Introduction to The Versatile B2B Gateway

(An SOA Design Pattern)

(DRAFT V0.3)

Hamid BenMalek, Jacques Durand (Fujitsu Software)
with contributions from:
Pete Wenzel, Monica Martin (Sun)
Matt McKenzie (Adobe)

Introduction

B2B exchanges as supported between pairs of business partners will require diverse modes of connectivity as well as different forms of back-end integration.

- Some messages will invoke Web services that are deployed behind the firewall, some will not.
- Some deployments will be strongly restricted in terms of access and security, others will not.
- Some endpoints will be up 24/7, some will not.

A gateway solution that recognizes the need for different styles of connectivity, as well as of back-end integrations, is desirable. Such a solution must also build on widely accepted protocols (SOAP, HTTP), in order to be adopted by a broad spectrum of users.

The solution introduced here under the name of "Versatile B2B Gateway" is designed to address these requirements. It can be seen as a way to extend a service-oriented architecture with a B2B connectivity solution.

B2B Integration: Supporting Various Patterns of Message Consumption

B2B exchanges can be of very diverse nature and supporting this diversity is likely to require a combination of technology packages that otherwise would be self-sufficient for some type of exchange. Some B2B exchanges are akin to service invocations, and some are not.

When a Message is a Service Invocation:

The notion of external message as a service invocation is appropriate in the following cases:

- The immediate consumer of the message is a status inquiry service that is able to query back-end systems running behind the firewall, while managing an interactive user session.
- The message is intended to leverage a computational resource or service remotely available on a public or private Web site. It is often generated automatically by an application or business process, or another Web service, as a way to delegate some part of the processing.
- The message carries form data (subscription, status info, etc) from a user, and the processing service is expected to provide a quick feedback to the user in terms of whether this form is valid, accepted or invalid.

These use cases are well supported by Web services stacks, though a B2B gateway may provide additional support on the connectivity side, as described later.

When a Message is *Not* a Service Invocation:

Messages may carry business documents that exist and persist independently from the way these are processed. The processing of such messages often follows an "event" model, instead of a service invocation model. This means the processing of the enveloped business document is not pre-determined. Some pre-processing may be required, before deciding which service they need to be submitted to – if they ever are intended to a service. Such cases include the following:

- The immediate message consumer - after message reception - is a business process instance (as supported by BPM engines) that needs this document at some point in its life cycle. The structure of accepted documents may vary: the validation of these may be controlled by the process. A publish-subscribe model is more appropriate here, combined with filtering that may involve both message header and business payload. This workflow process may later decide if a service must be invoked, and which one .
- The messages intended to an application component that does not require real-time processing, are batched for processing at a more convenient time of the day. The immediate consumer of such messages will not be an application service, but again a subscriber that may perform filtering based on both structure and semantic content before adding to the batch intended for this application.
- The immediate message consumer is a dispatching component that forwards internally the received documents based on some content analysis (header or payload).

Although the immediate message consumer in above cases could be wrapped as a service (defined in WSDL), there is little or no value in doing so. The above use cases typically involve a B2B gateway and / or an integration broker. The actual service invocation – with application semantics – would only occur later in the process and will originate locally, not on the partner initiative.

Manageability Considerations:

The description of these business documents (schemas) in WSDL is not without impact on manageability. In some large supply-chains, partners must frequently handle hundreds of different types of documents - and not uncommonly thousands. This is because customization of document types is the rule and not the exception even with strongly coercive standards such as RosettaNet.

As mentioned before, these business documents may or may not be directly associated with service invocation in a predefined way. If they are, their type definitions (e.g. XML schemas) will typically be referred to in WSDL files. Because the pool of document types that are to be handled is often evolving (additions, removals, versioning, customization), the set of related WSDL files is intrinsically unstable. It is imperative to reduce the impact of this instability on business partners, and also to make it more manageable, because these changes in service definitions usually cascade down to new code generation and client proxy generation.

By introducing a gateway component that shields partners from these upgrades, messages do not have to be turned down because the contained business document does not comply with the latest service upgrade. Instead, data transformation, document mappings, alternate routing – all functions ultimately supported by an ESB, but also by a more conventional integration broker – may take place after message reception and before the destination service is invoked. This allows for smooth, controlled transitions and upgrades among many non-synchronized business partners.

Heterogeneous B2B Exchanges:

As previously explained, B2B exchanges may have quite different profiles in terms of their nature, the stability of their content, the context in which they operate, the back-end connection they use, the way messages are processed. This heterogeneity in the message handling will persist even when adopting a single standard for messages or documents.

