December 2003

Web Services Enabled Architecture for Interorganizational Business Process Management

Ajit Patankar
University of California, Berkeley

WEB SERVICES ENABLED ARCHITECTURE FOR INTERORGANIZATIONAL BUSINESS PROCESS MANAGEMENT[1]

Ajit Patankar
Haas School of Business
University of California, Berkeley
Patankar@haas.berkeley.edu

Abstract

This research is based on an extensive review of a pioneering interorganizational web services (WS) installation that has been in operation for almost two years. The case study is used to evaluate the current WS technology and identify the critical research issues. In particular, the review concludes that the WS technology is capable of meeting the business need of reusability and flexibility. However, new enterprise architecture is required in order to fulfill the promise of web services technology. The paper introduces dynamic notification based enterprise architecture for managing interorganizational business processes. We further explain why the architecture is necessary in order to support complex business processes.

Introduction

Over the last couple of years, there has been tremendous interest in web services (WS) technology as a platform for integrating e-Business transactions and processes. While much of the work is related to the WS tools and standards, much less attention has been paid to applications running in one company that could automatically invoke applications in another company, building bridges between systems that otherwise would require extensive integration and development efforts (see Aissi et al. (2002)). We refer to these systems as web services enabled interorganizational business process management systems (IO-BPM).

The authors have extensively reviewed a pioneering interorganizational web services installation that has been in operation for almost two years. Based on this review, it is concluded that the WS technology enables an organization to build flexible and reusable systems. However, the current enterprise architecture cannot leverage WS to support work-flow and collaboration in interorganizational business processes. We identify the information system architecture as one of the basic impediments and describe our recent work in this area. The contributions of this paper are as follows:

- Identification of architectural and technical issues which are impediments to achieving the full potential of WS.
- Proposal for an innovative enterprise architecture based on dynamic generation of process components and event notification.
- Analysis of the architecture in terms of ability to meet the requirements of WS enabled IO-BPM.

The rest of the paper is organized as follows: In Section 2, we review the relevant literature. The business drivers and the current WS based system are reviewed in Section 3 in order define the requirements of the proposed architecture. The Section 4 presents the DB-BPM enterprise architecture. The paper concludes in Section 5 with a description of further research.

[1]This research was partly supported by CITM grants including the DoD contract (2000-2001), N00244-00-C0108, and by the State of California NGI Program.

Literature Review

Considering the comprehensive view point of the paper, the relevant literature encompasses many areas. In particular, the relevant areas include WS tools and standards, enterprise applications of WS, WS orchestration, and business impact of WS. Here we discuss selected material from each of the areas.

The fundamentals of web services are described in several books, for example, Cerami (2002). The WS tools industry seems to be converging towards two implementation platforms, namely, Java and .Net (see Java (2003) and Microsoft (2003)). These sites also describe a large number of case studies but mostly from a marketing perspective. There are a large numbers of domain specific web services standards that are in various stages of maturity. A description of the OpenTravel Alliance (OTA) which is the most relevant for this work is available at Open Travel Alliance (2003). However, it should be noted that the system described in this paper precedes the OTA standard and work is currently on-going to provide an OTA compliant interface to the system.

The adoption WS as an implementation platform and glue in loosely-coupled enterprise systems is documented in several sources such as Lambros et al. (2001). The use of WS as a wrapper technology around existing applications and information assets is described in Gottschalk et al. (2002). A methodology for designing business web services is given in Aoyama (2002). Latency and scalability issues in web services, particularly in the context of client server model, are discussed in Litoiu (2002). This study indicates that the latency numbers in a local area network range from 20ms (optimized system) to 250 ms for a base system. The authors experience with CarRent and other WS implementations confirms these performance statistics and was one of the motivations in proposing the new architecture.

Web service orchestration has also received considerable attention both in the research and industry communities. The vendor driven standardization of Business Process Execution Language for Web Services (BPEL4WS) is described in BPEL4WS (2003). It appears that the term choreography is gaining acceptance to describe the orchestration field, and the standardization in this area is described in the W3C standardization group web site (2003). A complex orchestration application that is dynamically built based on customer requirements is described by Geng et al. (2003). An innovative e-Business process model that leverages WS concepts is described by Segev et al. (2002).

