基于消息的SOA企业解决方案

第四章 ESB产品简介

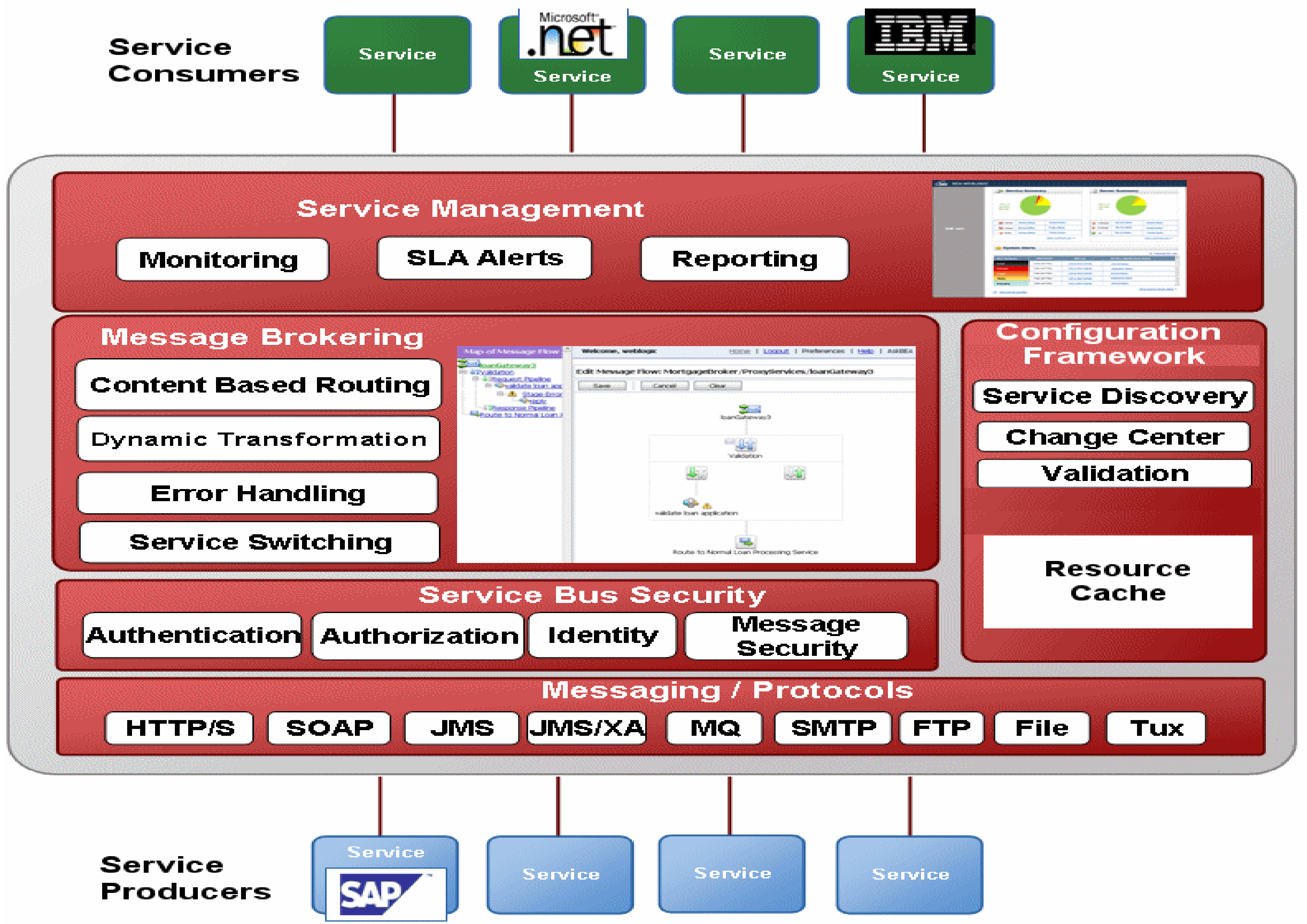
本章内容

1. Oracle Service Bus (OSB)
2. IBM的ESB产品
3. 开源ESB(Jboss,Mule,ServiceMix)

主流商业和开源ESB产品

类型	产品	公司
商业	Oracle Service Bus (OSB)	Oracle
	Oracle Enterprise Service Bus (ESB)	
	WebSphere Enterprise Service Bus	IBM
	WebSphere Message Broker	
	WebSphere DataPower	
	Sonic ESB	Progress
	ActiveMatrix Service Bus	TIBCO
开源	JBoss ESB	JBoss
	Mule	MuleSoft
	ServiceMix/FUSE ESB	Progress
	Synapse/WSO2 ESB	WSO2

Oracle Service Bus (OSB)的架构图



OSB的发展趋势

- 易用性增强
  - 开发工具从Web Console迁移到Eclipse，支持图形化拖拽和便于调试
- 性能提升
  - 嵌入Oracle Coherence(企业级的内存数据网格）产品，在特定场景下为服务调用提供缓存，性能提升80%。
- 管控能力增强
  - 采用自动化的生命周期服务治理，从服务设计、开发、部署和运行期的整个服务生命周期内和Enterprise Repository产品进行自动同步，无需人工干预。

OSB的优点

- 易用性
  - 在studio上直接集成测试功能，比如studio能提供直接发送和接收SOAP,JMS消息的功能，无需借助第三方工具，如SoapUI和编写JMS客户端代码。
- 性能
  - 采用Cache机制，为静态响应信息提升性能。静态响应信息是指在一段时间内不会发生变化的信息，如天气预报，手机套餐，人民币汇率等，这些数据变化的周期通常是1天，1月。
- 实现手段
  - 采用比较成熟的开源Memcached或者轻量级的JCACHE。

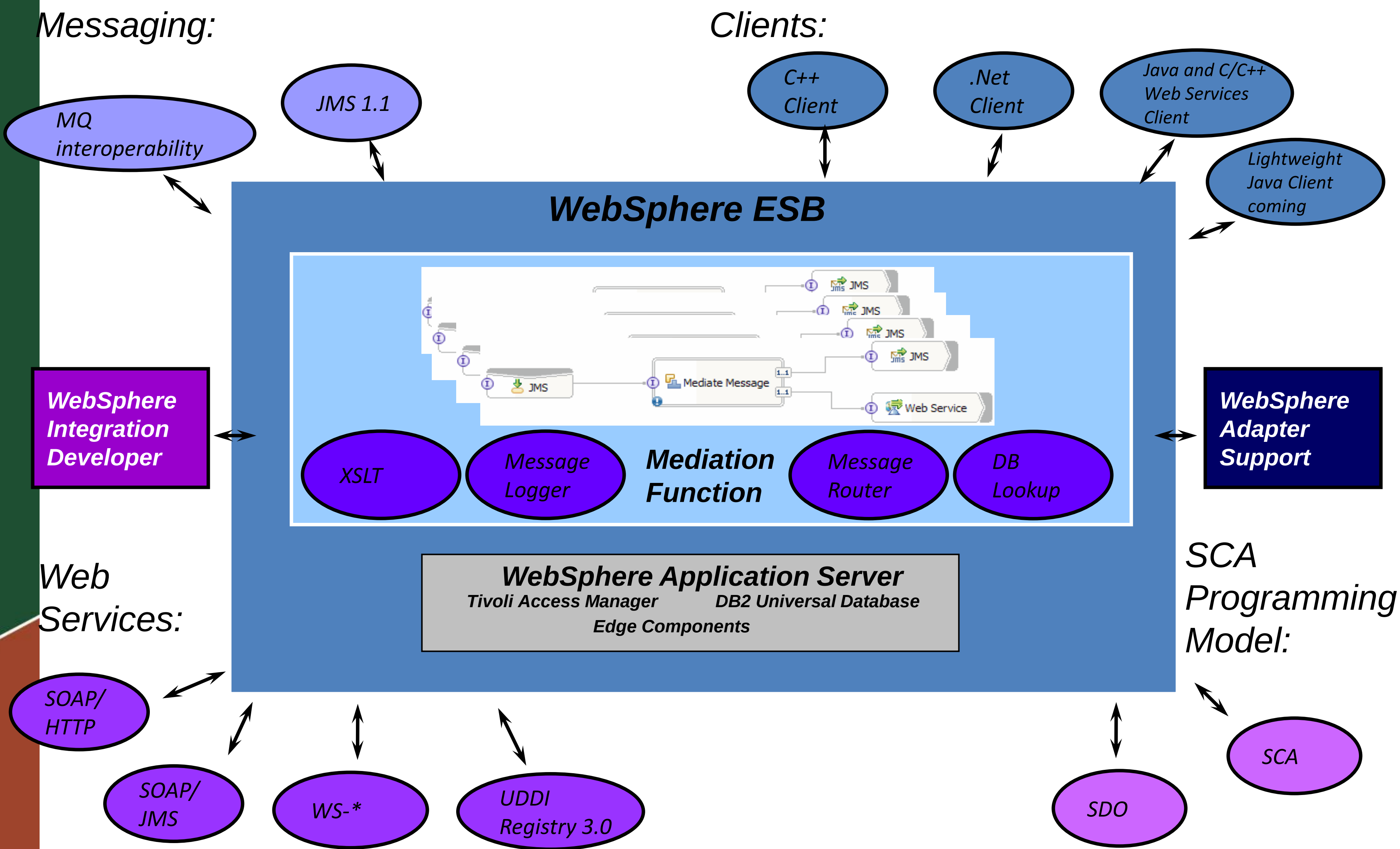
OSB的缺点

- 依赖于Weblogic
- 重量级的统一消息格式
  - OSB将各种协议（HTTP,WS,JMS…）接入的消息统一转换为SOAP Message，再通过Xquery Engine对SOAP Message进行XML操作。
  - 不适合处理以下场景：
    - 对HTTP下的大数据包进行XML操作比较耗CPU
    - 将JMS Object类型的大数据包转换为XML是繁重的操作

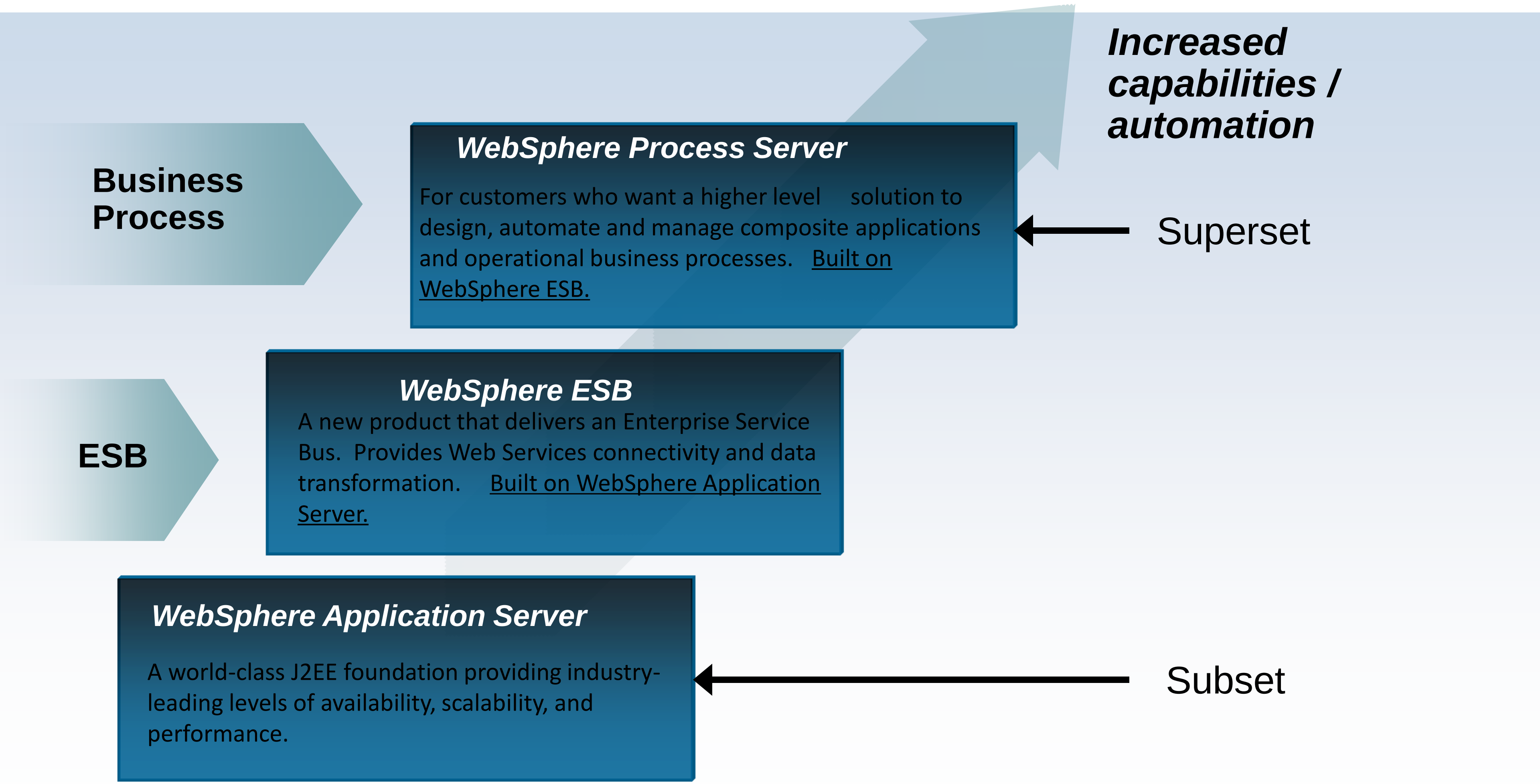
IBM的ESB产品

- WebSphere ESB(标准的企业服务总线)
  - WebSphere ESB是基于平台的ESB，并使用 WebSphere 应用服务器针对集成的SOA平台进行了优化。
- WebSphere Message Broker(高级企业服务总线)
  - WebSphere Message Broker是与平台无关的ESB，是在异构IT环境中为通用连接和转换构建的。
- WebSphere DataPower SOA Appliances(企业服务总线Gateway)
  - WebSphere DataPower SOA Appliances是同时包含硬件和软件的产品，提供了大量的重要功能：XML加速、安全措施执行和ESB功能。

WebSphere ESB的架构



WebSphere ESB的位置



WESB实现SOA ESB解决方案的特点

- 消息转换
  - WESB所处理的消息为XML格式的数据
  - 在WESB的消息流中，数据以SMO(IBM对SDO的扩展)形式存在，WESB可以对XML消息树的内容进行修改，包括改变某个节点的内容，增加新的节点以及删除某个节点等等。
- 支持的协议
  - WESB支持符合SOA标准的协议，比如 SOAP/HTTP、SOAP/JMS、WSDL V1.1、UDDI V3.0，WebSphere MQ 等
  - WESB目前只支持SOAP方式来描述服务，传输协议可以是HTTP、JMS、WebSphere MQ连接。对于多传输协议的基础，建议使用MB来做ESB的解决方案。

