor, document tracking)	Good (Health and activity tracking tool, debugger)
Performance	Not good (XLANG interpreter)	Very good (compiled to DLL, See test results)
Statistics	No	Yes (OLAP functionality included)

NOMS 2004

(2)

Platforms for OSS have some general requirements as described in [1],[2],[3]. If we use EAI/ Workflow, we can meet previous requirement more easily [2]. EAI/Workflow is a term used to describe the tasks, procedural steps, organizations or people involved, required input and output information, and interconnection with heterogeneous systems. There are some effects of EAI/workflow engine adaptations [2].

- Integration of enterprise applications.
- Automation of business process
- Integration of information in enterprise environments
- Short delivery cycle of services and products

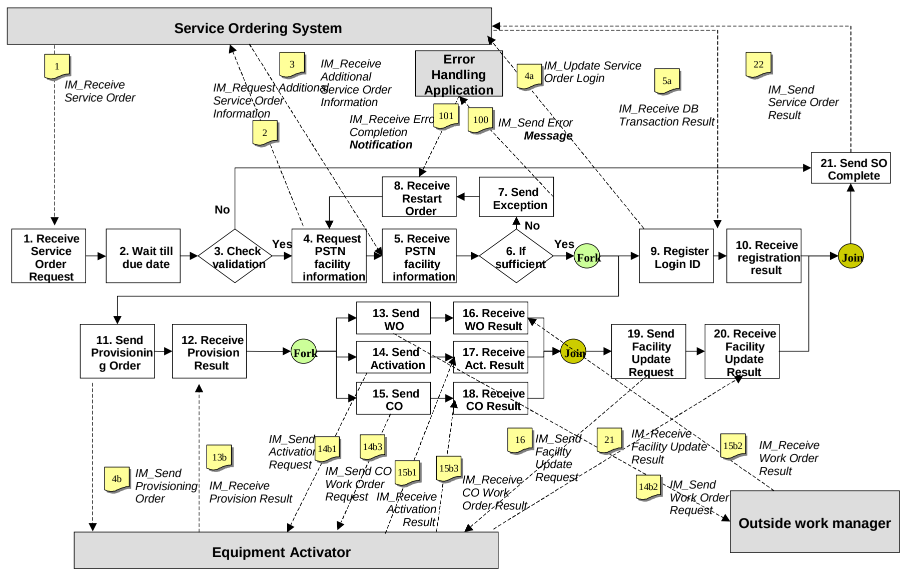
The characteristics of general EAI/workflow engines are as follows.

1. Messaging or middleware
2. Adaptor for interconnection with heterogeneous systems
3. Data mapping
4. Business workflow definition

BizTalk Server enables businesses to achieve application integration and workflow definition through the special functionalities as follows.

1. Messaging through MSMQ, web service and BizTalk messaging
2. Many adaptors such as for MQseries, SAP and web services
3. Mapper functionality for data mapping
4. Orchestration design to define business workflow

ADSL/PSTN Service Order Scenario



Service Ordering System
1
IM_Receive Service Order
IM_Request Additional Service Order Information
2
3
IM_Receive Additional Service Order Information
Error Handling Application
IM_Receive Error Completion Notification
101
100
IM_Send Error Message
4a
IM_Update Service Order Login
5a
IM_Receive DB Transaction Result
22
IM_Send Service Order Result
21. Send SO Complete
No
8. Receive Restart Order
7. Send Exception
No
1. Receive Service Order Request
2. Wait till due date
3. Check validation
Yes
4. Request PSTN facility information
5. Receive PSTN facility information
6. If sufficient
Yes
Fork
9. Register Login ID
10. Receive registration result
Join
11. Send Provisioning Order
12. Receive Provision Result
Fork
13. Send WO
14. Send Activation
15. Send CO
16. Receive WO Result
17. Receive Act. Result
18. Receive CO Result
Join
19. Send Facility Update Request
20. Receive Facility Update Result
4b
IM_Send Provisioning Order
13b
IM_Receive Provision Result
IM_Send Activation Request
14b1
14b3
IM_Send CO Work Order Request
15b1
IM_Receive Activation Result
15b3
IM_Receive CO Work Order Result
16
IM_Send Facility Update Request
21
IM_Receive Facility Update Result
15b2
IM_Receive Work Order Result
14b2
IM_Send Work Order Request
Outside work manager
Equipment Activator