A process abstraction and verification approach is described by Choi and Zhao (2002). The process flow engine described in this paper is similar to this approach. There does not appear to be a case study that directly evaluates the architecture of enterprise information systems incorporating web services.

Requirements of the WS Enabled IO-BPM

To set the stage for the system analysis, we first describe the company and the business drivers that led to the development of a WS enabled information system.

Business Drivers

The authors extensively reviewed the design and operation of an interorganizational web service implementation at a major car rental company. In this paper, the company will be referred to as CarRent. The key business processes that CarRent needs to support are as follows: reservation, car delivery, and car return. The reservation process is the focus of this paper as it was the primary motivation for utilizing web services in this company. The reservation process embeds other sub-processes such as location services, pricing, and inventory availability. These sub-processes in general follow the work-flow sequence for car reservation. In this work-flow, the user first determines the nearest location from which she would like to rent the car. The location is used to find the type and price at which CarRent is offering cars at this location. As CarRrent operates at more than 250 locations in several countries, the price and availability can be determined only when the location is specified. If the customer accepts the quoted price, a reservation is made at that particular location.

CarRent's reservation process should be understood in the wider context of the larger travel industry eco-system which was one of the first industries to be significantly impacted by the Internet. Internet fundamentally changed the nature of intermediation and enabled suppliers to have direct access to customers. The industry, which is highly dynamic, consists of several segments such as airlines, hotels, and travel agents. The travel industry has recognized the need for interorganizational information systems

for a long time. The industry response to this need and cost consciousness has resulted in systems known as 'Global Distribution Systems (GDS)'. Apollo, Sabre, Amadeus, and Worldspan are examples of GDS see for example, Cendant Corporation (2003), Sabre Corporation (2003). A typical GDS such as Apollo has been developed over the last 30 years and provides connectivity to over 45,000 travel agency locations, 500 airlines, 33 car rental companies, 50,000 hotel properties, and numerous tour operators and cruise lines. A detailed analysis of business drivers and the company's competitive situation is presented by Segev and Patankar (2002). Here we discuss the following important drivers:

Channel Partnerships. There was a strong business imperative to build channel partnerships to augment and even replace existing marketing alliances. It should be also noted that the number and variety of channel partners is significantly higher as compared to most other industries. For example, an external partner can be a major airline with sophisticated IT infrastructure or a small travel agency with bare-bone systems. In such an environment, flexibility and reusability of the architecture is of utmost importance.

Alternative to GDS system. In principle, the GDS system could have been used to support the channel partners. However, the GDS systems are controlled by the airlines and have a high transaction cost.

Current Web Service Implementation

In order to identify the requirements of the WS enabled IO-BPM, we first describe the current system architecture. The operational experience with this system is used identify the requirements for the proposed IO-BPM.

The current WS based system replaced a legacy system which is briefly described to set the context. This system comprised of two main components -- a reservation system based on VAX/VMS and the EDI interface system. The reservation system incorporates over 15 years of investment and is considered a strategic asset that needs to be preserved. This system performs functions such as inventory management, reservation, and pricing. Like most legacy systems, this is a homogeneous system that lacks the component architecture. The EDI interface system works on a separate hardware platform that parses EDI files sent to CarRent by several channel partners. The traditional EDI technique is not suitable for rapidly supporting new partners as a customized adapter is required in most cases. Development of a custom adapter is a cumbersome process that can span several weeks of development time. In order to meet the business objectives, CarRent had to create a highly reusable and flexible link into the legacy reservation system.