WESB实现SOA ESB解决方案的特点

- 消息路由
  - 开发环境WID中提供了一个节点专门来负责消息路由，WID也提供了良好的对路由规则定义的开发支持，开发人员可以很容易的定制负责的路由规则。
  - 若要实现动态路由的功能，则需要和一个服务的存储单元中来动态的查找服务，目前WSRR是一款优秀的提供该功能的工具。WESB从V6.1开始提供了Endpoint lookup节点来支持WSRR的集成，简化了开发过程。
  - 如果要实现简单的动态服务路由的功能，则可把服务的定义存放在数据库中，在WESB中通过DB lookup来查找服务的Endpoint，然后注入到消息流中，WESB V6.1之前的版本就已经支持与数据库的集成。

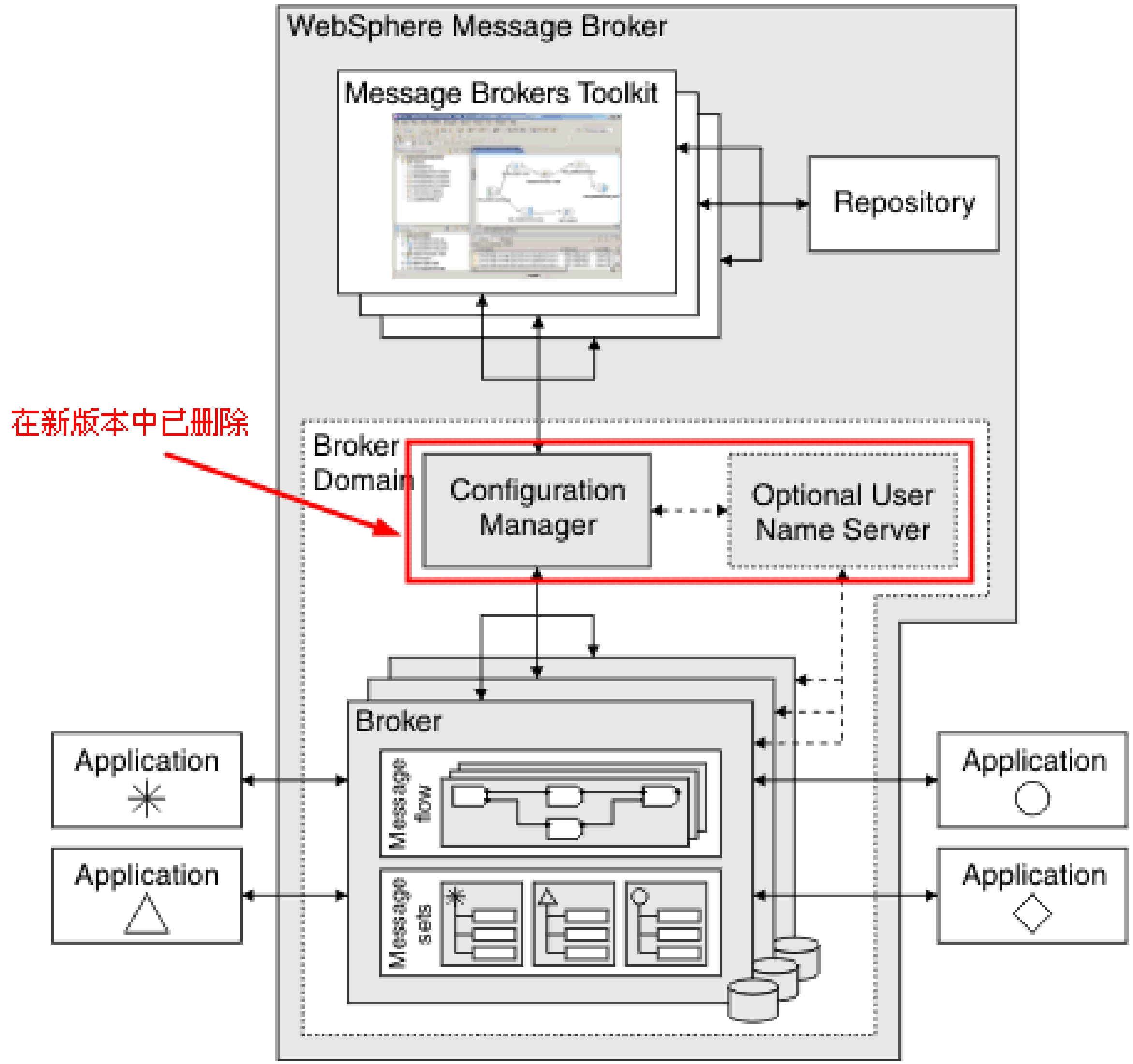
WESB实现SOA ESB解决方案的特点

- 对Web Service的支持
  - WESB 天生运行在 J2EE 的环境里面，对 Web Service 有着天然的支持。
- 事件处理
  - 在消息流中，我们需要跟踪消息各节点的状态，以满足统计和出错处理的要求，在 WESB 中，通过 CEI 机制来处理消息。
- 与遗留系统的集成
  - WESB 通过 Adapter 与遗留系统进行集成，支持 IBM Websphere Adapter 和 JCA Adapter，通过 JCA 我们就可以将遗留系统里面的服务和数据通过标准（SCA/SDO）的形式整合到集成环境中。

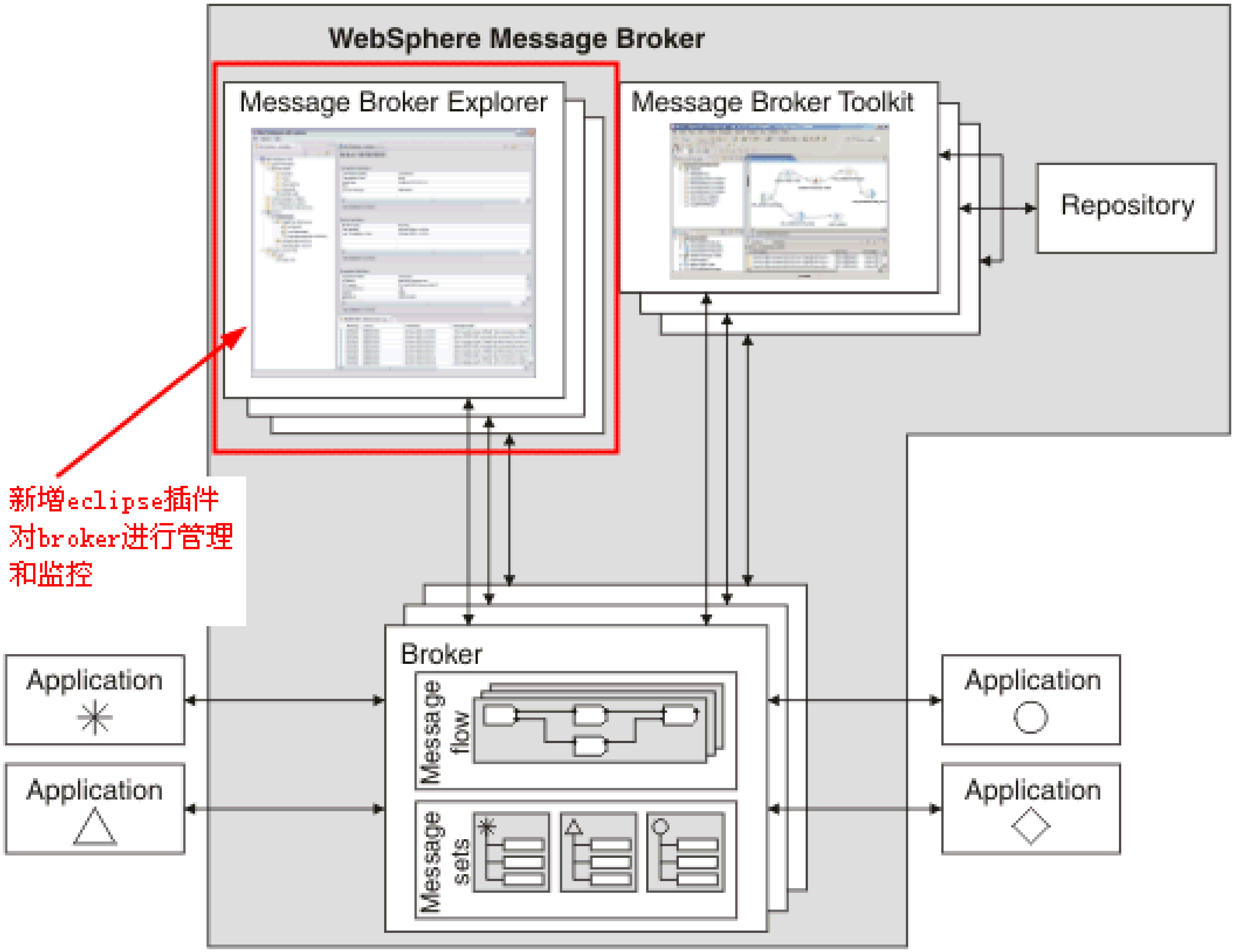
WESB实现SOA ESB解决方案的特点

- 安全方面的支持
  - WESB 没有对安全做特殊的处理，使用 WAS 的安全支持来实现 ESB 的安全。
- 性能
  - WESB 是一个纯 Java 的应用，运行效率上有些限制，同时可以处理的消息流的数量级为几十到几百之间。
- 开发和部署
  - 开发工具是 WID，一个 ESB 的消息流在 WID 中被称为 Mediation Module，它是一个 J2EE 应用，开发和部署工作无异于普通的企业级应用。

WebSphere Message Broker的架构 (V6.0)



WebSphere Message Broker的架构(V7.0)

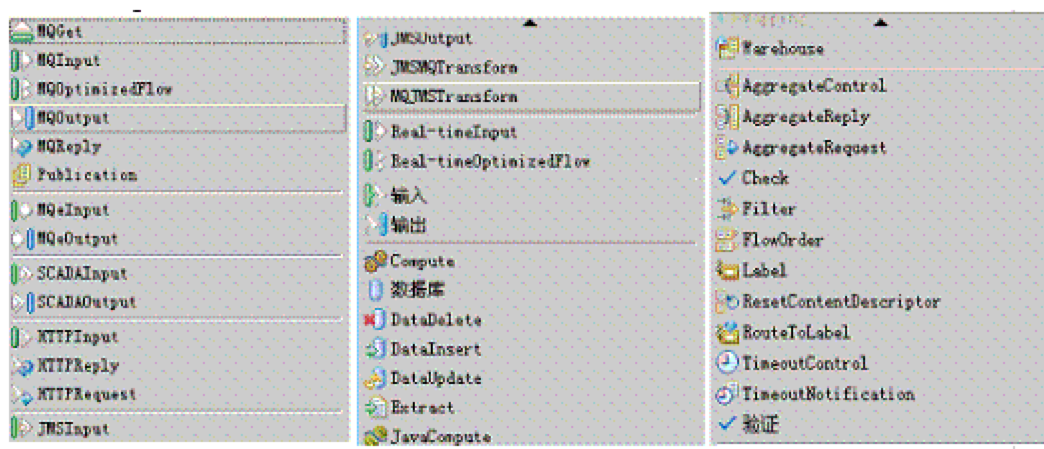


WMB实现SOA ESB解决方案的特点

- 消息转换
  - WMB 在消息处理方面功能非常强大，对 XML 格式的消息或非 XML 格式的消息，如 C Header，COBOL 等都有很好的支持。
  - 通过开发消息对应的消息集（Message Set），可以在消息流中对任意格式的消息进行修改。WMB 提供了内置的 mapping 和 database 节点，用户可以通过图形化的方式方便的实现消息的转换或与数据库的交互。

WMB实现SOA ESB解决方案的特点

- 支持的协议
  - WMB支持所有 WESB 支持的传输协议，除了常用的 HTTP、JMS 等，还对 FTP、Socket、Mobile、Telemetry、Biztalk 和 Tuxedo 等有良好的支持。WMB 与 MQ 有着紧密的联系。
  - WMB 内置的功能节点对这些协议提供了很好的支持，仅需配置即可。



WMB的内置开发节点

WMB与WebSphere MQ的关系

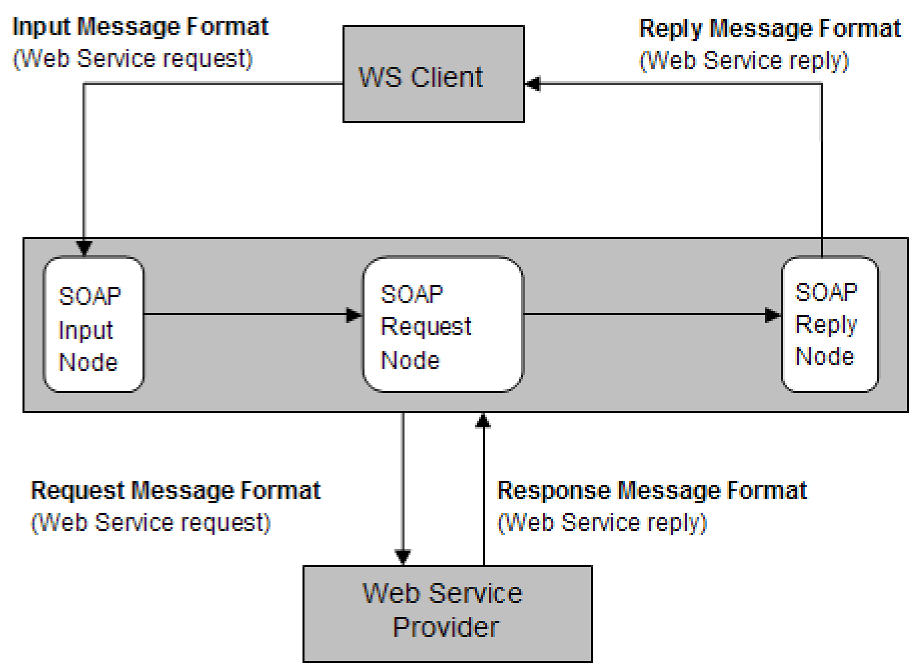
- WebSphere Message Broker和WebSphere MQ可满足不同的业务需求。
  - WebSphere MQ在应用程序和系统之间提供安全而又可靠的连接，支持80多个平台配置。这使得能够在几乎所有业务环境（可能要在业务基础设施中部署之间移动未更改的数据。
  - WebSphere Message Broker可用作在应用程序之间移动数据的传输工具，但是它通过了解数据格式还能够执行其他任务，并可以提供XML数据格式的智能路由和转换。企业仍需要 WebSphere MQ连接多个不同的环境，这些环境构成围绕企业部署的典型IT基础设施，但是也可能需要ESB增加环境的价值，ESB可以对在基于标准的应用程序之间交换的结构化数据起到较好作用