As a consequence, the service invocation model, as currently implemented in WS stacks, has value for some exchanges and less for others. When it has value, its scalability and manageability still depends on the appropriate architecture.

Supporting Several Modes of Connectivity

The term "connectivity" denotes here the type and style of a B2B connection between two partners. We have seen previously that some exchanges will be service-centered, others will be document-centered (in the sense that an RPC-style messaging is not the most appropriate for processing these documents). In addition, different connectivity restrictions may require that some business partners only pull messages they receive, while others can receive messages pushed to them via requests. Such modes of message transfer should remain orthogonal to the service definition. The notion of WSDL binding may capture only some aspects of these connectivity patterns, but in a way that ties these to the service definition.

An enterprise gateway must also accommodate various business partners, in particular SMBs. Depending on their profile, trading partners may experience intermittent connectivity, restricted addressing due to the use of ISPs and/or firewalls, different processing flow capabilities, reduced hand-shake capability and no out-of-band visibility. These requirements are addressed by ebMS V3.

Outline of the Versatile B2B Gateway

The versatile B2B gateway has the following general features:

- It is able to directly bind to various message consumption models behind the firewall: queuing, service, application callback, enterprise bus.
- It leverages protocol-level Web services standards (WS-Security, WS-Addressing, WS-Reliability / WS-ReliableMessaging) and is compatible with key WS-I profiles such as SSP 1.0, AP 1.0) so that compliant WS stacks and SOAP processors can be used.
- On the external side, it leverages messaging features from the ebMS V3 standardization work in progress, in order to address the diversity of connectivity styles that is found between business partners, and also within a single enterprise.

Figure 1 illustrates the various ways the B2B gateway may connect to enterprise middleware and applications.

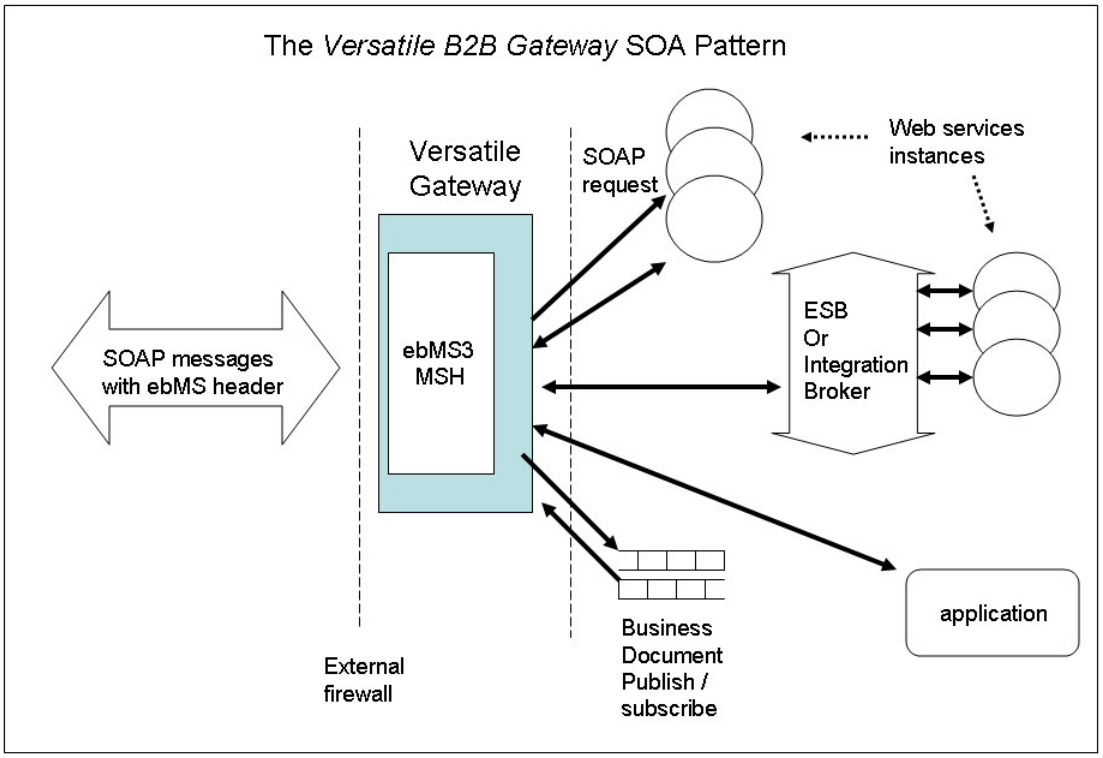

Figure 1. Overview of Integration Options in the Versatile B2B Gateway.