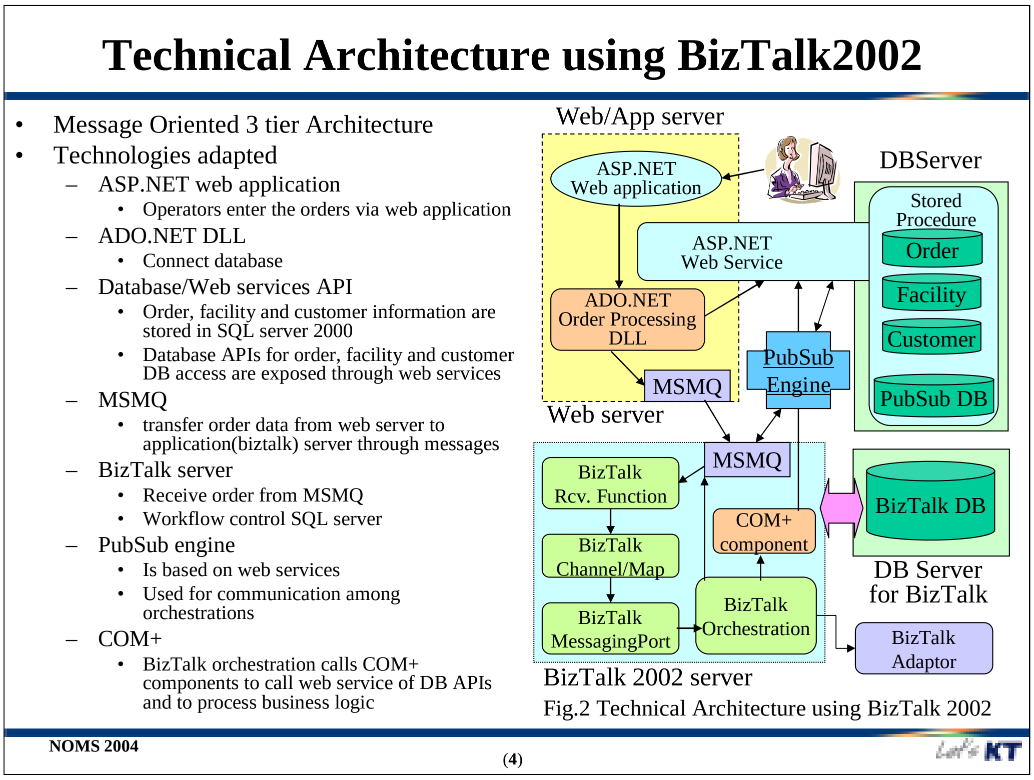
Fig.1 ADSL/PSTN ADD Service Order Scenario

NOMS 2004

(3)

There are a lot of telecommunications services and lots of corresponding ways to manage them. But generally OSS business processes can be categorized as five major processes. Addition, deletion and modification of services, problem handling and monitoring of network devices. Addition is a process of creating a new customer entry in database and setting all the required network equipments to provide the requested service. Deletion is a process of canceling the service, and modification is a service change. Problem handling includes customer data checking, testing the equipment, outside worker dispatch and etc. Monitoring is active surveillance process for network resources. Works in those five categories are not very different especially in technology viewpoint. So, we choose 'ADSL Addition' scenario as a sample scenario because it is the most popular process that occurs. Addition process includes service order validation, due date waiting, registering customer information including login ID, activation of network equipment and order dispatch for office and outside workers. It has many aspects to test EAI/workflow functionalities such as fork and join of process, common parts which can be extracted as a sub-process, conditional branch and interconnections with previously implemented legacy systems such as Equipment Activator.

We have already seen the possibilities using a simplified scenario and BizTalk2002 [2], but in this paper we will implement almost real scenario using BizTalk2002 and the next version, Jupiter. We will compare the two versions in many aspects,too.



Technical Architecture using BizTalk2002

- Message Oriented 3 tier Architecture
- Technologies adapted
  - ASP.NET web application
    - Operators enter the orders via web application
  - ADO.NET DLL
    - Connect database
  - Database/Web services API
    - Order, facility and customer information are stored in SQL server 2000
    - Database APIs for order, facility and customer DB access are exposed through web services
  - MSMQ
    - transfer order data from web server to application(biztalk) server through messages
  - BizTalk server
    - Receive order from MSMQ
    - Workflow control SQL server
  - PubSub engine
    - Is based on web services
    - Used for communication among orchestrations
  - COM+
    - BizTalk orchestration calls COM+ components to call web service of DB APIs and to process business logic

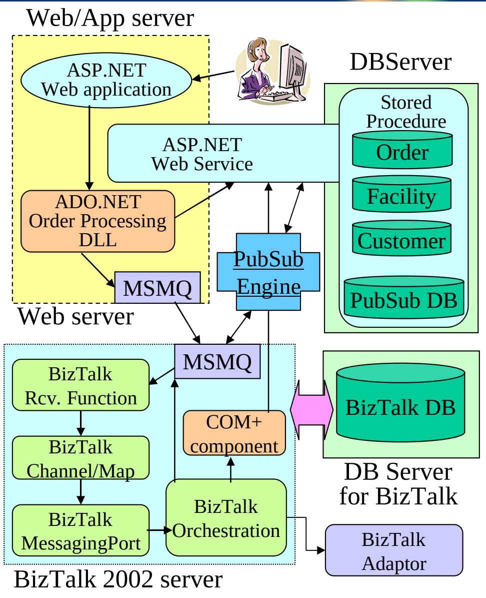


Fig.2 Technical Architecture using BizTalk 2002

Web servers have ASP.NET web applications which enable the operators to enter service orders. ASP.NET server controls enable an HTML-like style of declarative programming that let us build great ordering pages. Displaying customer data, validating operator input, and uploading files are all easy. Best of all, ASP.NET pages work in all browsers including Netscape, Opera, AOL, and Internet Explorer. ADO.NET order processing DLL (Dynamic Link Library) connects the order database and save the service order input data.