WMB实现SOA ESB解决方案的特点

- 消息路由
  - WMB 提供了很多功能强大的内置节点支持消息的路由，如 Filter 节点、Label 节点等，在新版本的 WMB 中又引入了 Router 节点，该节点几乎和 WESB 中的 Router 节点一样。
  - 若需要实现动态路由，可以使用 WSRR 作为服务的存储，WMB 和 WSRR 有很好的集成，通过 RegistryLookup 和 EndpointLookup 我们可以在消息流中实现动态路由。
  - WMB 可以提供一个消息输入，多个消息输出的功能，可以实现一个消息同时路由到多个输出端。

WMB实现SOA ESB解决方案的特点

- 对Web Service的支持
  - 在WMB中，消息流可以作为Web Service暴露出去供外部调用，也可以作为客户端调用外部提供的Web Service。WMB不仅提供了内置的 SOAPRequest、SOAPInput 等节点实现对Web Service的支持，而且对WS扩展，如 WS-Security 和 WS-Addressing 也有良好的支持。



WMB实现SOA ESB解决方案的特点

- 事件处理
  - 在WMB中可以通过Trace Service来记录所发生的事件。Trace分两种，一种是User Trace记录消息流级别的事件，另一种是Service Trace，可以记录整个WMB的事件，如Broker的部署执行。WMB提供了Trace内置节点，可以实时的记录流程中消息内容的变化。
- 与遗留系统的集成
  - WMB对遗留系统有良好的支持。对SAP，PeopleSoft等大型的EIS系统，直接提供了内置的节点，通过JCA Adapter配置的方式和EIS系统交互。对于比较特别的遗留系统，如CICS、VSAM等，WMB提供了丰富的SupportPacs，客户可以下载并安装。

WMB实现SOA ESB解决方案的特点

- 安全方面的支持
  - WMB本身提供了两个层次上的安全，一个是部署时安全性，管理部署bar文件到Broker以及运行WMB管理命令的权限控制；另一个是运行时安全，涉及的权限控制包括发送消息到相应的消息流，以及消息流可以访问哪些MQ资源和非MQ资源，如数据库系统。
- 性能
  - WMB底层是用C++开发的，在性能上相对于WESB有很大提高。同时可以处理的消息数量级为几千到几万之间
- 开发和部署
  - 开发工具是WMB Toolkit，开发的消息流和消息集被打成bar文件通过配置管理器部署到Broker中。

WMB实现SOA ESB解决方案的特点

- 安全方面的支持
  - WMB本身提供了两个层次上的安全，一个是部署时安全性，管理部署bar文件到Broker以及运行WMB管理命令的权限控制；另一个是运行时安全，涉及的权限控制包括发送消息到相应的消息流，以及消息流可以访问哪些MQ资源和非MQ资源，如数据库系统。
- 性能
  - WMB底层是用C++开发的，在性能上相对于WESB有很大提高。同时可以处理的消息数量级为几千到几万之间
- 开发和部署
  - 开发工具是WMB Toolkit，开发的消息流和消息集被打成bar文件通过配置管理器部署到Broker中。

WMB实现SOA ESB解决方案的特点

- WMB提供的开发模式
  - 将常用场景模式化，比如服务穿透，studio自动生成配置文件，自动完成服务开发和服务组装的所有工作，用户只需填入参数。

File Processing	MQ one-way	Record Distribution to WebSphere MQ: one-way
Service Virtualization	Static endpoint	Service Proxy: static endpoint
Service Enablement	MQ one-way with acknowledgment	Service Facade to WebSphere MQ: one-way with acknowledgement
Message Based Integration	MQ request-response with persistence	Message Correlator for WebSphere MQ: request-response with persistence
Application Integration	MQ one-way (IDoc)	Data distribution SAP to WebSphere MQ: one-way (for IDoc)

WebSphere DataPower SOA Appliances 产品

- WebSphere DataPower Integration Appliance XI50 & Integration Blade XI50B
  - IBM的硬件ESB XI50和 XI50B是为实现某用途而特别设计的，旨在简化部署、加强安全性、桥接多种协议，并执行线速转换。
- WebSphere DataPower B2B Appliance XB60
  - XB60能够利用特制的B2B硬件来扩展ESB，在一个高性能的DMZ就绪的设备中提供AS1/AS2/AS3消息传递和贸易合作伙伴配置文件管理
- WebSphere DataPower Low Latency Appliance XM70
  - XM70允许在产生极少延迟的情况下在现有网络上路由大量的消息
- WebSphere DataPower XML Security Gateway XS40
  - XS40 由全球顶级的XML和Web服务安全性专家设计，提供了一整套可配置的安全性和策略实施功能。
- WebSphere DataPower XML Accelerator XA35
  - XA35是真正能够帮助减轻超负荷的 Web 和应用服务器的负担，方法是以线速处理 XML、XSD、XPath 和 XSLT。确保快速地从应用投资中获得回报。

DataPower实现SOA ESB解决方案的特点

- 消息转换
  - Datapower对XML消息有强大的支持，但是不仅仅支持XML，可以在Policy中使用Transformation节点来对消息进行任意我们需要的转换，是使用XSLT来实现的，开发人员定义自己XSLT Stylesheet并在Transformation节点中指定，Datapower负责转换。
- 支持的协议
  - Datapower支持以下传输协议：HTTP，HTTPS，WebSphere MQ、WebSphere JMS，TIBICO EMS，FTP Poller,FTP Server,NFS等等。Datapower的MPGW就是一个处理不同协议的应用系统的互联的服务对象。

DataPower实现SOA ESB解决方案的特点

- 消息路由
  - Datapower 支持对服务和消息的路由，根据消息流中的上下文连接将消息动态的分发到不同的消息提供者。但是 Datapower 的动态路由和 WESB 以及 MB 的动态路由还是有区别的，Datapower 的动态路由需要由开发者定义路由的 Map，而 WESB 和 MB 支持在消息头的属性里动态的设置 Endpoint 的地址。目前 Datapower 可以和 WSRR 集成来定义 WS-Proxy（Datapower 中的一种服务对象），但不支持直接和 WSRR 联合实现动态访问 Endpoint 的功能。

DataPower实现SOA ESB解决方案的特点

- 对Web Service的支持
  - Datapower的XMlFirewall和 WS-Proxy提供了强大的对Web Service的支持，而且提供了细粒度的对Web Service的控制，可以从服务级（Service)，端口级(port)，绑定级（binding)，操作级（operation）来对消息体进行控制。此外，对WS-Security也提供了强大的支持。
- 事件处理
  - Datapower中可以通过Probe的方式来跟踪消息流的中间状态，在Probe中可以看到消息流的每个节点的消息内容。Probe一般用于开发调试过程，在生产模式下一般不使用。Datapower不支持与CEI类似将消息发送到其他应用系统的机制。

DataPower实现SOA ESB解决方案的特点

- 与遗留系统的集成
  - Datapower 不支持和 Adapter 的连接，若要与遗留系统的集成，则需要通过其他中间件转换在遗留系统和 Datapower 之间做而桥梁来连接。
- 安全方面的支持
  - Datapower 的强大之处在于其对安全方面的强有力的支持，它提供对 XML-attack 的原生支持（关于 XML-attack 的知识参见参考资料）；此外，Datapower 可以对 Web Service 提供细粒度的安全支持，包括加密 (Encryption)，解密 (Decryption)，签名 (Sign) 和确认 (Verify)，以及 HTTS 方面的支持。这些支持在 Datapower 上开发起来都异常简单。

DataPower实现SOA ESB解决方案的特点

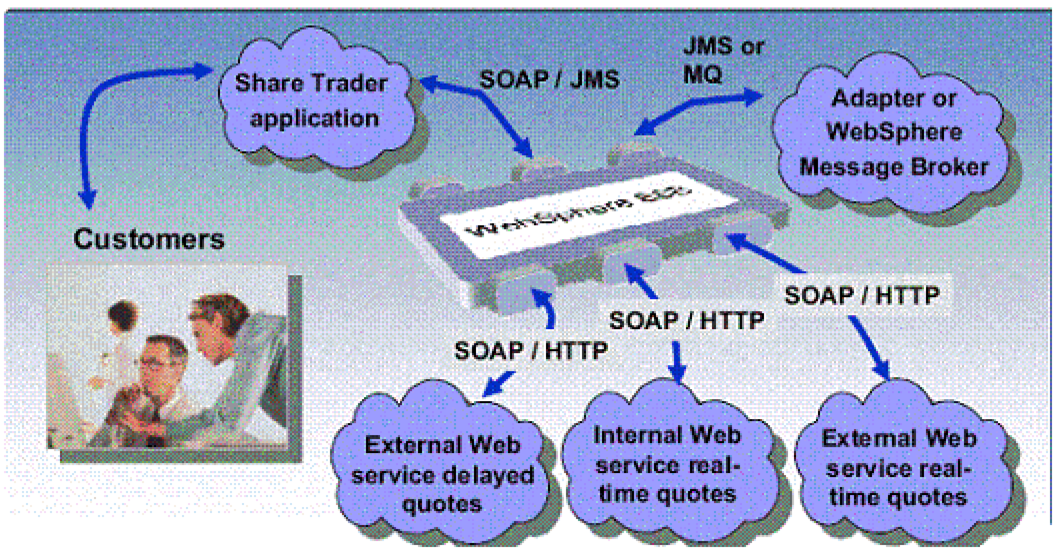
- 性能
  - Datapower无疑是三款ESB产品中性能最高的，对XML的处理速度达到线速。如果去除网络传输在其中的比例，对XML的处理性能所提高的倍数可达到上百倍。
- 开发和部署
  - Datapower 的没有相应的开发工具，但是提供了 Web GUI 的管理控制台和 CLI 方式的管理支持。我们在 Web GUI 下开发消息流，开发即部署。

IBM ESB产品的比较

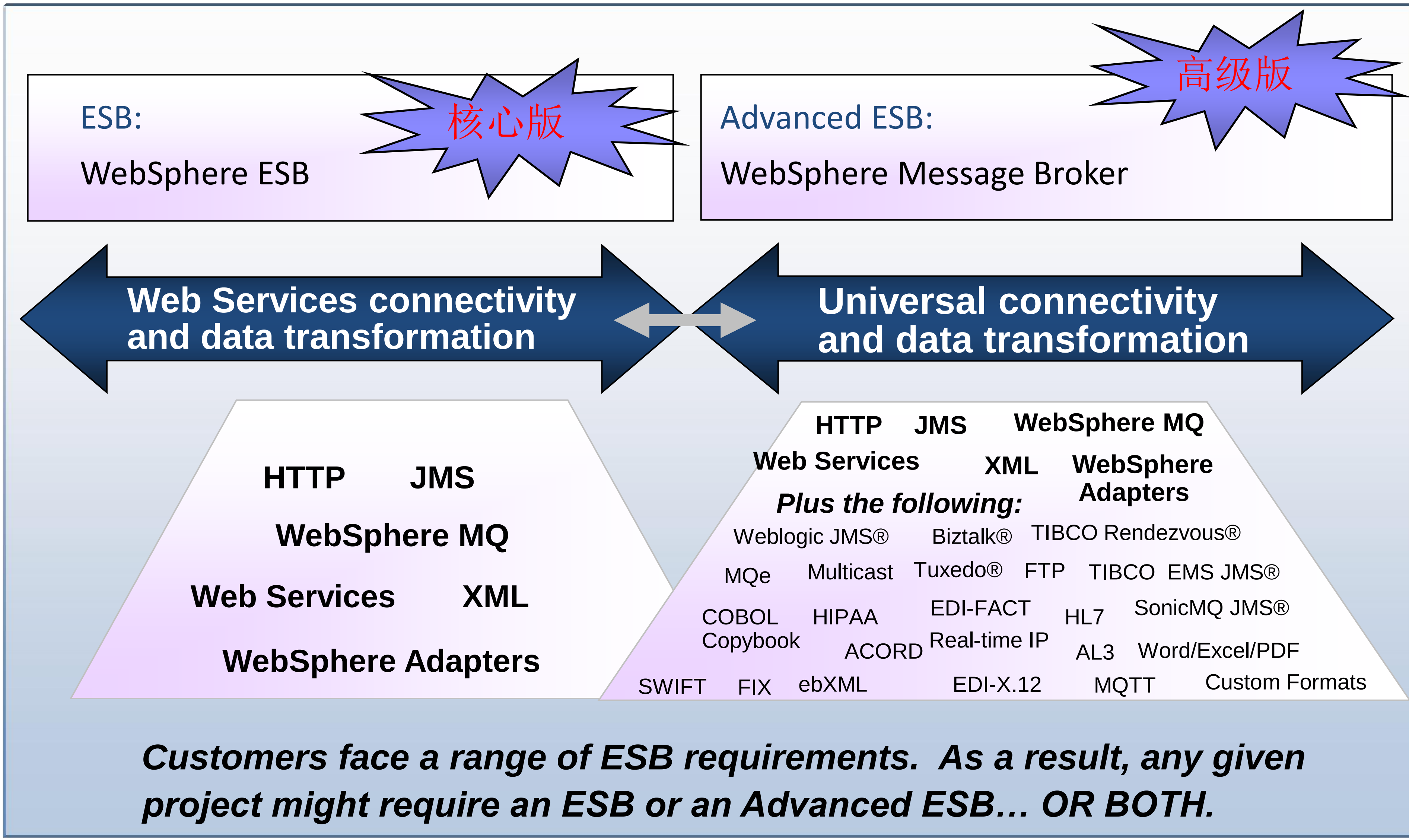
- 三款产品都提供了ESB必须的功能，但各有侧重：
  - WESB是一个轻量级的ESB，侧重于标准协议SOAP,JMS 等应用的基础，构建于WASND基础之上，提供了和 J2EE应用很好的集成功能；由于WESB是一个纯Java的应用服务器，在性能上相对较差，可并发执行的Mediation Flow的数量级在几十个左右。
  - WMB是一款高级的ESB，提供了比WESB多很多的传输协议，数据格式的支持，它所支持几乎大部分常用的数据格式和协议。并且WMB提供了良好的扩展功能，开发人员可以在WMB基础上开发自己的数据格式解析的节点。WMB使用C/C++编写，在处理性能上比ESB也要高出很多倍，可并发执行的流可以达到上百个或上千个。
  - Datapower是SOA中的又一重要的ESB，在WESB和WMB中都是用软件来实现XML解析和安全支持的，而Datapower使用硬件的XML解析和加速器，在性能上有了很大的提高。在安全和性能要求都比较高的环境中，Datapower是首选ESB，因为Datapower可以在实现高性能的同时也保证安全。

WESB的应用场景

- WESB适合使用于对性能要求不是很高，且遵循标准协议的SOA整合环境中。
- WESB 的优势是提供了和流程服务器 WPS 以及 J2EE 服务器 WAS 良好的整合。