The gateway may interface directly with Web services as defined by WSDL, or with other integration middleware, or also directly with a back-end piece of application. In order to achieve this, it leverages addressing capabilities as provided by WS-Addressing. The forwarding of a Web service request is decided upon analysis of the wsa:To header, which must be present when the message destination is a WS endpoint known prior to invocation. The forwarding of other SOAP messages (no having the wsa:To header) is decided upon content of the ebMS header (eb:CollaborationInfo and eb:MessageProperties).

The gateway performs the usual proxy function: the sender of the message only uses the gateway URL. The actual physical address of a Web service deployed behind the firewall does not have to be known from the external client. The service may be redeployed on another internal server without affecting the client code: only the routing function in the gateway is affected.

Figure 2 illustrates the use case where the B2B gateway must allow business partners with diverse connectivity capabilities, to cooperate. A roaming endpoint with no fixed IP address is expecting an asynchronous response from a Web service previously invoked

by another partner. Not being able to receive incoming requests, it will pull the response from the gateway. The Web service on the back-end is not aware of the way this response will reach its destination: it has issued the response as a separate protocol request to the gateway, which in turn may push this response or get it pulled by the destination party. The proper mode of transfer is decided by the gateway, based on the content of wsa headers (wsa:To, wsa:ReplyTo, wsa:RelatesTo) and on their association with a preconfigured transfer mode for this destination.

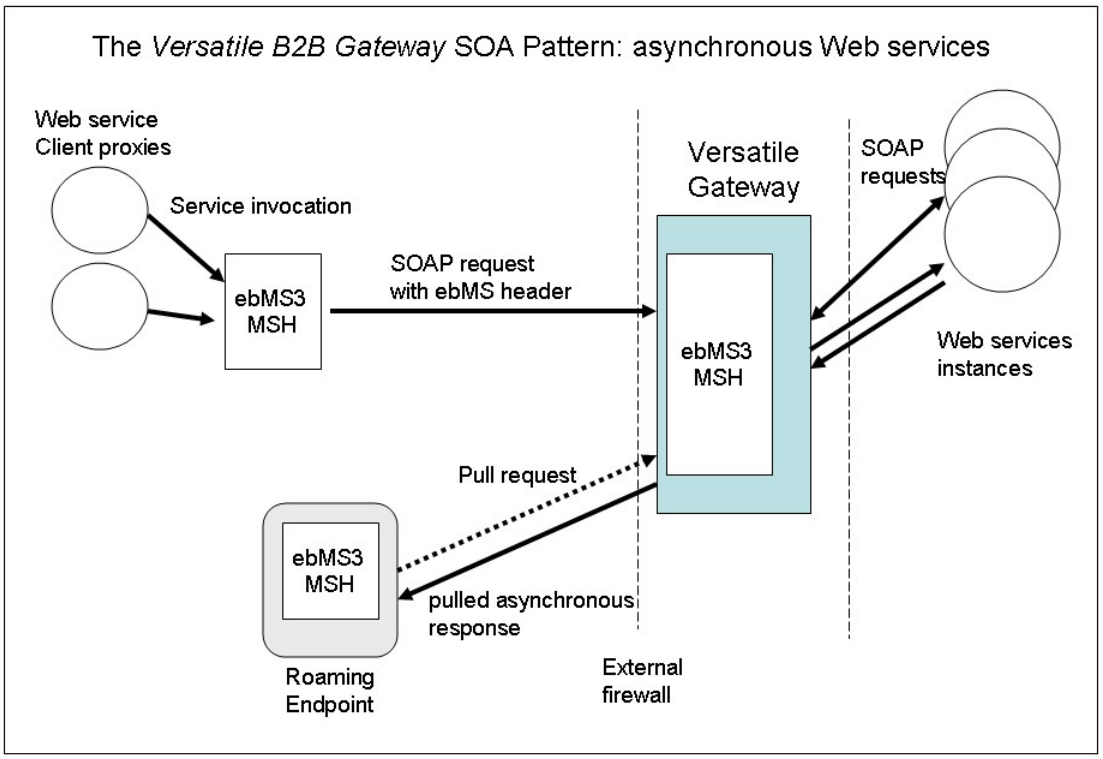

Figure 2. An Asynchronous Web Service Use Case.