We could build robust 3-tier applications using MOM(Message Oriented Middleware), MSMQ(Microsoft Message Queuing) with .NET platform[1],[2]. MSMQ enables web servers to send messages with delivery guarantees that can be applied on a message-by-message basis. MSMQ will ensure that messages get delivered using disk-based storage mechanisms and log-based recovery techniques. MSMQ messages can contain data in any format that is understood by both the sender and the receiver. The web servers send the orders in XML format which becomes a defacto standards for data exchange [4].

Workflows and EAI functions are conducted by BizTalk server. The COM Component shape enables us to use preexisting components to perform database interactions within an XLANG schedule. Using BizTalk server, we cannot implement sub-process so we developed web services based pubsub(publish/subscribe) engine. This is for dynamic calling mechanisms among orchestrations. Details on BizTalk will be explained later.

There are two kinds of databases. One is user data such as order data, facility data and customer data. These are exposed using web services. Data are accessible only through these APIs. The other is for BizTalk server. BizTalk use its own SQL database for management of workflow status.

Design using BizTalk2002 (1/2)

- BizTalk Receive Function
  - monitor documents posted to specified locations (directory, queue or by web site).
- BizTalk Editor
  - enables users to define schema for XML, EDI, and flat files.
- BizTalk Mapper
  - Build document maps that allow applications and business partners who use different document definitions
- BizTalk Messaging Manager
  - Use this wizard-based tool to rapidly define trading partner relationships.
- BizTalk Orchestration Designer
  - Visually define and build robust, distributed business processes.
- BizTalk Server Administration
  - enables administrators to perform common administrative tasks such as adding and removing servers.
- BizTalk Document Tracking
  - enables users to track documents as they move through various stages of the business process.

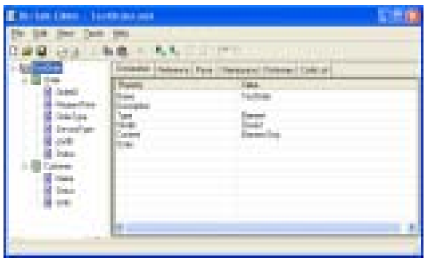


Fig.3 BizTalk Editor

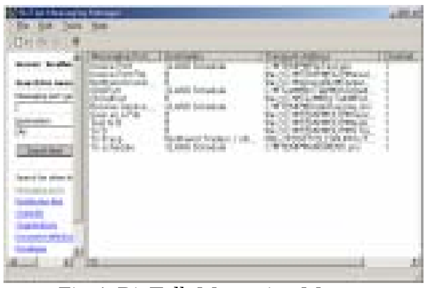


Fig.4 BizTalk Messaging Manager

BizTalk Server provides a set of sophisticated graphical tools that businesses can use to build, transform, manage, track, and analyze business documents. The BizTalk Server tools were designed to simplify the common tasks required to integrate applications and business systems, and ensure the desired functionality is achieved.

BizTalk Server includes the following graphical tools:

•BizTalk Orchestration Designer enables users to visually define and build robust, distributed business processes.

•BizTalk Editor enables users to define schema for XML, EDI, and flat files.

•BizTalk Mapper enables users to link disparate schemas and define steps for successful document transformations.

•BizTalk Messaging Manager is a wizard-based tool that enables users to manage the details of business-process transactions.

•BizTalk Server Administration enables administrators to perform common administrative tasks such as adding and removing servers.

•BizTalk Document Tracking enables users to track documents as they move through various stages of the business process.

Design using BizTalk2002 (2/2)

- Orchestration Designer
  - Visio 2002 based tool
  - No debugger
  - Not designed for complex workflow

- ADSL/PSTN Service order characteristics
  - Action : 28
  - Total Orchestration: 5 (main: 1, subsystem: 4)
    - Sub-orchestration cannot be defined in BizTalk 2002. Subsystem orchestrations are implemented as independent ones and called via web-service based Pub/Sub engine, BizTalk Messaging and MSMQ
  - DB access : 3
  - Decision : 1
  - Fork : 2
  - join : 2
  - Non-realtime work call : 3 (Provisioning, Central office work, Outside work) → orchestration dehydration occurs.
    - If dehydration does not occur, there would be too many running instances. Too many running instance make the system stop.