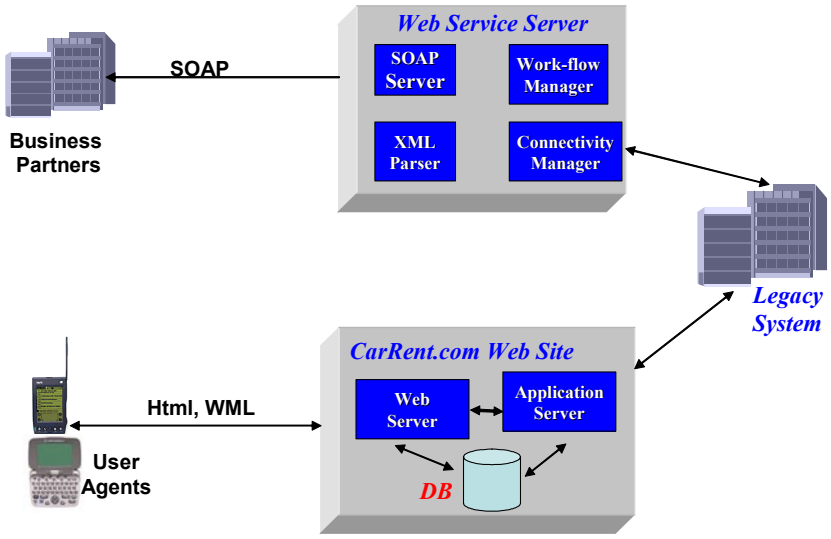

Figure 1. Version 2 of Web Services Implementation

Next we describe the current web services based system architecture and functionality. As shown in Figure 1, the core component of the architecture is a Web Services engine that runs on a hardware platform separate from the legacy reservation system. This component comprises of the following modules:

- SOAP message processor. The message processor communicates with web services consumers via the SOAP protocol. The message processor receives connection requests from web service consumers, sets up the connections, receives the connection, and closes the connection. The processor does not guarantee security or delivery which has to be ensured in higher application layers.

- WSDL and XML processors. A message embedded in SOAP packet is parsed using the WSDL contract and the required parameters are extracted in this step. These parameters are passed on to the connectivity modules.

- Connectivity modules. These modules abstract the functionality of CarRent reservation systems from the point of view of legacy system and communicate with it.

Description of Web Services

The system provides extensive services that cover almost all the functionality of the reservation system. While most of the services are designed for consumption in real time, some of the services can be used for caching on the client side. The services are as follows:

- Single valued WS for example, GetLocation, GetRate, GetRes, MakeRes, etc. These services can be further qualified using parameters such as price and location code.

- Aggregate WS, for example, GetAllLocations, GetCarTypes, etc. These also support qualifying parameters such as geographic area or car categories like med-size.

Note that it is not possible to build an atomic transaction by combining multiple web services. For example, GetRate is used to obtain a rate quote and MakeRes service is used to reserve a particular car at a particular rate. However, it is possible that a rate may change while the user reviews the rate obtained from GetRate service and decides to book a car at the quoted rate.

Motivation for the Proposed WS Architecture

The above implementation was considered as a significant business success as it allowed a new partner to access CarRent's reservation system with less than two weeks of development efforts as compared to several months of efforts in the legacy systems. To enable a new partner essentially requires creating a table entry for the access privileges and possibly developing new WSDL documents. Both these tasks can be done significantly faster in case of most of the partners.

The WS technology thus allows separation of internal business process management from process invocation. This re-engineering has ensured reusability in both of these areas. This has resulted in not only much lower cost but also met the strategic objective of building an IT infrastructure that has an ability to integrate a wide range of channel partners.

Note that the system as developed so far enables external partners to access the Car Rent's information systems. This part of the implementation was considered a tremendous business success. However, the review also pointed out the following issues with WS enabled system:

- Partner Legacies. The initial partners had hard-coded their systems based on CarRent's published WS interfaces. This in-turn created new 'legacy' applications in the partner's IT system.

- Internal Legacies. CarRent had directly exposed enterprise components such as reservation engine directly to partners. As partners were dependent on these interfaces, CarRent had a limited flexibility in modifying these interfaces.

- Architecture limitations. CarRent could support transactions generated by partners. However, CarRent could not incrementally enhance the architecture to support system enhancements such as those described below.

CarRent's next business objective was to support a broader range of travel services such as being a hub for vacation planning. A scenario that was used to evaluate the system involved a customer logging in to the CarRent's web site and dynamically building a customized vacation package. This required the CarRent information system to coordinate a long lasting business process that spanned multiple business partners. The scenario raises several issues in process coordination, process management, performance, and scalability. In this context, we identify the following requirements:

- Dynamic role resolution. The current system enforces a static binding between the role and actors in the system. This problem cannot be alleviated by using WS yellow page standards such as UDDI but is partly as a result of lack of WS support in channel partner's information systems.

- Powerful and flexible notification scheme. It is necessary to support asynchronous processes in order to implement interorganizational work-flow. The current WSDL/SOAP is a state-less, synchronous protocol which implies that the state management and asynchronous notification must be supported in the higher application layers.