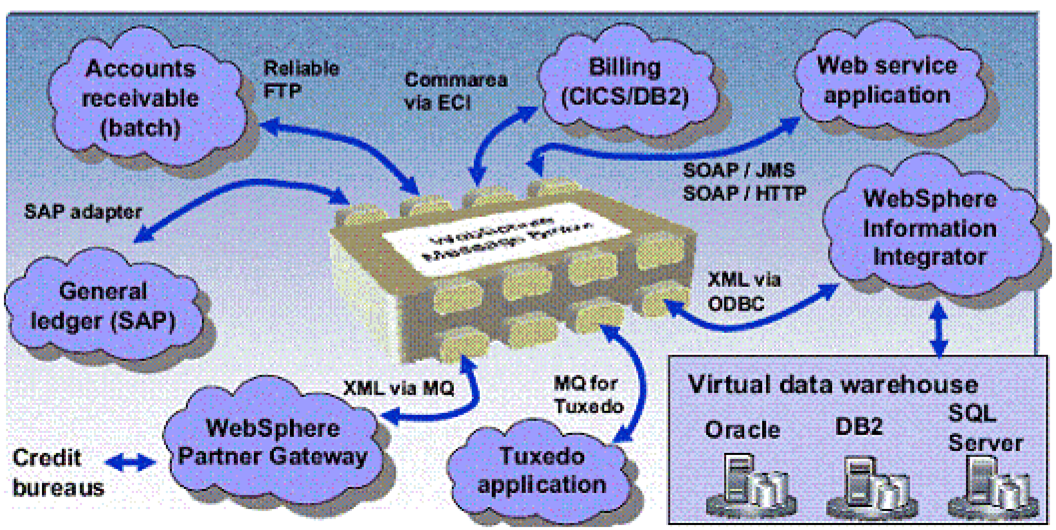


WESB与WMB的比较



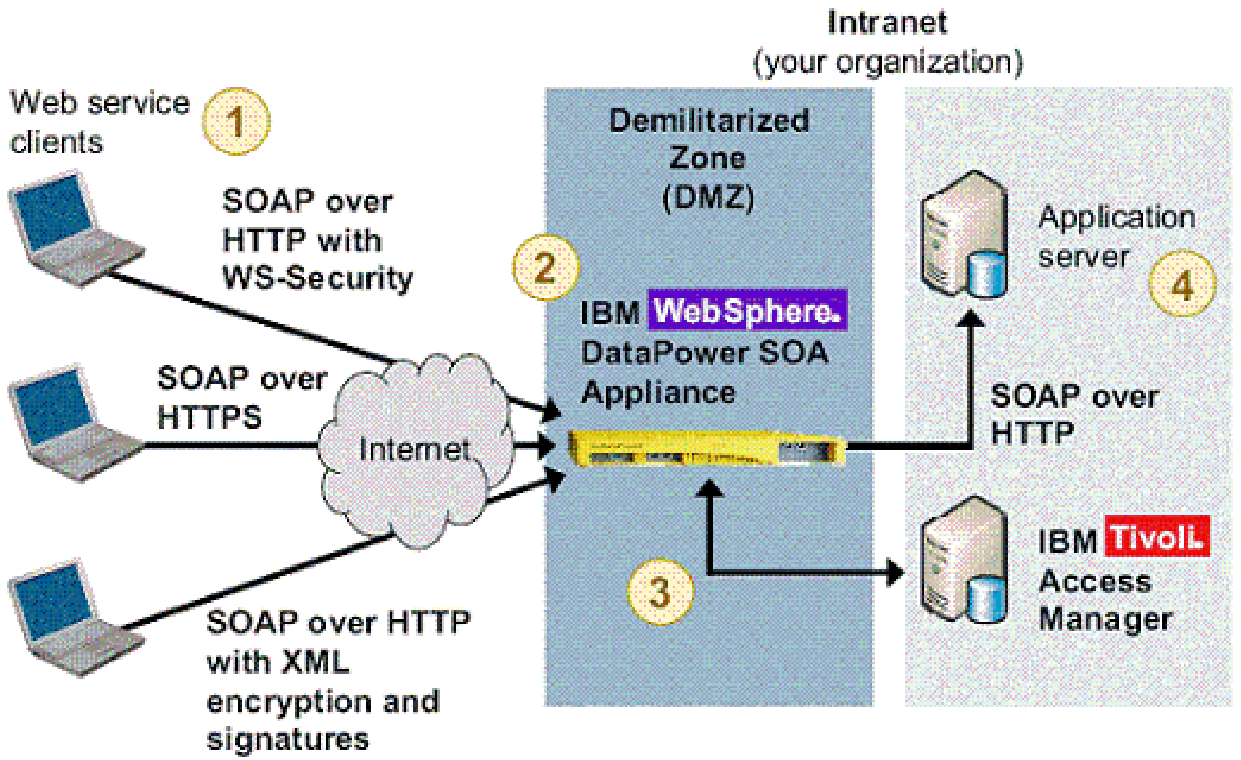
WMB的应用场景

- WMB应用于对性能要求相对较高，多种复杂协议存在的集成环境中。
- WMB构建于WebSphere Message Queue之上。WMQ提供了对异步消息提供了可靠的传送机制，比较适合于信息传输量较大，信息交互频繁的场景中



DataPower的应用场景

- 在安全和性能要求都比较高的环境中，Datapower是首选ESB，因为Datapower可以在实现高性能的同时也保证安全。



IBM ESB产品比较总结

ESB 功能特点	WESB 的支持	MB 的支持	Datapower
消息转换	XML	XML、非 XML	XML、非 XML
支持的协议	HTTP,JMS, WMQ 等	多达上百种	介于前二者之间
消息路由	强大，灵活	功能强大，灵活	灵活度比前二者稍弱
Web Service	强大的支持	支持 WS 扩展	强大的支持
事件处理	CEI，可以和外部事件消费系统监控	Trace Service	用于调试 Probe
遗留系统的集成	Adapter	丰富的 SupportPac	特定的遗留系统
安全	依赖 WAS 的安全	部署和运行时两个级别的安全	超强的安全支持
性能	几十到几百每秒	几千到几万每秒	达到线速
开发和部署	WID 集成开发环境	WMB Toolkit	WebGUI

IBM ESB产品在实际应用场景中的比较

- 业务场景
  - A银行最近和B银行及C银行形成合作关系，合作合同的一项指出，在其中任何一个银行有存款的用户，可以在其他任意两家银行用该存款作为贷款担保来获得一定倍数数量的贷款。如，若某人张三在A银行有1万元的存款，则该用户可以用这1万元的存款作为担保在B，C银行取得10倍于1万元（10万元）的贷款。A需要创新的解决方案，使得这项新的协议在 IT系统中实现，并服务于他们的客户。如果用户在B和C之一具有一定的存款，则A的解决方案将自动从B和C取得该客户的存款额，并将该存款额应用到贷款流程中
  - A银行决定使用SOA来个构建这一新的解决方案，利用IBM SOA Foundation来构建体系结构模型，用IBM ESB作企业服务总线，统一进行服务的注册，查找，路由等，并在ESB上运行其他应用程序。前提条件：B银行和C银行都已经向A公司提供了各自的获取某用户在本公司存款的服务，而且已经定义了良好的接口。

使用WESB实现业务场景

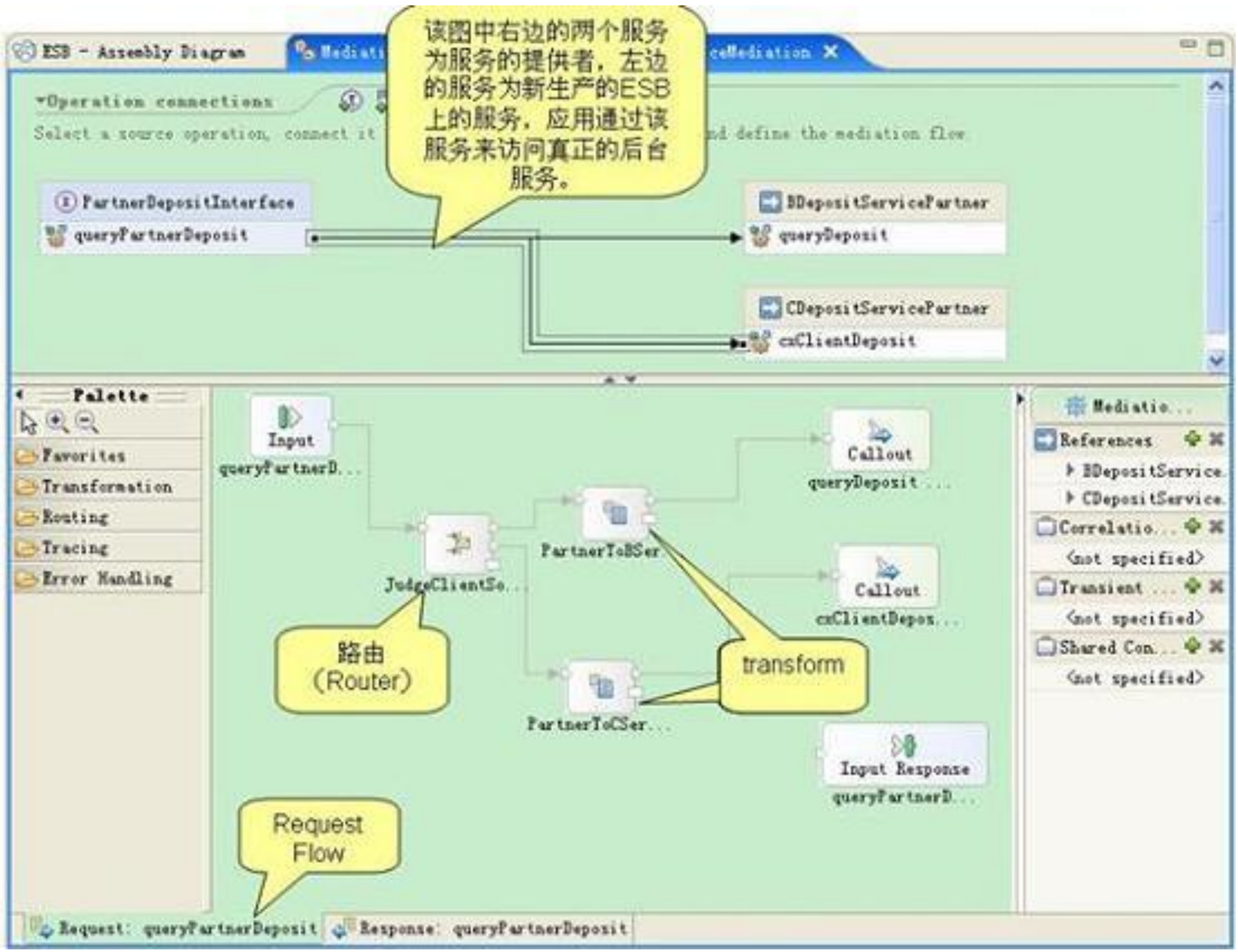
- 场景描述：
  - B、C银行提供的查询客户存款的服务已通过Web Service方式实现；
  - 并发的请求不会很多；
  - A银行的整合中多个应用都会使用到B、C银行提供的查询客户存款的服务；
  - 希望通过ESB向整合环境提供统一的、可复用的查询客户存款的服务，该服务自动根据客户的来源，动态路由到B、C提供的客户存款的服务。
  - B、C现有服务定义：

服务来源	接口	操作	输入	输出
B 银行	BDepositService	queryDeposit	Customer	DepositInfo
C 银行	CDepositService	cxDeposit	Client	ClientDeposit

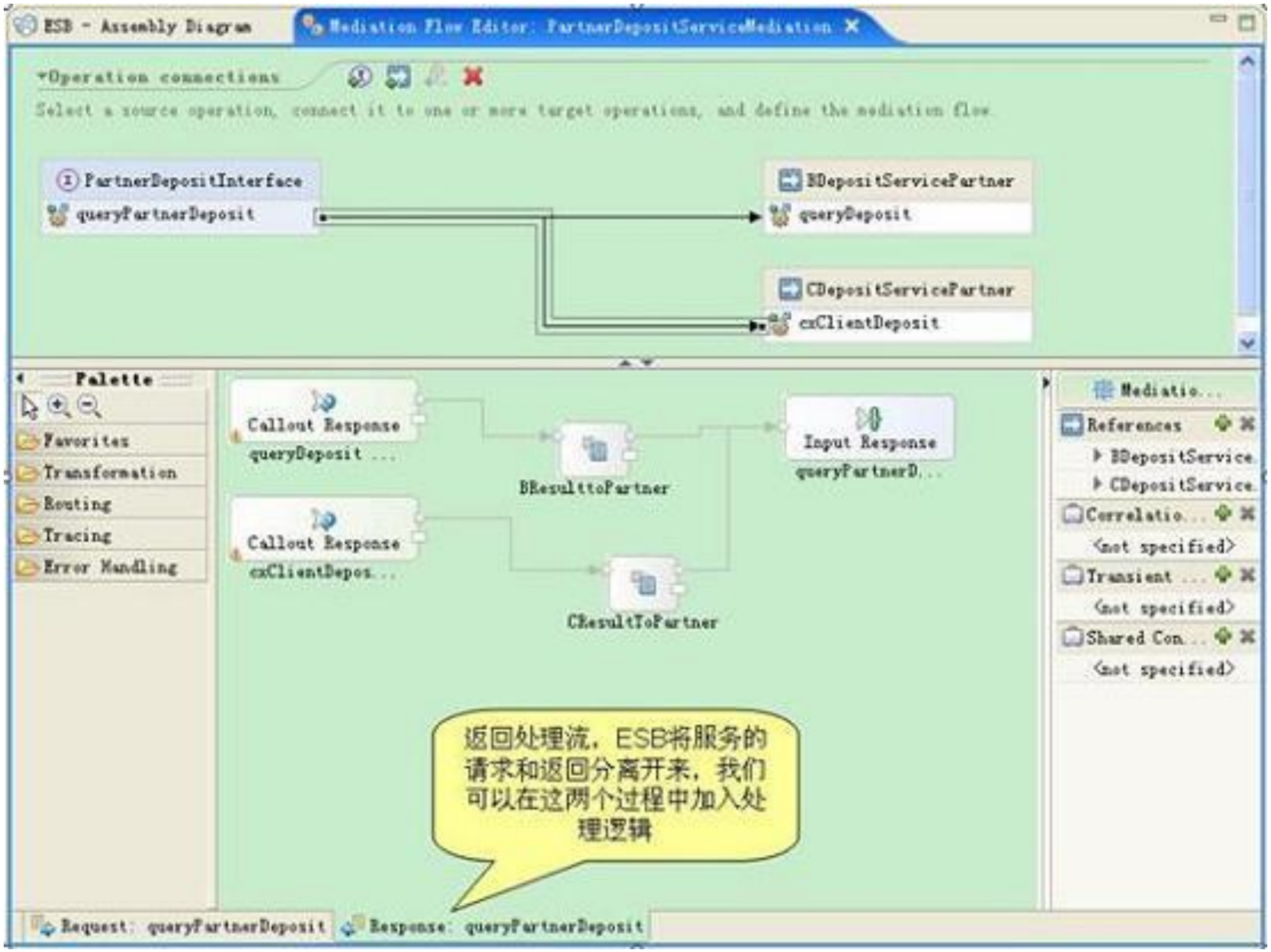
WESB开发的三个步骤

- 第一步：
  - 在 WID（集成开发环境中）将服务的提供者（B银行 C银行）引用到开发环境中，每一个服务对应于一个SCA Import，根据不同的服务提供者选择不同的绑定（Binding）这里由于服务都已Web Service方式提供，所以选用Web Service绑定。
- 第二步：
  - 通过一个Mediation模块来处理服务的路由，消息格式转换等。
- 第三步：
  - 将新的统一服务通过Web Service的方式Export出去，使用该Web Service的应用都通过该Export调用。