Fig.5 Main orchestration

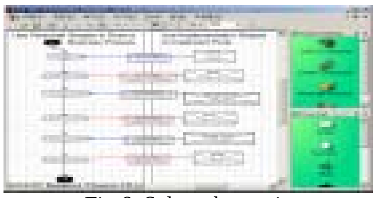


Fig.6 Sub orchestration

NOMS 2004

(6)

Orchestration Designer is a Visio 2002 based tool which includes no debugger. It is not designed for complex workflow design[2], so it has some defects as a workflow engine for OSS implementation.

The characteristics of the orchestration for ADSL/PSTN service orders are as follows:

•Action : 28

•Total Orchestration: 5 (main: 1, subsystem: 4)

•Sub-orchestration cannot be defined in BizTalk 2002. Subsystem orchestrations are implemented as independent ones and called via web-service based Pub/Sub engine, BizTalk Messaging and MSMQ

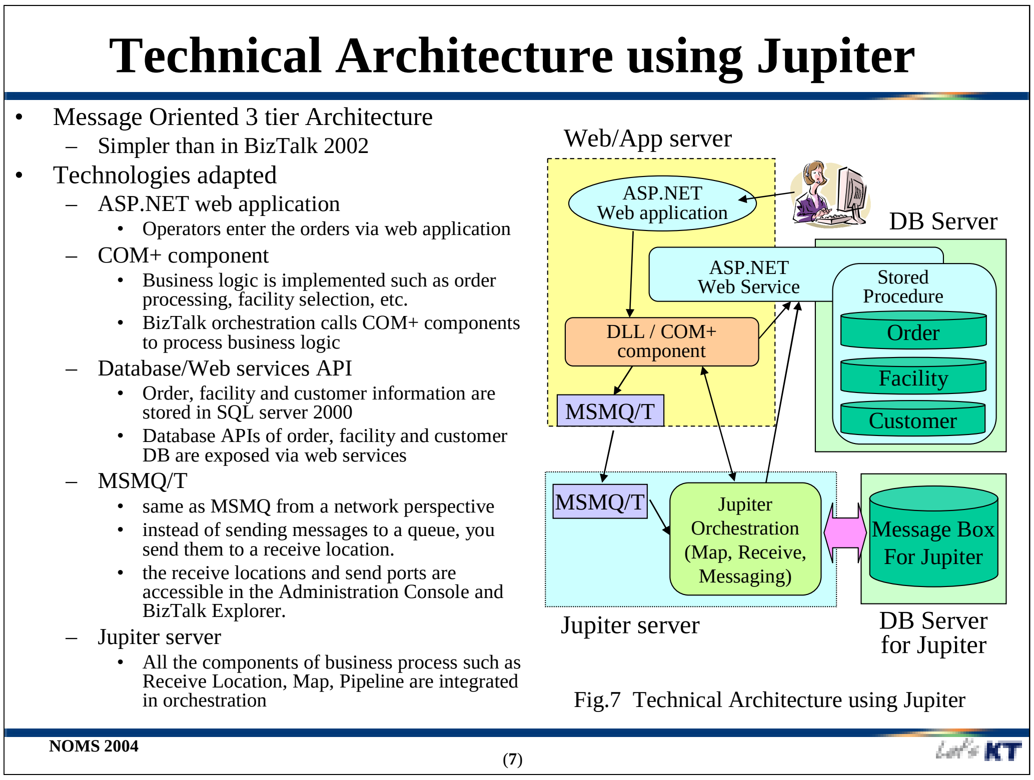
•DB access : 3

•Decision : 1

•Fork : 2

•join : 2

•Non-realtime work call : 3 (Provisioning, Central office work, Outside work) orchestration dehydration occurs. If dehydration does not occur, there would be too many running instances. Too many running instance make the system stop.



Technical Architecture using Jupiter

- Message Oriented 3 tier Architecture
  - Simpler than in BizTalk 2002
- Technologies adapted
  - ASP.NET web application
    - Operators enter the orders via web application
  - COM+ component
    - Business logic is implemented such as order processing, facility selection, etc.
    - BizTalk orchestration calls COM+ components to process business logic
  - Database/Web services API
    - Order, facility and customer information are stored in SQL server 2000
    - Database APIs of order, facility and customer DB are exposed via web services
  - MSMQ/T
    - same as MSMQ from a network perspective
    - instead of sending messages to a queue, you send them to a receive location.
    - the receive locations and send ports are accessible in the Administration Console and BizTalk Explorer.
  - Jupiter server
    - All the components of business process such as Receive Location, Map, Pipeline are integrated in orchestration