- Mechanism for business constraint management. Many business processes are definitions of constraints over entities and processes rather than explicit invocation of individual web services. For example in the above travel industry scenario, the user would like to define an overall constraint of total vacation cost rather cost of only rental car. Current WS standards and capabilities cannot support these aggregate business constraints.

- Large number of diverse resources. In the traditional work flow systems, the number of resources and the types are limited. WS enable participation by a large number of diverse of resources in forming the interorganizational business process. For example in the above vacation planning scenario, the CarRent can request services from over five thousand hotels in order to complete the business process.

- Trust and quality of service (QoS). As resources are dynamically discovered and used in the process execution, trust and QoS becomes an increasingly important issue.

- Effect of long lasting processes. The case study shows that certain business processes can last for days or weeks as they may involve waiting for decision from a user. This significantly affects the system flexibility as a version of the business policy may have to be maintained for each process instance.

In order to address the above requirements, we concluded that the current enterprise information architecture is a key impediment in achieving interorganizational systems. Therefore we present an innovative architecture in the next section and show how it addresses these issues.

Web Services Enabled Enterprise Architecture

We introduce an innovative enterprise architecture called Dynamic Notification based business process management (DN-BPM) architecture towards a web services based interorganizational systems. Architecture overview is shown in **Error! Reference source not found.**

The DN-BPM architecture is based on a notion of materialized process components and addresses unique requirements that arise due to WS. Under the proposed architecture, each business process is composed from process components, resources, and coordination events. Components represent services that a business process needs to execute in order to meet the functional requirements. However, components are dynamically generated from the underlying enterprise components for each process instance. Resources are bound to a process component and execute the service. Resources are analogous to roles in work flow management systems as suggested in Stohr and Zhao (2001). An event based system manages the process flow and coordination of component services. Such a separation promotes flexibility in the architecture by allowing dynamic enhancement to the system along any direction, namely, services, or resources or coordination rules. Note that the current business process standards such as BPEL have a static binding between the components and resources as described in BPEL4WS (2003). Next we describe the architectural components and their interaction in details.

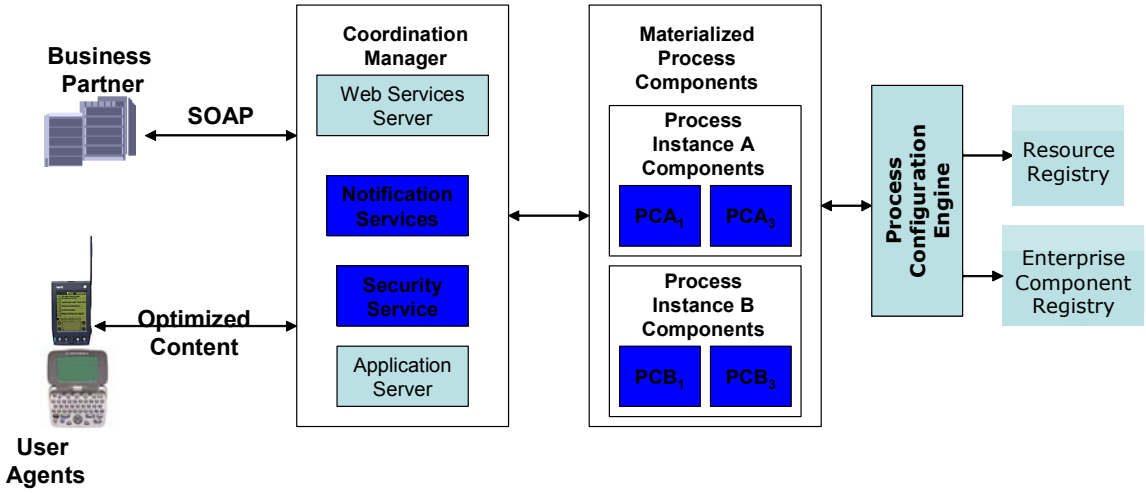

Figure 2. DN-BPM Architecture

Architecture Operation

A business process in DN-BPM is initiated with the occurrence of an event that may originate within or outside the enterprise. The event is notified to the coordination manager which initiates the following actions:

- Process Instance Creation. A process transformation engine correlates the event to a process specification to dynamically transform the process specification to a process instance. A single event can trigger creation of any number of process instances in each of which any aspect of the process instance (such as flow, resources, or coordination rules) may be context dependent.