WESB Mediation模块的请求消息流



WESB Mediation模块的响应消息流



使用WESB实现业务场景的总结

- 将来自不同服务提供者的两个服务注册在WESB上，由WESB提供一个统一的服务；服务请求者不需要去关心具体应该调用由哪个后台服务；整个开发过程不需要写一行代码。
- WESB基于WAS J2EE容器之上，对安全事务处理等方面都有很好的支持，同时WESB遵循标准的SCA/SDO规范，使得我们开发的组件可以很容易的和其他应用集成。

使用WMB实现业务场景

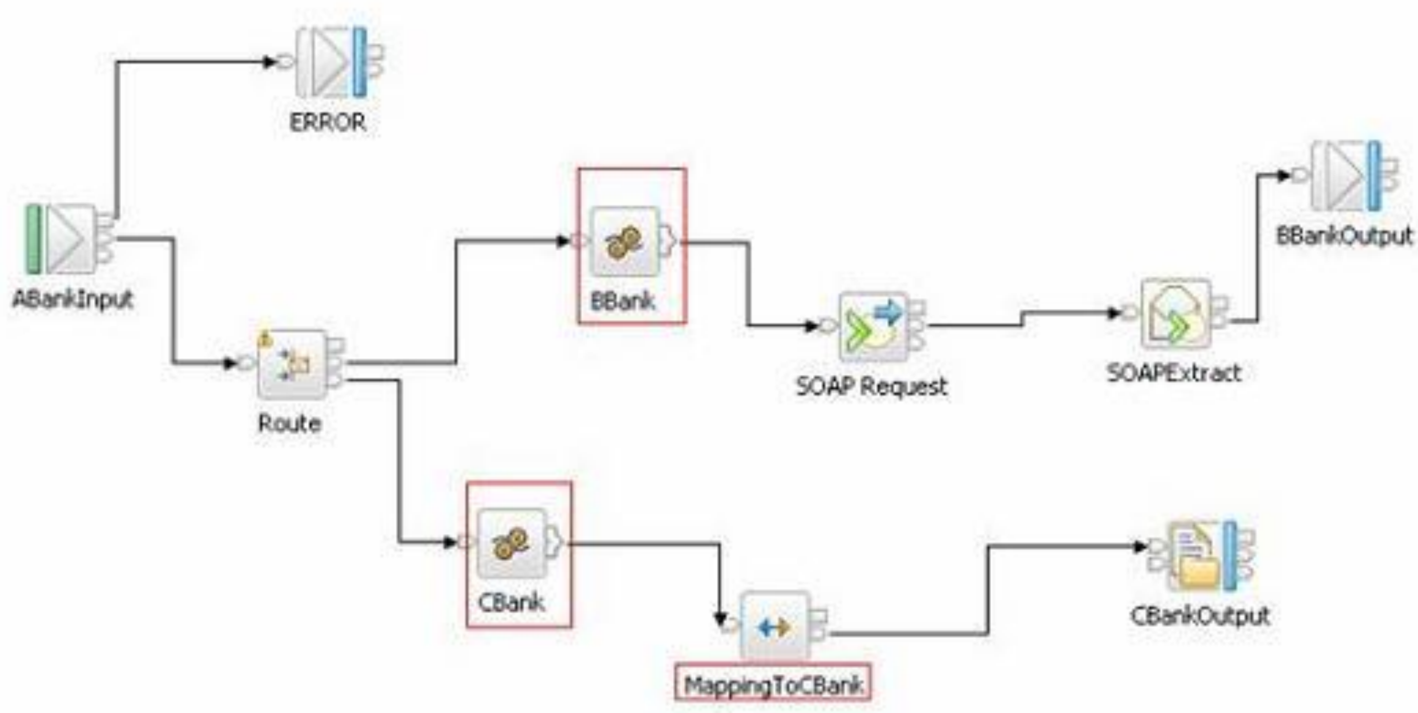
- 场景描述：
  - B银行提供的服务由Web Service方式实现，C银行提供的服务由FTP方式实现，只要把消息放到C银行指定的FTP Server即可,数据格式由C银行指定
  - 对B、C服务性能要求较高，需要每秒钟能同时处理1000到10，000条消息；
  - B银行和C银行都支持通过MQ的方式对其提供的服务进行访问；
  - 利用ESB构建一个统一的查询客户存款服务的，该服务通过查询不同的客户来源，动态路由到不同的服务提供银行。

WMB开发的四个步骤

- 第一步:
  – 将开发好的BBank提供的WSDL导入WMB中，在SOAPRequest节点中可以利用该WSDL文件提供对BBank的访问。CBank提供的是某个FTP服务，WMB中提供的FileOutput节点可以实现对FTP的访问。
- 第二步:
  – 利用WMB提供的Route节点实现对消息的路由，Route节点是WMB6.1的一个新feature，开发起来和WESB中的 Route节点非常类似。之前的WMB版本一般利用Filter节点来实现类似的路由功能。
- 第三步:
  – 提供一个MQ Input节点给A客户，A客户可以通过该MQ节点发送消息，从而访问BBank和CBank提供的服务。
- 第四步:
  – 由于A银行支持对MQ的访问，故B，C银行的返回结果都存放在MQ中，A银行只要访问相应的队列就可以取回结果。

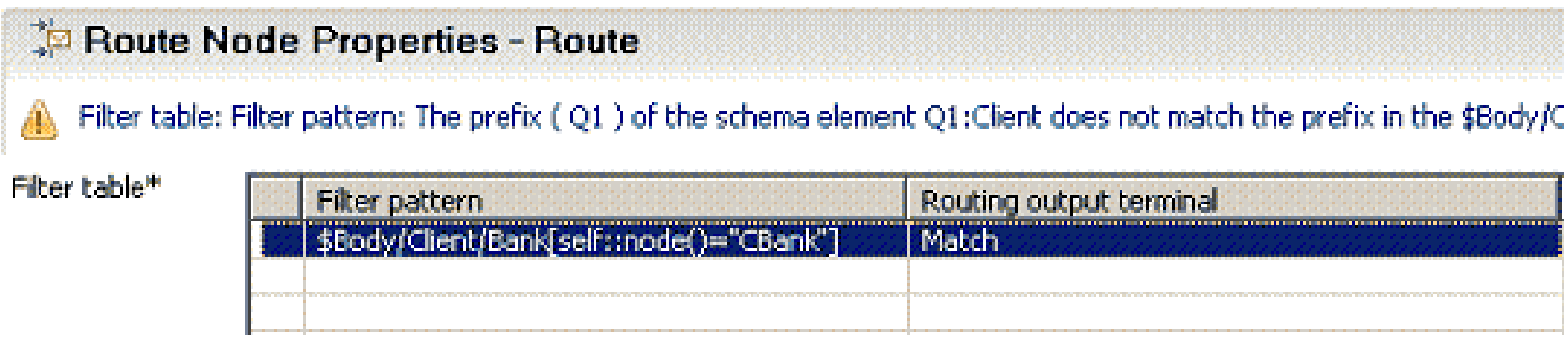
使用WMB开发 Mediation 消息流

- 在BBank Compute节点和MappingToCBank mapping节点中分别采用了ESQL和mapping节点实现消息格式的转换。在BBank中我们也利用了WMB特有的 ESQL实现到SOAP消息的映射。
- 在CBank Compute节点中对存放在FTP中的文件名进行了动态赋值，其文件名字根据消息中唯一的ID信息来标识。



Router规则表

- Route节点的主要信息非常简单，根据消息中的Bank的字段路由到不同的服务。



使用WMB实现业务场景的总结

- WMB提供了更为丰富的内置节点，支持不同协议间的转换，在本例中我们采用FTP作为演示，WMB还支持 JMS、HTTP、TCP/IP 等其他常用协议。
- 由于和MQ天然的内在联系，支持MQ访问的应用系统使用WMB作为ESB将非常自然。
- 和WESB相比，WMB不仅提供了的丰富的消息处理机制，在性能方面也更为优越。

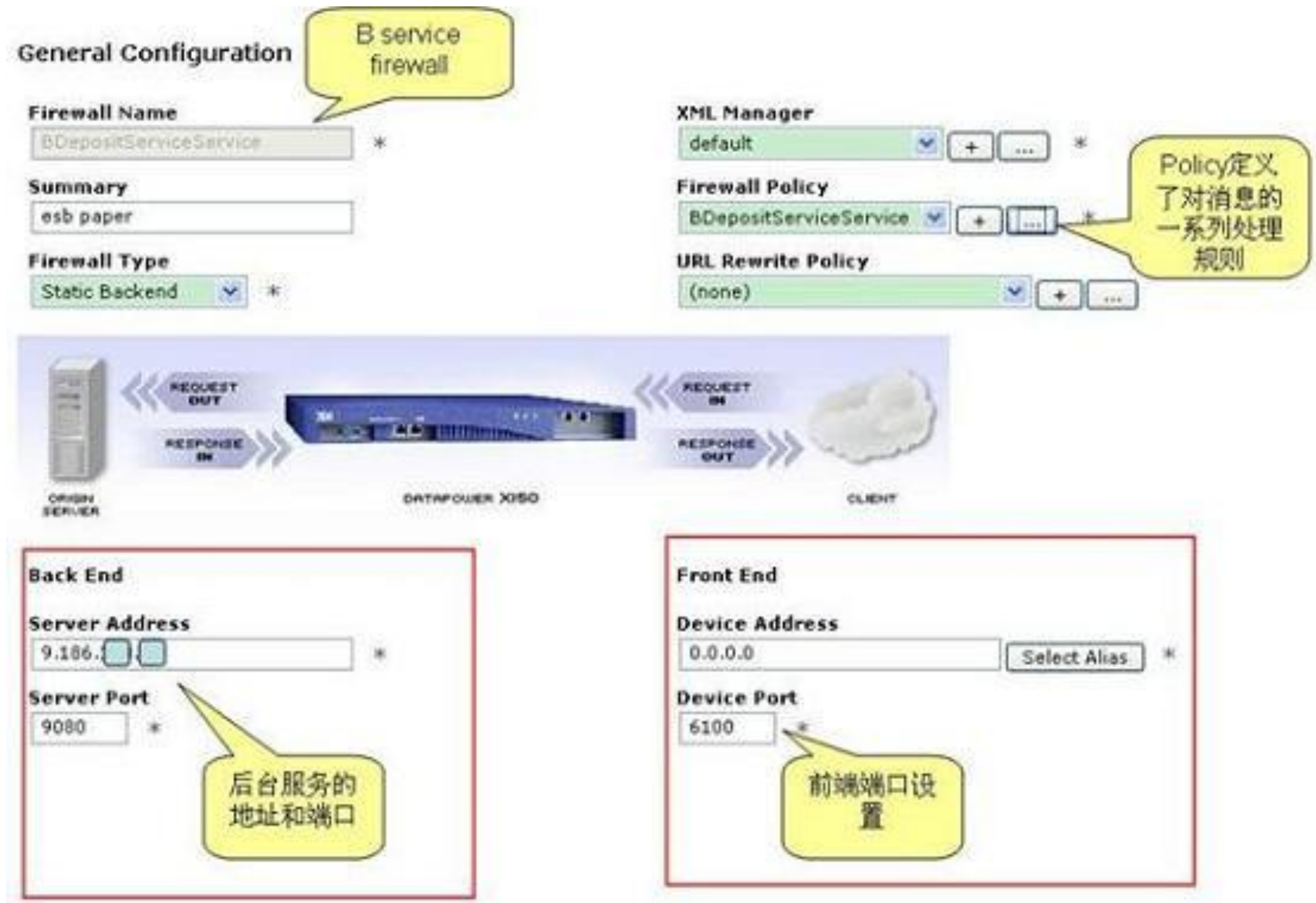
使用DataPower实现业务场景

- 场景描述：
  - 在该场景中，服务的注册，路由等功能和前面描述的 WESB 相似，除此之外，还需要以下安全方面的支持：
    - B,C 提供的服务在服务端已经实现了服务端的安全机制，请求者只有满足相应的机制才能请求服务。
    - 要求服务的请求和返回在安全的传输层（SSL）之上传输。
    - 返回的消息是加密的，需要请求者解密消息。
    - 请求的消息要数字签名，保证消息在请求过程中未被修改。
    - 防范 XML 攻击（XML 攻击的介绍参见参考文献部分）
    - 以上这些安全方面的要求在 WESB 中完全可以实现，但是对安全性的增加会导致：1）开发的复杂度；2）系统性能的大幅下降。