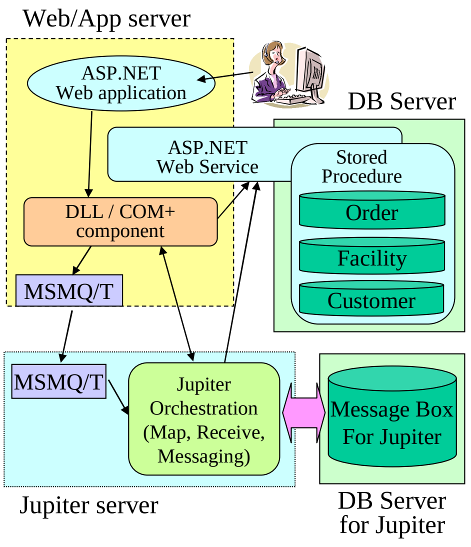


Fig.7 Technical Architecture using Jupiter

NOMS 2004

(7)

We implemented message oriented 3 tier OSS architecture as in with BizTalk 2002. Almost all the portions except that BizTalk server has changed to Jupiter are same, and many separate functions such as map, ports are integrated to orchestrations.

ASP.NET web application are same as in BizTalk 2002. Business logics such as order processing, facility selection, etc are implemented with COM+ components. BizTalk 2002 uses COM+ directly to communicate with business logics outside the BizTalk. But Jupiter can use many ways such as web service to call other business logics, so COM+ components are placed in Web/Application server as Microsoft recommended for their general 3 tier architecture. It is natural because web components are generally tightly coupled with business logic components.

Databases are totally accessible through web service API. Web service is a good technology for database API because it can be easily accessed regardless of the platform of the clients and in many cased transactions with databases are completed in a minute.

MSMQ/T is same as MSMQ from a network perspective but a little different at some points that instead of sending messages to a queue, you send them to a receive location. The receive locations and send ports are accessible in the Administration Console and BizTalk Explorer.

Jupiter server will be explained in detail at the next two pages.

Design using Jupiter (1/2)

- All development tools are integrated in Visual Studio .NET
  - Receive Location (former BizTalk Receive Function)
    - Receive messages
    - Added by BizTalk Explorer
  - Editor (Improved)
    - enables users to define schema for XML, EDI, and flat files.
  - Mapper (Improved)
    - Build document maps that allow applications and business partners who use different document definitions
  - Pipeline Designer (former BizTalk Messaging Manager)
    - is a graphical tool that makes a pipeline which is a series of COM or .NET components that send or receive messages.
  - Orchestration Designer (Improved)
    - Visually define and build robust, distributed business processes.
  - Explorer (former BizTalk Server Administration)
    - enables users to view the different modules in the Configuration database and the associated resources.
  - Orchestration Debugger (new)
    - allows you to see the progress of an Orchestration instance on a shape-by-shape basis
  - Tracking Profile Editor (new)
    - used to define the interesting parts of their business process as well as interesting business payload data.

Fig.8 Orchestration Designer

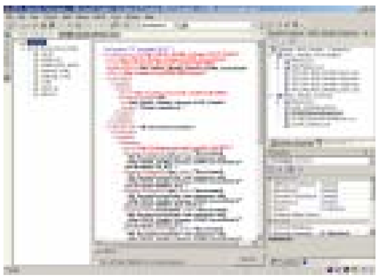


Fig.9 Message Design (Editor)

All development tools are integrated in Visual Studio .NET. Receive location, former BizTalk Receive Function, is the physical, design-time notion of a location (such as a URL) and a protocol type.

•A receive location in the Host node in the BizTalk Server Administration console, defines the receive functionality.

•Editor is a visual tool that simplifies the process of creating structured schemas, specified in XSD, for both XML and non-XML formats.

•Mapper is a graphical user-interface tool that simplifies the process of specifying an XML message transformation, based on two schemas created with the Schema Editor, producing an XSLT stylesheet as compiled output to be used by the server transformation run-time.

•Pipeline Designer, former BizTalk Messaging Manager, is a graphical tool that makes a pipeline which is a series of COM or .NET components that send or receive messages.

•Orchestration Designer visually define and build robust, distributed business processes.

•Explorer, former BizTalk Server Administration, enables users to view the different modules in the Configuration database and the associated resources.

•Orchestration Debugger allows you to see the progress of an Orchestration instance on a shape-by-shape basis

•Tracking Profile Editor used to define the interesting parts of their business process as well as interesting business payload data.

Design using Jupiter (2/2)

- Workflow engine functionality improved
  - Business Activity Monitoring
  - Real-time Tracking
  - Human-based Workflow
  - Business Process Execution Language based
  - Business Rules
  - Sub-orchestration design
  - Expand/Collapse Group assignment
  - Technical part/business part separation
  - Debugging

- ADSL/PSTN Service Order
  - Action : 30
  - Orchestration: 5 (Main: 1, Sub: 3, Common: 1)
  - DB access : 3
  - Decision : 2
  - Fork : 2
  - Join : 2
  - Non-realtime work call : 3 (Provisioning, Central office work, Outside work) → orchestration dehydration occurs.