Figure 3 below illustrates the use case where the same B2B gateway must allow partners to exchange business documents of diverse types, using a publish-subscribe model to integrate with back-end applications or services. The documents may be XML or not. When they are of XML format, they may comply with a large number of different schemas or DTDs. The posting to the back-end queue is decided based on ebMS header information (eb:CollaborationInfo and eb:MessageProperties). In turn, business documents made available to the gateway may be accessed from outside, for example using a pulling mode of transfer.

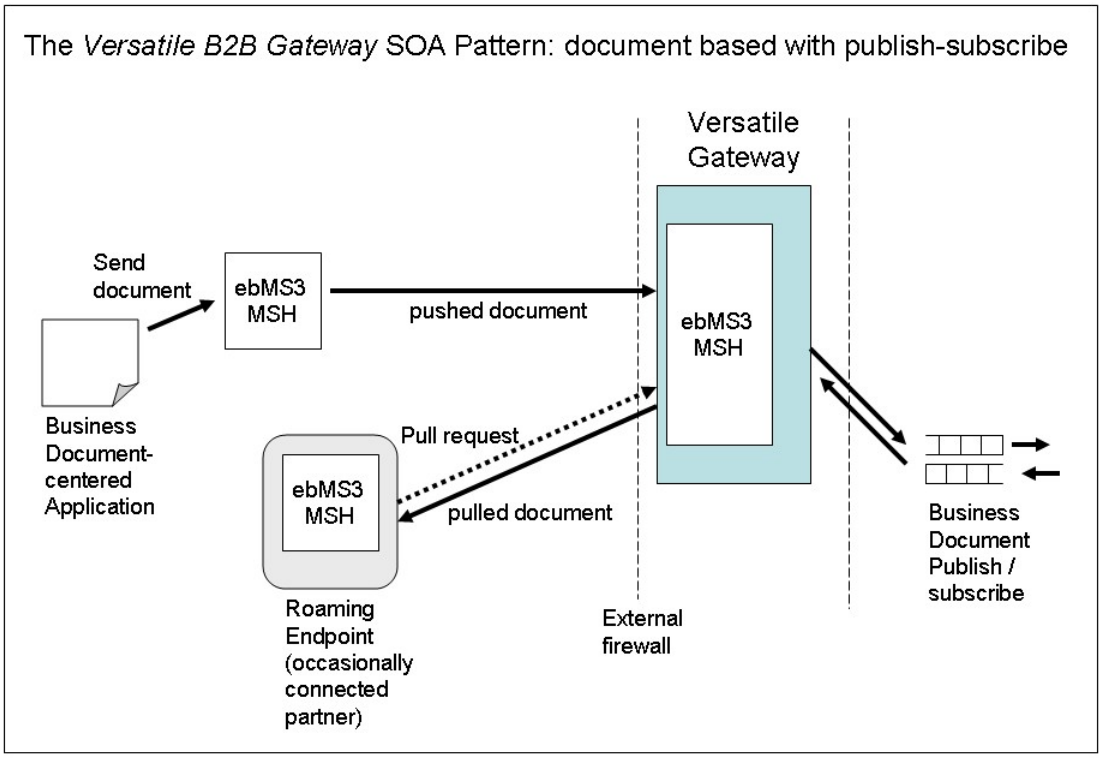

Figure 3. Posting and Fetching Business Documents.

Figure 4 below illustrates the use case where some message transfer mode may be possible with a business partner (here hosting Web service A) but not with another (here hosting Web service B), due to different security constraints. In this example, the partner hosting service B will not allow for external incoming requests, or will allow it only from a very limited number of partners. In such a case, it is desirable to keep the client code and also the Web service definition independent from such constraints. The security environment may vary over time for A and for B and the client application should not be affected, remaining unaware of whether its requests are pushed or pulled toward the destination service.