DataPower开发的两个步骤

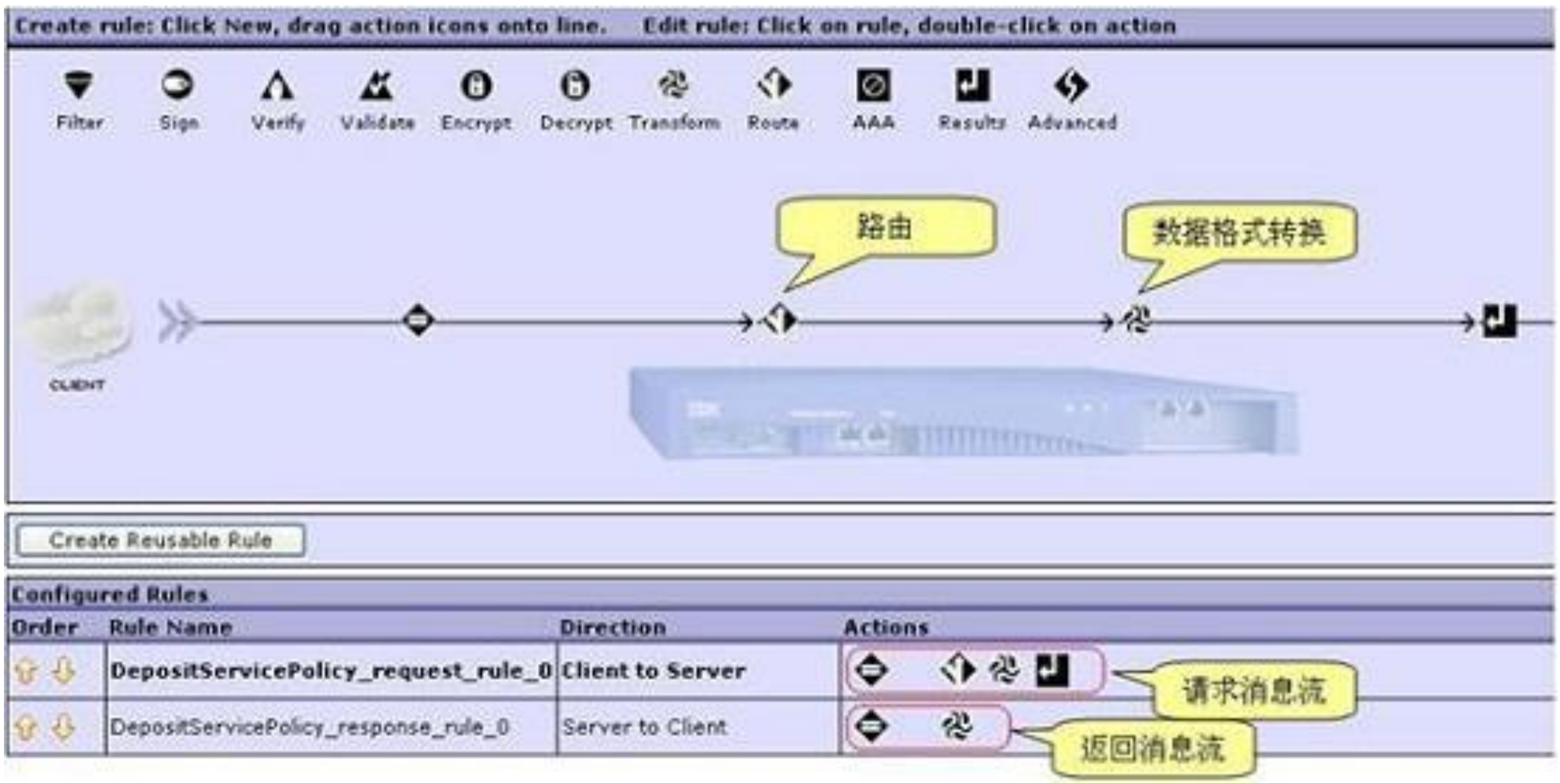
- 第一步:
  - 实现基本的ESB服务注册、路由、消息转换等功能。通过两个XML Firewall来封装B，C银行提供的服务，然后建立一个新的Firewall来实现ESB中的路由和消息转换。
- 第二步:
  - 增加对安全方面的支持

封装B服务的Firewall开发配置界面



ESB路由Firewall

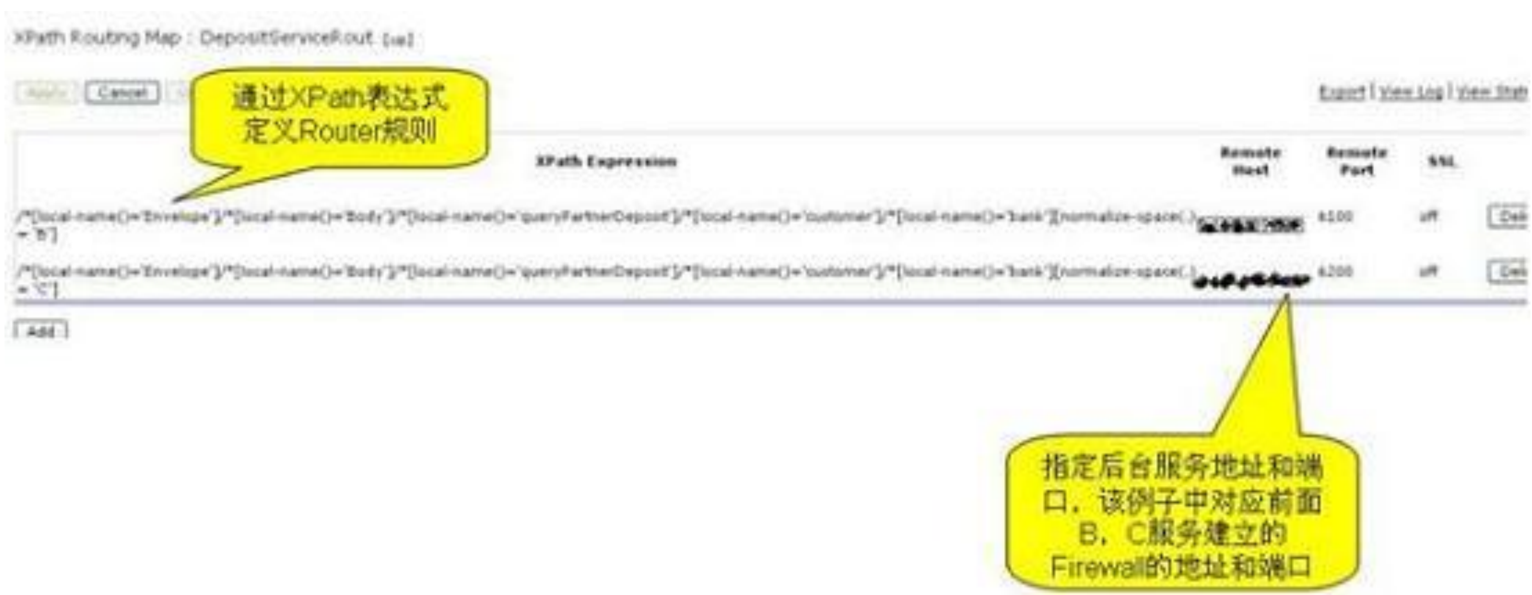
Policy的定义



Create rule: Click New, drag action icons onto line.   Edit rule: Click on rule, double-click on action
Filter
Sign
Verify
Validate
Encrypt
Decrypt
Transform
Route
AAA
Results
Advanced
路由
数据格式转换
CLIENT
Create Reusable Rule
Configured Rules
Order
Rule Name
Direction
Actions
DepositServicePolicy_request_rule_0
Client to Server
请求消息流
DepositServicePolicy_response_rule_0
Server to Client
返回消息流