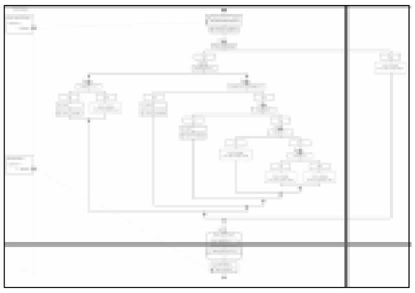


Fig.10 Main Orchestration

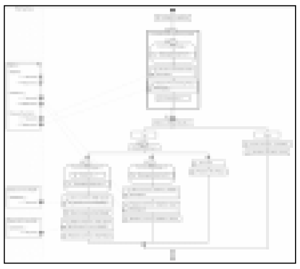


Fig.11 Sub Orchestration

New features in BizTalk 2004 Orchestration design are as follows:

•Business Activity Monitoring: Give information workers a real-time view of running business processes with the Microsoft Office tools

•Real-time Tracking: Follow the real-time progress of documents and processes in your BizTalk Server applications.

•Human-based Workflow: Integrate people and processes with a single orchestration engine.

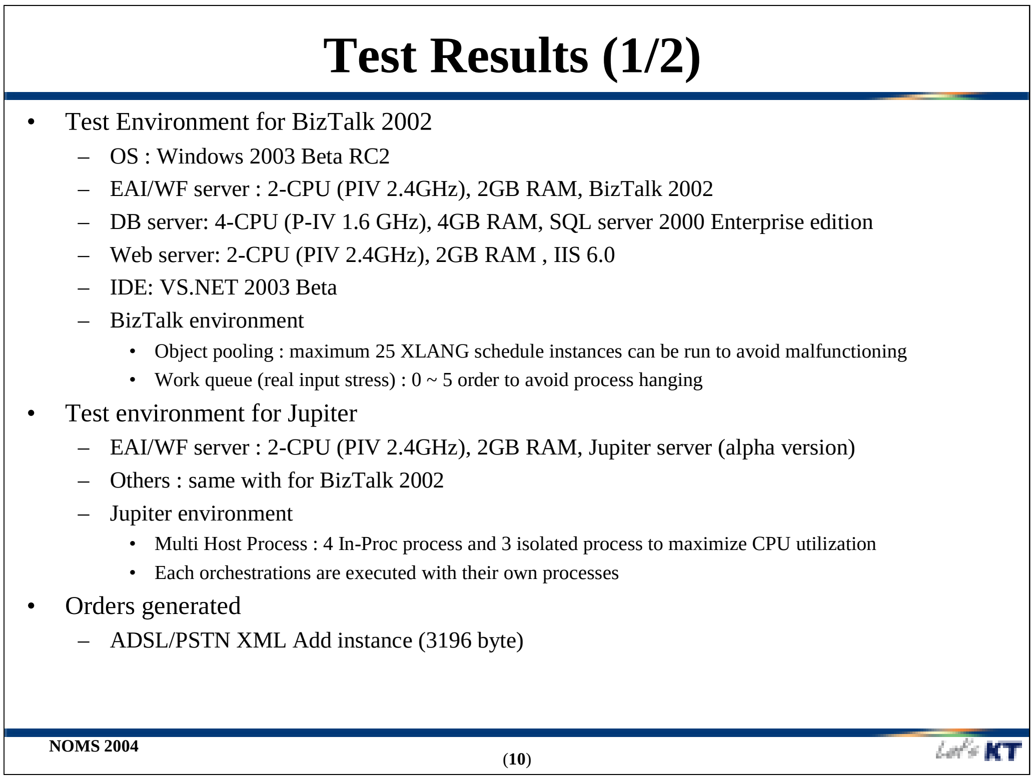
•Business Process Execution Language (BPEL) : Simplify cross-platform interoperability for process orchestration with standards developed in conjunction with other industry leaders.

•Business Rules : Dynamically change business processes to maximize organizational flexibility.

•Sub-orchestration design, Expand/Collapse Group assignment, Technical and business part separation, Debugging

Four newly added functionalities are especially valuable in implementing process. First, sub-orchestration can be designed in one project. In BizTalk 2002 there can be just one orchestration diagram in one Visio file. So when we extract common orchestration, we cannot reuse it. We have to draw it again and again. Second, expand/collapse functionalities are added. This makes our complex diagrams more visible, recognizable and easy to draw. Third, technical design and business logic design are separated. In BizTalk 2002, workflow designer must have some knowledge about COM+ components, BizTalk messaging and script components, etc. This separation liberate the OSS business process designer from technical pressure. The last thing is with Jupiter server step-by-step bebugging becomes possible. We can find our mistakes or typos very easily with debugger.