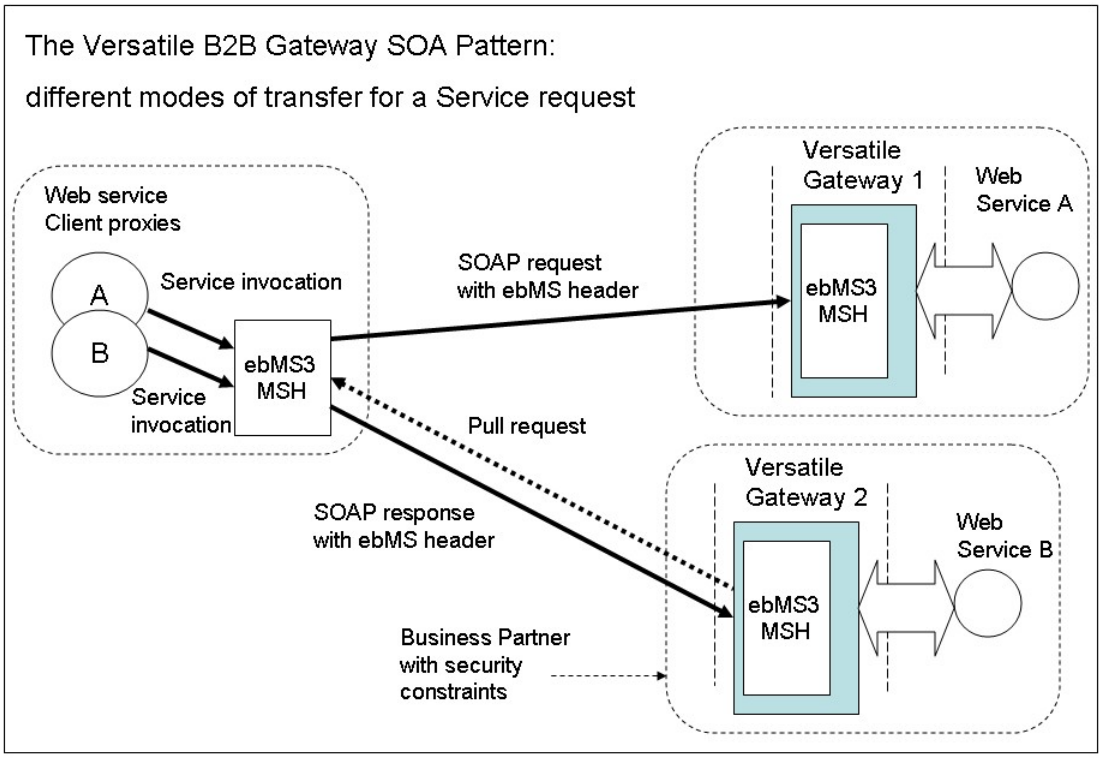

Figure 4. Pushing or Pulling a Service Request.

The mode of tansfer above is decided based on an agreement shared between the two ebMS endpoints. This agreement may vary over time without affecting the service and client code.

As a summary, the versatile B2B gateway introduced here from a user perspective is an SOA design pattern characterized as follows:

- On the external side, it supports messaging functions such as those found in ebMS V3 which address connectivity constraints previously mentioned, along with built-in support for monitoring the compliance of exchanges to business transactions.
- On the internal side, it allows for several integration solutions:
    - The Web service intermediary model where the receiving (ebMS) message handler, as a common endpoint and SOAP intermediary, is internally dispatching messages after stripping them from ebMS-related message headers to one or more WS deployed behind the firewall,
    - The broker feeder model, where the immediate message consumer is an ESB or integration broker, that will mediate the Web service invocation.

    - The message queue model, where the receiving message handler is posting the received documents to a queue for internal consumption, or subscribing to a queue for messages to be sent out.
    - Direct ad-hoc communication with an application, e.g. using a callback "onMessage" type of function.
- It supports message forwarding depending on the back-end consumption mode based on WS-Addressing headers (wsa:To, wsa:Action, wsa:RelatesTo) as well as on ebMS headers.

A more detailed specification of the versatile B2B gateway will the the object of a separate document.

The Versatile B2B Gateway as an SOA Design Pattern

The term "SOA Design Pattern" denotes here a design pattern that addresses a specific SOA function in a way that leverages existing Web services and e-Business standards. Such patterns only pretend at providing one solution among several alternatives.

An SOA Design Pattern is not a specification profile, although it may include some elements of profiling for underlying specifications. Instead, an SOA Design Pattern refers to supporting specification profiles when these exist, and may be compatible with several of these profiles.

The "Versatile B2B Gateway" is an SOA design pattern for B2B integration. It supports a well-established principle in SOA: loosely coupled interactions, where the messaging function is carried out independently from the service interface logic, and is not narrowly associated with a service invocation even if ultimately its processing will result in some service(s) invocation(s). In this model, the B2B gateway acts as an eBusiness extension to an Enterprise Service Bus (ESB) or to a more conventional substitute of it (e.g. an integration broker). However, the gateway solution presented here also allows a tight coupling between message and service, when appropriate – hence the name "versatile gateway". In this usage, the B2B gateway behaves as a SOAP intermediary (in the sense given in SOAP 1.1 specification). This intermediary has its own address, and acts as a proxy to the ultimate WS endpoints.

Other SOA functional areas that may be subject to other SOA design patterns are:

- business transaction monitoring,
- repository access and profiling,
- service orchestration,
- agreements and policies management,