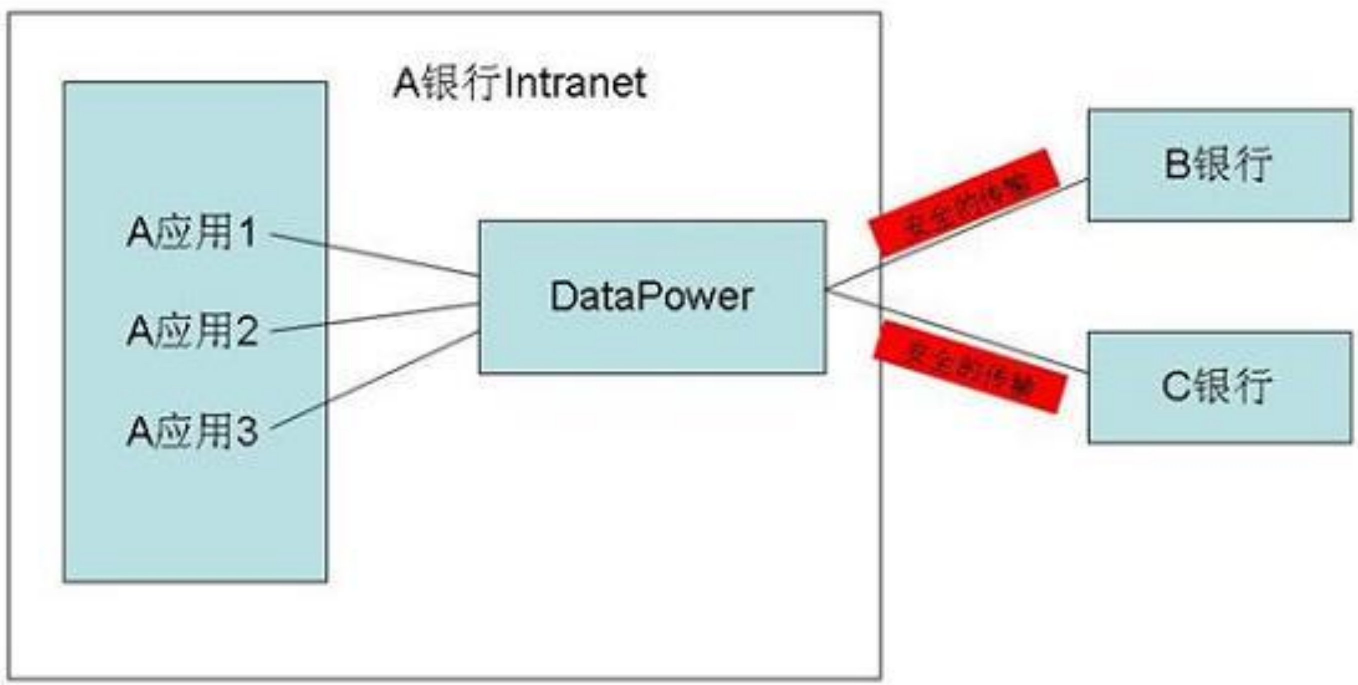
路由规则表

- 配置请求消息流和返回消息流中的各个节点来完成ESB的功能。转换节点中，我们开发XSL来进行格式转换；在路由节点中，我们要定义路由表。



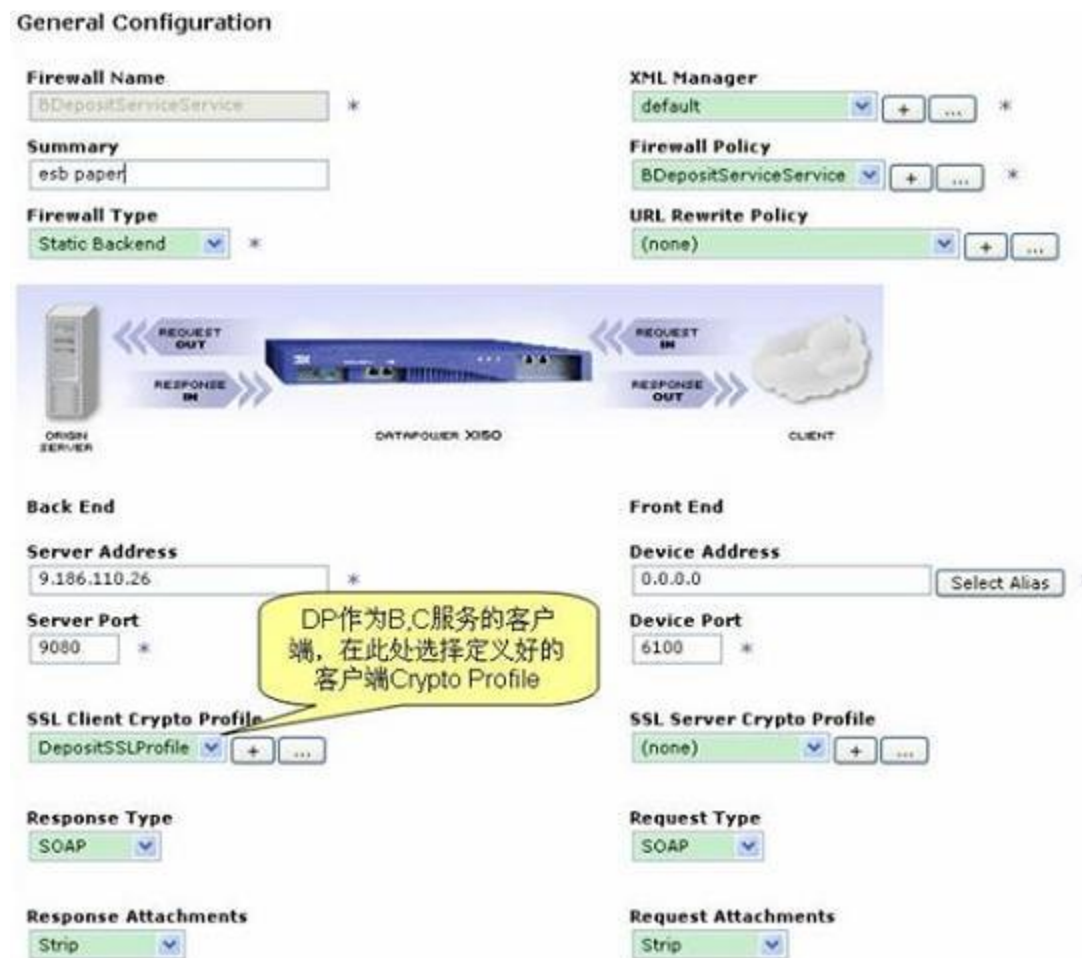
实现场景图中的安全需求

- 假定Datapower用在A银行的内部网中，我们只需要在和B，C银行之间的传输中增加安全支持。



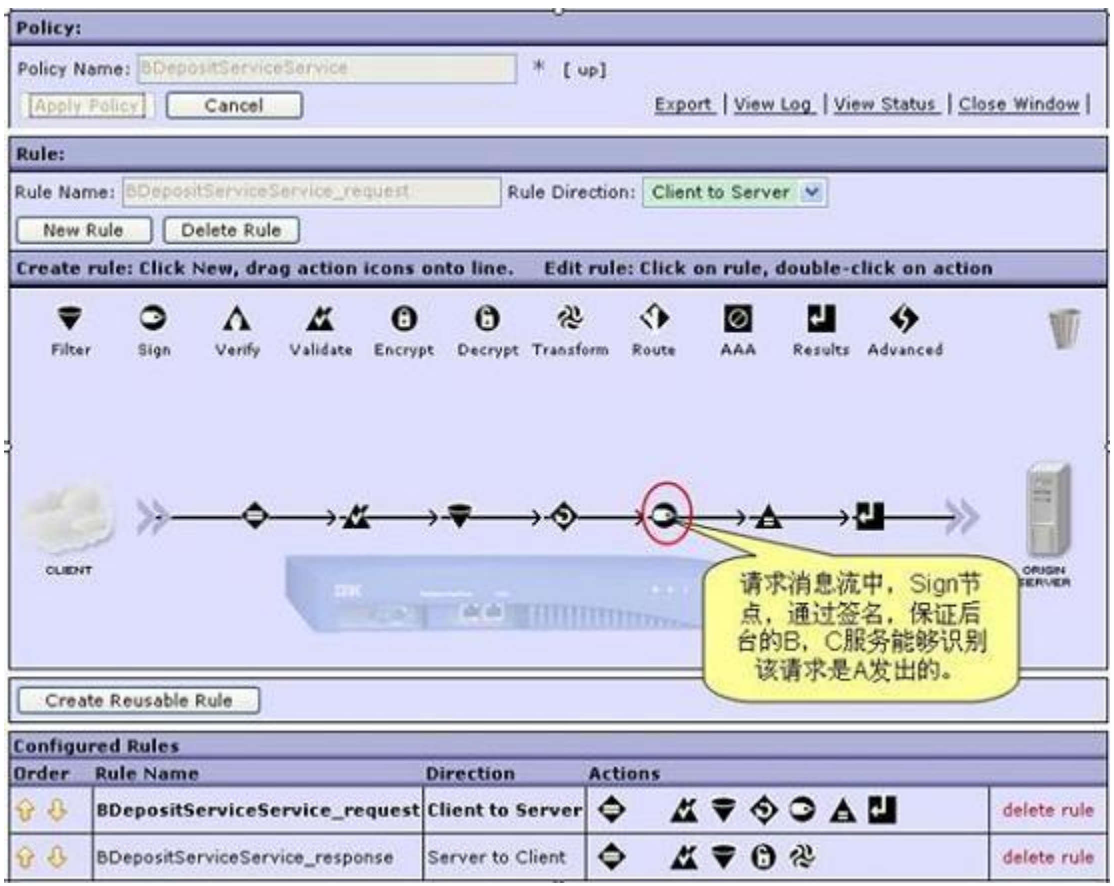
加入SSL

- SSL在Datapower中是一个独立开发的对象，开发好SSL对象后，只需要在XML Firewall的配置界面上选择该对象即可。



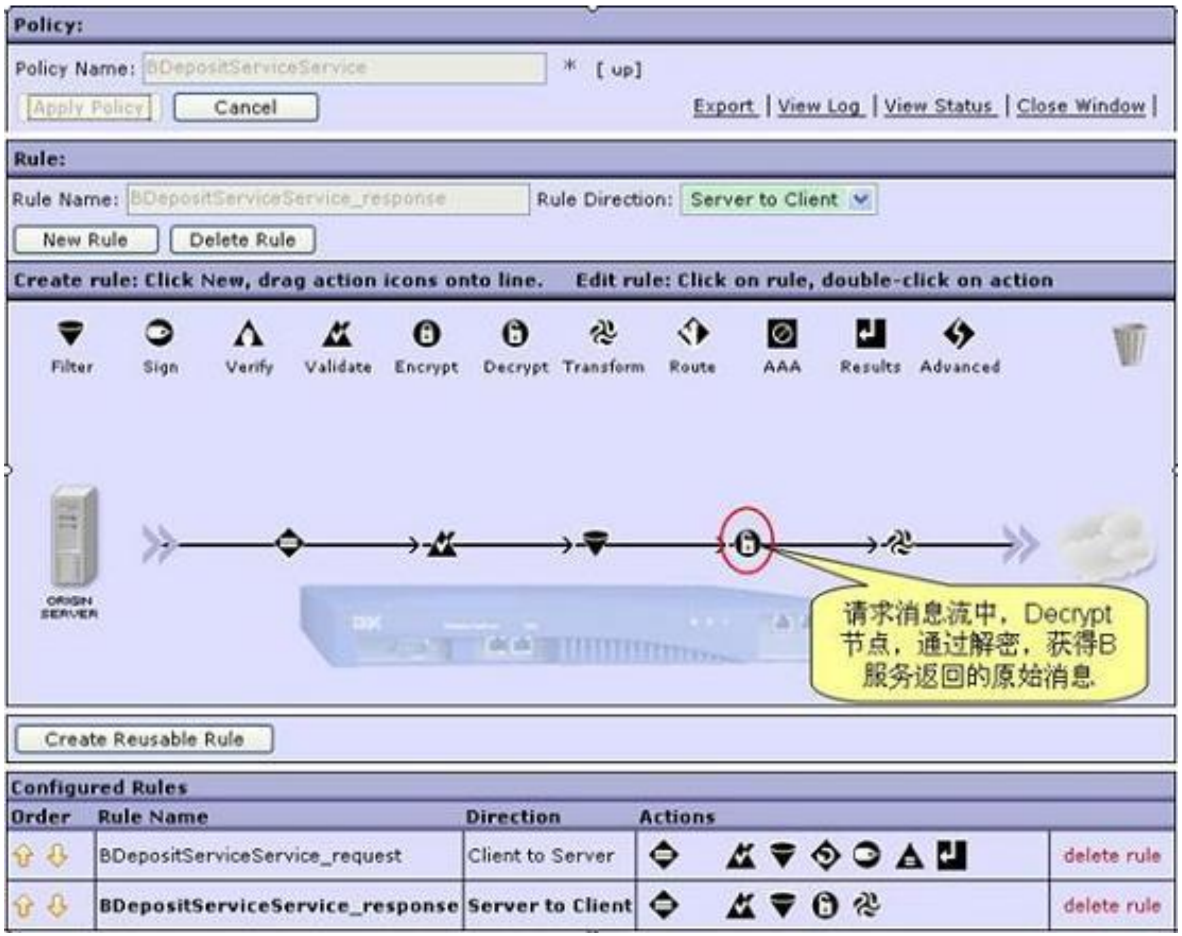
签名和确认

- B银行要求传过去的请求消息带有数字签名，需要在B FirewallService中加入Sign（签名）节点来支持此项功能。



加密解密

- B银行返回消息是加密的，需要在B FirewallService的Firewall Policy中加入 Decrypt（解密）的节点。



使用DataPower实现业务场景的总结

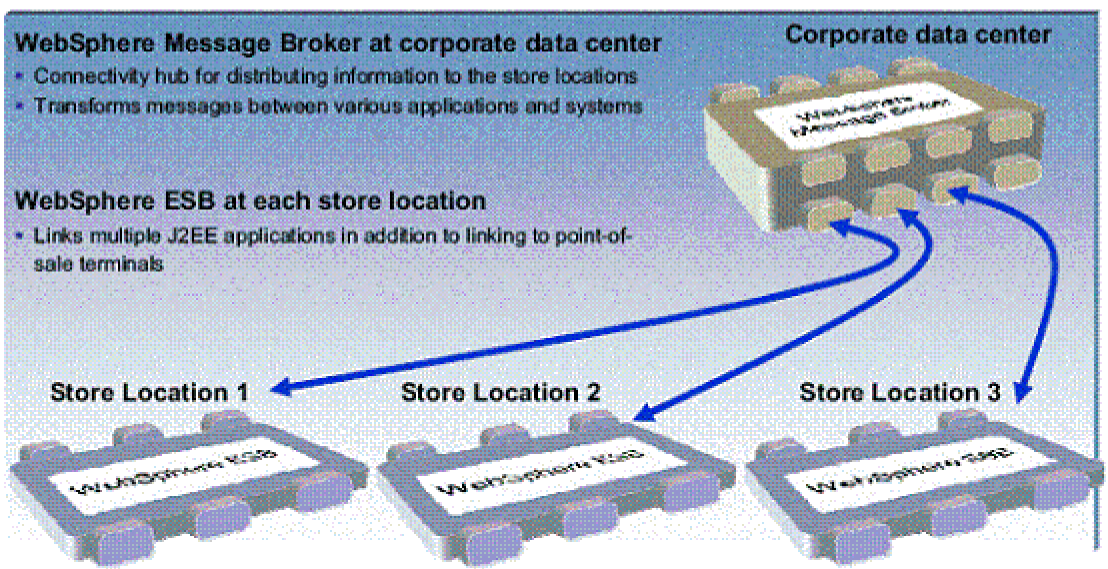
- 在Datapower和B,C银行之间提供安全的服务请求，通过配置的方式即可完成安全的支持，节省了很大的开发成本。
- Datapower对XML的处理速度达到线速。

联合ESB

- 三款产品并不是独立使用的，在某些环境下可能需要三款产品的联合使用。
- 更大型的异构企业通常作为某种自治域的联合体出现，这些自治域根据各个业务部门的职能或治理来定义。在此类环境中，某些服务可以在单个域中进行共享或重用，而其他服务可以在整个企业中进行共享或重用。在这些情况下，我们建议采用某种形式的ESB联合，该形式的ESB联合与域联合的需要相匹配。ESB联合允许在不同的域中使用不同的ESB产品，并支持域需求与产品功能之间的最佳匹配。

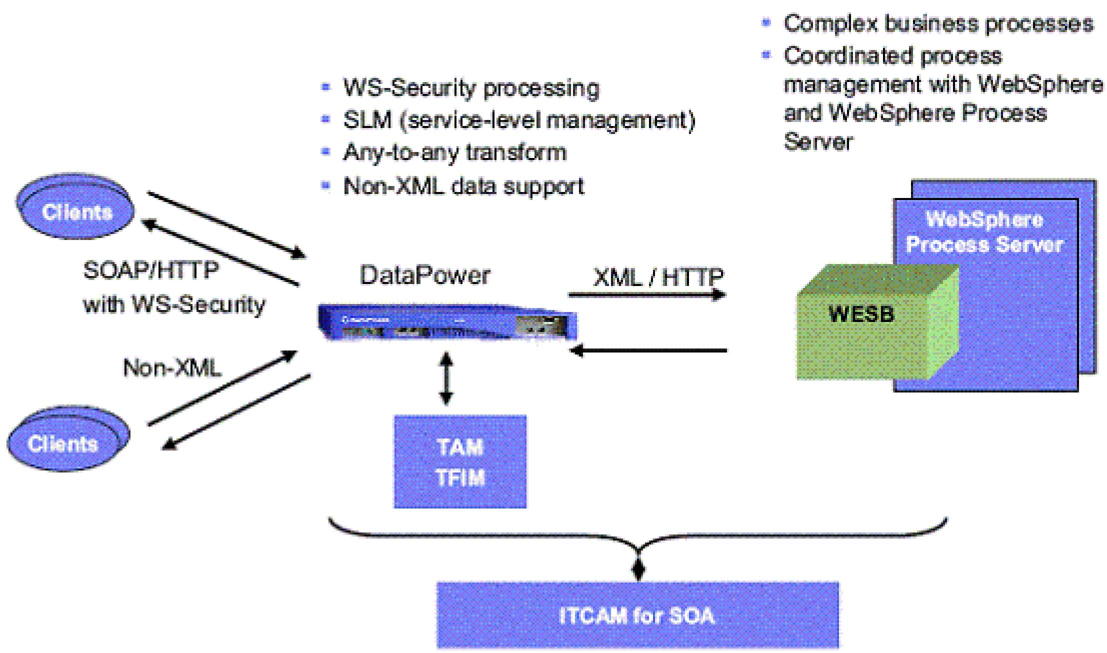
WESB和WMB联合使用的场景

- 某跨国公司在世界各地都有分公司，由于时区的原因，每天的信息需要通过异步的方式统一到总公司，由于总公司的业务量大，我们可以使用WMB做总公司的 ESB，而在分支机构，业务量小，且都是 J2EE 和 Web Service 的应用，我们可以使用 WESB 作为分公司的 ESB。



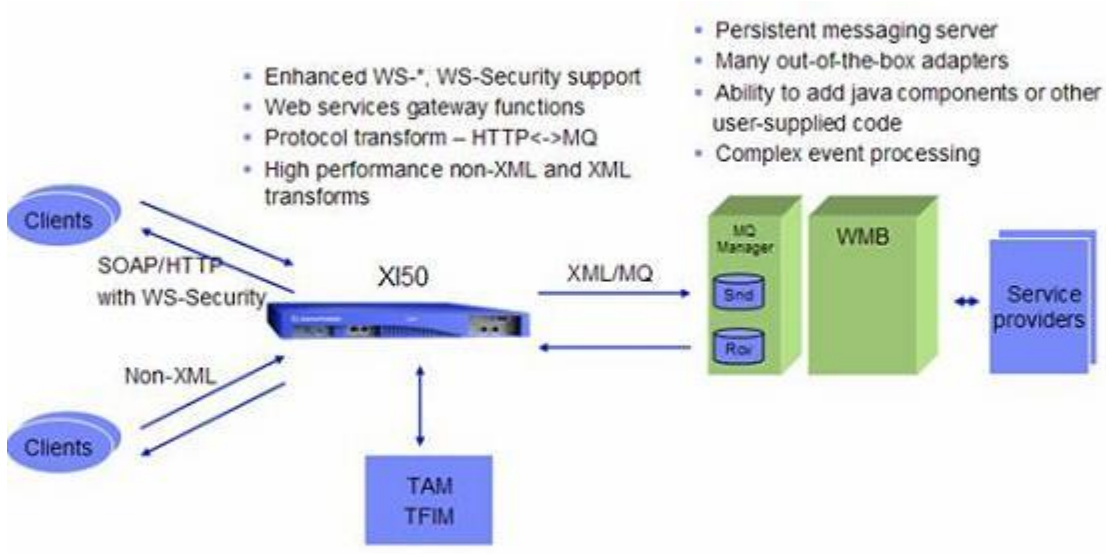
Datapower和WESB联合使用的场景

- 可以在WPS上实现负责的业务流程，在企业内部使用WESB作为ESB。在该场景中WESB只负责服务的注册、路由和查找功能，而安全方面的处理，以及部分消息格式转换的功能由Datapower处理（外部传来的非XML格式的数据通过Datapower处理成XML格式的数据），该场景既利用到Datapower在安全和XML处理方面的优势，有利用到WESB和WPS集成的优势。



Datapower 和WMB联合使用的场景

- WMB提供了多种多样的消息协议和格式的支持，比如遗留的EIS系统，SAP、PeopleSoft等，也可以充分利用WMB的扩展性自定义消息集或者消息节点，以此来满足特殊的应用需求。Datapower可以提供高性能的Web Service安全网关。客户端通过SOAP over HTTP可以访问到Datapower，而Datapower用MQ访问WMB。该场景兼顾了Datapower在性能上强大的优势和WMB丰富的消息协议与格式支持。



IBM的与ESB相关的其他产品

- WebSphere MQ为多种不同平台提供可靠消息。很多公司都使用WebSphere MQ作为消息传递中枢。
- WebSphere Service Registry and Repository提供了集成的服务元数据存储库来治理服务和管理服务生命周期。它可促进服务可见性、一致性，减少SOA中的服务冗余。
- WebSphere Transformation Extender提供了通用转换功能，非常适合满足极为复杂的需求或快速变化的需求，如EDI等。
- WebSphere Adapters是外接程序，可帮助为打包应用程序或其他遗留资产启用服务，以便参与到SOA中来。所提供的适配器允许将各种遗留信息系统和技术包含到ESB中来。