The implementation summary to process ADSL/PSTN sample service order are shown above. The logical characteristics are about the same as in using BizTalk 2002. But the diagrams shown are very different.



Test Results (1/2)

- Test Environment for BizTalk 2002
  - OS : Windows 2003 Beta RC2
  - EAI/WF server : 2-CPU (PIV 2.4GHz), 2GB RAM, BizTalk 2002
  - DB server: 4-CPU (P-IV 1.6 GHz), 4GB RAM, SQL server 2000 Enterprise edition
  - Web server: 2-CPU (PIV 2.4GHz), 2GB RAM , IIS 6.0
  - IDE: VS.NET 2003 Beta
  - BizTalk environment
    - Object pooling : maximum 25 XLANG schedule instances can be run to avoid malfunctioning
    - Work queue (real input stress) : 0 ~ 5 order to avoid process hanging
- Test environment for Jupiter
  - EAI/WF server : 2-CPU (PIV 2.4GHz), 2GB RAM, Jupiter server (alpha version)
  - Others : same with for BizTalk 2002
  - Jupiter environment
    - Multi Host Process : 4 In-Proc process and 3 isolated process to maximize CPU utilization
    - Each orchestrations are executed with their own processes
- Orders generated
  - ADSL/PSTN XML Add instance (3196 byte)

NOMS 2004 (10)

Test Environment for BizTalk 2002 are as follows:

•OS : Windows 2003 Beta RC2

•EAI/WF server : 2-CPU (PIV 2.4GHz), 2GB RAM, BizTalk 2002

•DB server: 4-CPU (P-IV 1.6 GHz), 4GB RAM, SQL server 2000 Enterprise edition

•Web server: 2-CPU (PIV 2.4GHz), 2GB RAM , IIS 6.0

•IDE: VS.NET 2003 Beta

•In BizTalk 2002 environment, Object pooling must be used to limit maximum 25 XLANG schedule instances can be run to avoid malfunctioning. If the number of running instances exceeds reasonable number, the system will fall into a coma status. And work queue length, which means real input stress are maintained 0 ~ 5 order to avoid process hanging. Microsoft suggests about 20, but 0 ~ 5 is better in our H/W configurations.

Test environment for Jupiter are the same as above

•EAI/WF server : 2-CPU (PIV 2.4GHz), 2GB RAM, Jupiter server (alpha version)

•Others : same with for BizTalk 2002

•In Jupiter environment, multi host process (4 In-Process process and 3 isolated process to maximize CPU utilization) are used, so each orchestrations are executed with their own processes

Orders generated are

•A series of ADSL/PSTN XML Add instance (3196 byte)

Test Results (2/2)

- Throughput
  - 18.5 times better in Jupiter server
  - BizTalk 2002 is slower because it is based on XLANG interpreter
  - Very high orchestration dehydration costs for non-realtime functions
    - Performance degraded by 26.88 times in BizTalk 2002 by dehydration
    - In Jupiter, dehydration automatically occurs (cannot be controlled)
- Processing time
  - 2.94 to 8 times better in Jupiter server
  - BizTalk 2002 is slower because it is based on XLANG interpreter
- Number of average running instances
  - are almost same
  - Not an important factor

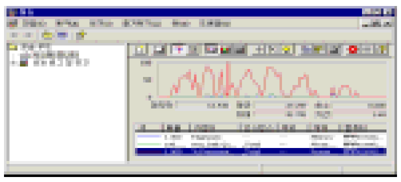


Fig.12 CPU utilization of Jupiter server

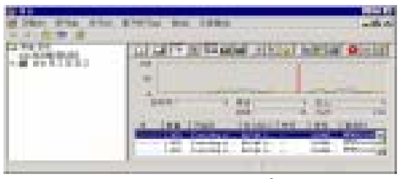


Fig.13 Average instances of Jupiter server

Measure	BizTalk 2002	Jupiter
Throughput	3 orders/min.	55.5 orders/min.
Processing Time	1 ~ 8 minutes	0.34 ~ 1 minutes
Average running instances	20 instances	18 instances

Table.2 Performance comparison

Performance test is very important because that is a main feature to choose a platform or tool. Especially EAI/Workflow engine is a core component of OSS implementation. The results for throughput are shown in Table 2 and as follows:

•18.5 times better in Jupiter server

•BizTalk 2002 is much slower because it is based on XLANG interpreter

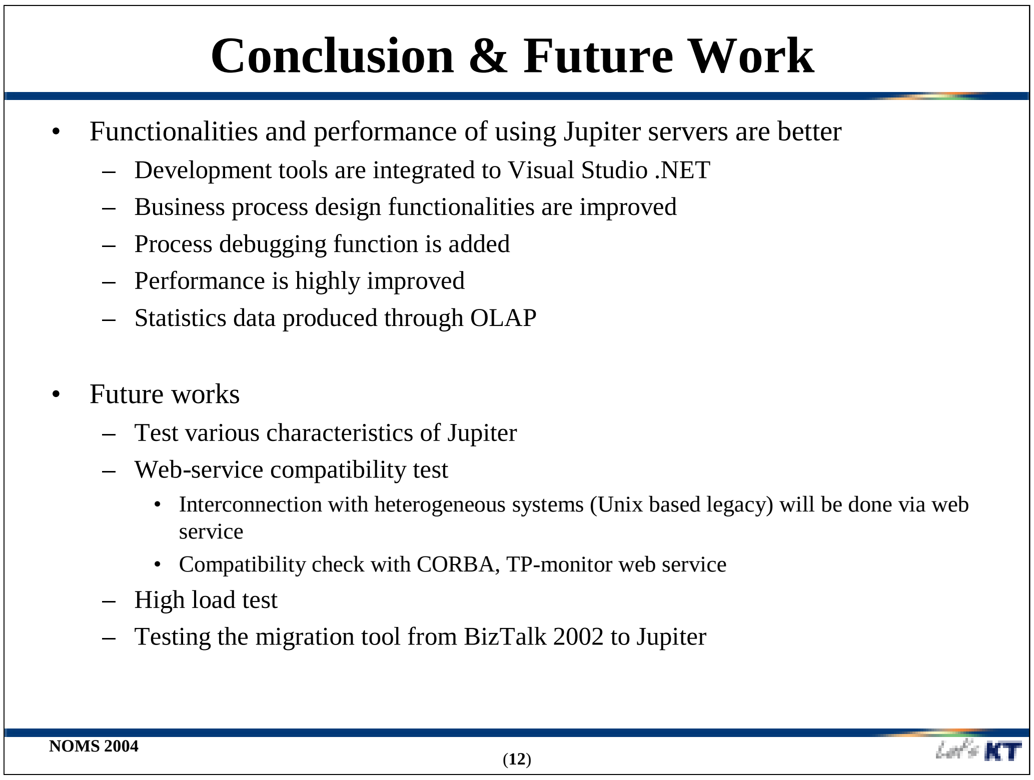
•Very high orchestration dehydration costs for non-realtime functions : Performance degraded by 26.88 times in BizTalk 2002 by dehydration. In Jupiter, dehydration automatically occurs (cannot be controlled), but it does not degrade performance very much.

The results for processing time are as follows:

•2.94 to 8 times better in Jupiter server

•BizTalk 2002 is slower because it is based on XLANG interpreter

•Number of average running instances are almost same



Conclusion & Future Work

- Functionalities and performance of using Jupiter servers are better
  - Development tools are integrated to Visual Studio .NET
  - Business process design functionalities are improved
  - Process debugging function is added
  - Performance is highly improved
  - Statistics data produced through OLAP
- Future works
  - Test various characteristics of Jupiter
  - Web-service compatibility test
    - Interconnection with heterogeneous systems (Unix based legacy) will be done via web service
    - Compatibility check with CORBA, TP-monitor web service
  - High load test
  - Testing the migration tool from BizTalk 2002 to Jupiter

NOMS 2004

(12)

Conclusion and Future work

In this paper, we have presented the functionalities and performance of BizTalk 2002 and Jupiter covering EAI/workflow based OSS implementations. We also presented the the architecture using these two engines. We have argued that the Jupiter based approach is superior to BizTalk 2002 in many aspects. Jupiter is functionally richer, since it provides easier and well integrated development tools such as orchestration designer and process debugger. We have also presented the performance test results using implementations with both engines.

In further works, we will test various characteristics of Jupiter, web-service compatibility with heterogeneous systems (Unix based legacy), CORBA and TP-monitor based web service. We will also test in highly loaded environment.

References

1. Jae-Oh Lee, "Enabling Network Management Using Java Technologies", IEEE Communications Magazine, Jan. 2000.
2. Seong Ik Hong, Young Il Kim, Woo Sung Kim, “3 tier OSS architecture with .NET platform”, APNOMS 2003, Oct. 2003.
3. J.G.Brinsfield, “Unified Network Management Architecture (UNMA)”, ICC 1988.
4. John-Luc Bakker and Ravi Jain, “Next Generation Service Creation Using XML Scripting Languages”, pp.2001 – 2007, ICC 2002.
5. “OSS Java Community Process Program”, http://jcp.org/jsr/tech/oss.jsp
6. “OSS Working Group”, http://www.microsoft.com/SERVICEPROVIDERS/ossbss/osswg.asp
7. “BizTalk 2002 overview”, http://www.microsoft.com/biztalk
8. “New features in Jupiter”, http://www.microsoft.com/biztalk/beta
9. “The Essential Client/Server Survival Guide”, 2nd edition, Robert Orfali, Dan Harkey, Jeri Edwards, Wiley