IBM的与ESB相关的其他产品

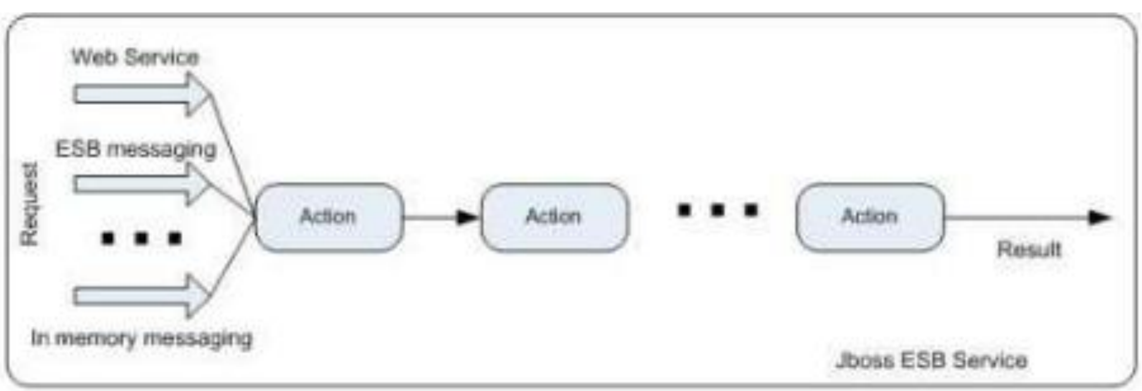
- WebSphere Process Server提供流程工作流功能，并包括WebSphere ESB。有些情况需要对来自很多个系统的服务调用和响应进行协调，而此解决方案提供了在此情况下编排复杂ESB交互的功能。
- IBM Tivoli Composite Application Management for SOA 是专门针对SOA环境的Tivoli监视产品。通过IBM Tivoli Composite Application Manager for SOA，可充分利用现有系统管理工具，提供SOA环境的全面操作视图
- WebSphere Business Services Fabric是用于组合业务服务的建模、组装、部署、管理和治理的端到端SOA平台。WebSphere Business Services Fabric可帮助创建业务级的服务，以供组装为扩展的跨企业业务流程和解决方案，而且可以基于服务请求的业务上下文对这些流程和解决方案进行动态个性化和交付。

JBoss ESB

- JBoss ESB是JBoss推出的ESB实现,也是JBoss的SOA产品的基础。
- Jboss ESB基于RosettaNet ESB，支持服务的创建、部署和整合。这些ESB服务可以通过多种传输暴露出来。所有的ESB服务都有一个方法（doWork），可以通过下面的接口（由所有的服务共享）描述：在Jboss ESB中，ESB消息和SOAP消息类似，包括标头（header），消息体（body），错误（fault），附件（attachments），等等。每个部分包括一个可序列化的java对象集合（map），通过集合中定义的name进行访问。

JBoss ESB

- JBOSS ESB建立在三个核心的体系结构组件上:
  - (1)消息监听器和消息过滤器代码。消息监听器监听消息并路由，然后引导消息到到管道。消息过滤器则过滤消息，并将消息路由到另一个消息端点。
  - (2)一个基于路由服务的目录。
  - (3)一个消息存储库，存储在ESB上交换的消息/事件。

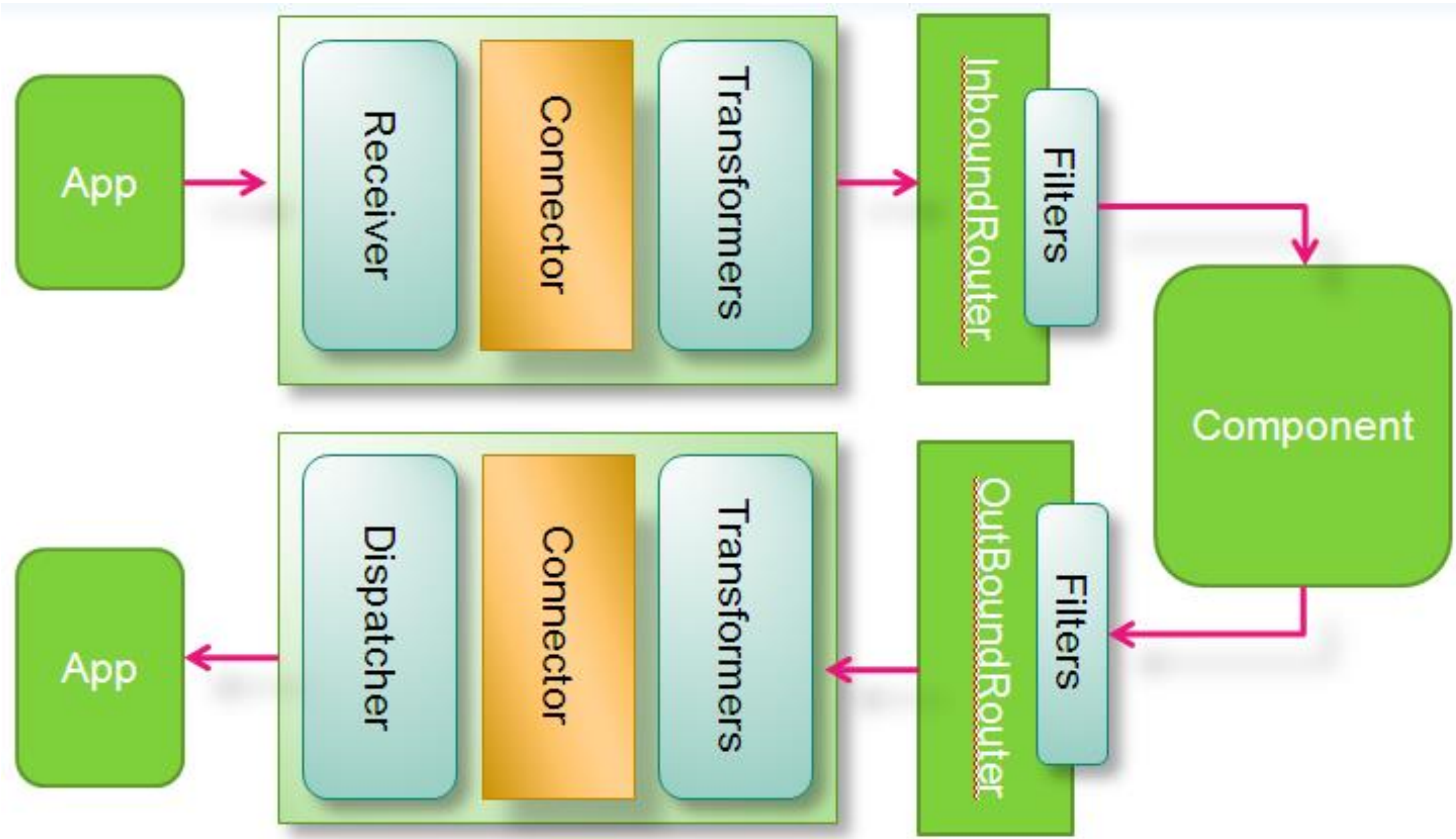


JBoss ESB Service

JBoss ESB的优点

- JBossESB在功能上是较为强大，较之其它服务总线，它的技术架构方案是最独立的。
- 它除了支持 J2EE标准、Web Service外，还支持多种的远程调用协议，例如JMS。
- JBoss ESB是完全开放源码的免费软件， 具有丰富完备的文档资料，在开发和扩展上减少阻力。且依托于成熟的JBoss社区，周围齐全的开源项目支持，为后期的平台扩展提供了丰富的选择空间。

Mule的架构图


App
Receiver
Connector
Transformers
InboundRouter
Filters
Component
App
Dispatcher
Connector
Transformers
OutBoundRouter
Filters

Mule的发展趋势

- 社区活跃度
  - 活跃程度最高，用户量大，不断推出新版本。
- 易用性
  - “让一切变得更简单”是Mule的宗旨。2次重构核心架构、推出接入云应用，消息流，基于模式的配置以及热部署；Mule IDE3.0，将支持图元拖拽，简化开发。
- 扩展性
  - 增加一个新协议非常简单，只需实现5个接口类即可。
- 管理性
  - 用Mule Management Console管理、部署和监控应用
- 文档
  - 文档非常丰富，降低了使用门槛。

Mule的优点

- 基于模式的配置

Pattern Name	Usage
Simple Service	Exposes JAX-WS annotated components as SOAP web services. Exposes JAX-RS annotated beans as RESTful components. Can also handle JAXB, XML and raw content with simple POJO components.
Web Service Proxy	Proxies remote web services. Can perform transformations on the SOAP envelope. Can rewrite or redirect remote WSDLs to local ones.
Bridge	Establishes a direct conduit between an inbound endpoint and an outbound endpoint. Supports request-response and one-way bridging. Can perform transformations. Supports transactional bridging of inbound to outbound.
Validator	Validates inbound messages against a defined acceptance filter. Returns an ACK or NACK response synchronously and dispatches valid messages asynchronously.

  - 基于web service proxy模式的web service的穿透场景的配置（3个属性）
    - <ws:proxy name="muleWsProxy" inboundAddress="http://localhost:8080" outboundAddress="http://webservice.webxml.com.cn/WeatherWS.asmx"/>

Mule的优点

- 易扩展
  - 新增一个协议/transport只需实现5个接口类
    - org.mule.api.transport.Connector
    - org.mule.api.transport.MessageReceiver
    - org.mule.api.transport.MessageDispatcher
    - org.mule.api.transport.MessageDispatcherFactory
    - org.mule.api.transport.MuleMessageFactory

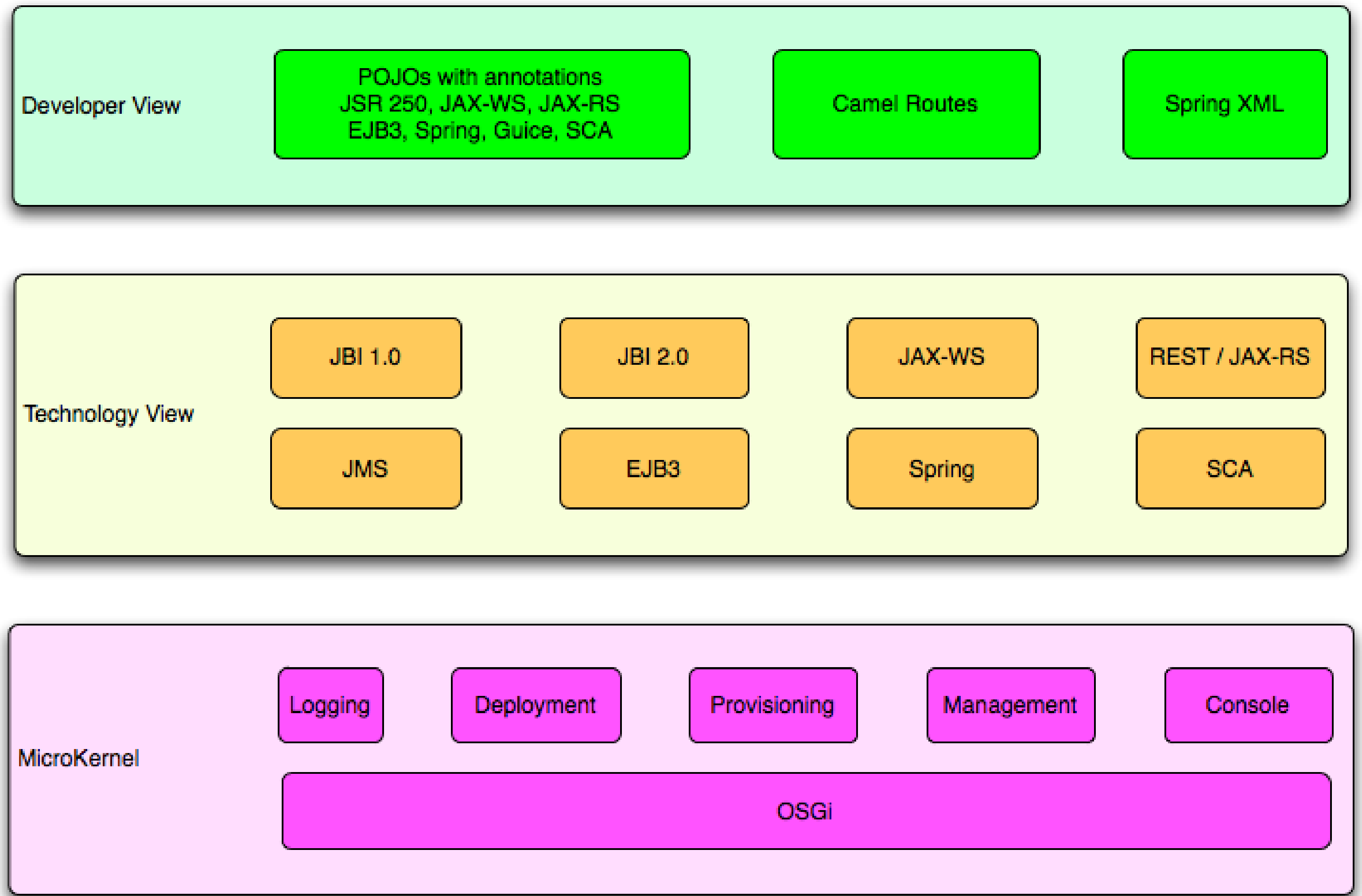
Mule的优点

- 异常处理框架
  - 异常策略设置级别:model和service
  - 异常处理方式:
    - 1. 将异常路由到指定的目的地
    - 2. 根据异常类型过滤异常，并路由到指定目的地
    - 3. 设置重试次数
    - 4. 当采用了事务时，可以在异常处理策略中设置当发生异常时是继续提交还是回滚事务。

Mule的缺点

- 集群功能弱
  - 1.只能配置一个主实例和一个从实例
  - 2.不支持flow和基于模式的配置
  - 3.某些路由会丢失或者获得重复的消息
- Mule IDE
  - 目前的IDE只提供XML级别的编辑，还不能实现图元的拖拽
- 稳定性
  - 需要在测试场景下进行验证

ServiceMix的架构图



ServiceMix的发展趋势

- JBI2.0规范发展缓慢
  - IT巨头Oracle,IBM投了反对票，目前只有几家小公司投支持票
- ServiceMix迁移到OSGi
  - JBI2.0中增加了对OSGi的支持；
  - ServiceMix4.x完全基于OSGi，
  - ServiceMix3.x继续前行
- 孵化新项目
  - Camel
  - Karaf

ServiceMix的优势

- 无缝集成CXF, ActiveMQ, Camel和ODE
  - ServiceMix, ActiveMQ, CXF, Camel都是FUSE的开源产品
- JBI的优势
  - 组件BC, SE可以在任何JBI容器（不限于ServiceMix）中直接运行，复用性强
- 基于OSGi
  - 具备OSGi的优势：模块化，热部署，易扩展
- 基于Karaf
  - 提供了非常丰富的命令，管理、部署和监控ServiceMix

ServiceMix的缺点

- JBI规范太复杂
  - 已被主流中间件厂商抛弃,没有受到业界的青睐
- 架构复杂
  - 由于JBI的复杂性所致，其架构并非轻量级
- 缺少IDE的支持
  - 必须手写大量的XML配置文件
- 缺少governor的支持
  - ServiceMix4只是借助Flex的web console管理OSGi的bundle
- 学习门槛高
  - 用户文档和相关资料比较少