- Binding with resources and rules. The process instance components are bound with resources and the accompanying rule base that governs the execution of process instance.

- Notification to initial set of instantiated components. The coordination manager notifies the initial set of instantiated process components so that they can start to execute the business process.

The initialized components are derived by applying transformation operations on enterprise components. Thus, there may not necessarily be a one-to-one correspondence between enterprise components and process instance components.

The instance of an execution model is incorporated in the coordination rules base. As the rules base can be distributed across the internet the architecture supports both centralized and peer-to-peer process execution models.

Architecture Components

We describe the main architectural components in this section.

Enterprise Components

Enterprise components abstract the services that enterprises offer. These represent distinct activities that a component can perform in a work flow management paradigm. Enterprise components are not directly accessed by interorganizational business processes for the following reasons:

- Contractual obligation. An organization does not want to contractually commit to providing exactly the same services for a long period.

- Security. A direct access to enterprise components may pose security risks particularly of the semantic type which are not well understood.

- Flexibility. Enterprise components that are directly exposed to interorganizational systems cannot be easily modified in response to changing business or technical situations.

Process Instance Components

The business process functionality is delivered through the process instance components. These components perform the mapping to the underlying enterprise components and use them to physically deliver the service. The mapping between the process and enterprise components is many-to-many. The notion of process instance components is a powerful one and allows construction of interorganizational business processes that are at variable level of granularity.

Process Configuration Engine

The process configuration engine creates an instance of a process based on the current context. The context can include: the process initiator (the user), the initiating event, and the current state of enterprise component and resource registries.

The engine creates two types of process flow -- static and dynamic. In the static process flow, the binding between the process instance components and resources is done a-priori to starting the process execution. In the dynamic process flow, one or more process instances are bound to resources after the process execution begins. The dynamic process flows provide more flexibility in implementing interorganizational business processes but require a more complex process configuration engine.

The detail design of a process configuration engine is a matter of further research and is beyond the scope of this paper.

Process Coordination Manager

The process coordination manager performs the following functions:

- Notification services. The coordination manager monitors external events and notifies them to the appropriate process components. The coordination manager can only communicate with process components as a direct communication with enterprise components would imply their direct participation in interorganizational business processes.

- Repository of events. The coordination manager also acts as a repository of events which cannot be notified immediately to process components. Examples of these events include fail over and asynchronous events which are generated during the execution of long lasting business processes.

- Manage dynamic process flow. In the dynamic process flow, if the currently executing process component does not know identify of the next process component then it communicates its status to the coordination manager. The manager invokes the process flow engine to determine the next process component.

Registries

There are two primary registries in the architecture – enterprise component and resources. The enterprise component registry is expected to be static in nature while the resources registry is expected to represent real time status of resources. For this reason the enterprise component registry can be implemented using UDDI standard. However, the current UDDI technology is not suitable for maintaining real time status of large number of diverse resources.

Enabling Factors

Web services are having impact in several areas including system design and managerial practices. Here we focus on selected enabling factors that are essential for the successful deployment of the new enterprise architecture.

- Partner Enablement: In the CarRent interorganizational system, only CarRent is web services producer while other partners are strictly consumers of the web services. It is essential that the partners should also become producers of web services. At minimum, CarRent can then invoke web services interfaces on these partners to 'push' static information so that it can be cached in their systems. The web services push mechanism ensures that the static data cached at partners site is current and can be used without rechecking it for each and every transaction. The push mechanism is also useful for coordination of activities that are initiated by CarRent. Such examples include marketing campaigns, notification of exception cases, and submission of billing or commission statements.

- Fine Grained Enterprise Components: The current CarRent legacy system is essentially exposed as a single enterprise component. Our review has shown that such a legacy system needs to be reformulated into fine grained components that can be more easily composed into process components. This reformulation would involve functional decomposition and possible off-loading of functionality to purchased software packages.

Architecture Evaluation

In this section we explain how the architecture supports the important requirements outlined in Section 3. In particular, we focus on support for a significantly higher level of performance and scalability in the proposed architecture. It may be noted that a WS service request can be processed in one the following ways: A request can be processed locally if the service component is hosted locally, a request can be serviced from a locally available cache, or a request can be processed remotely. These modes are illustrated in Figure 3.

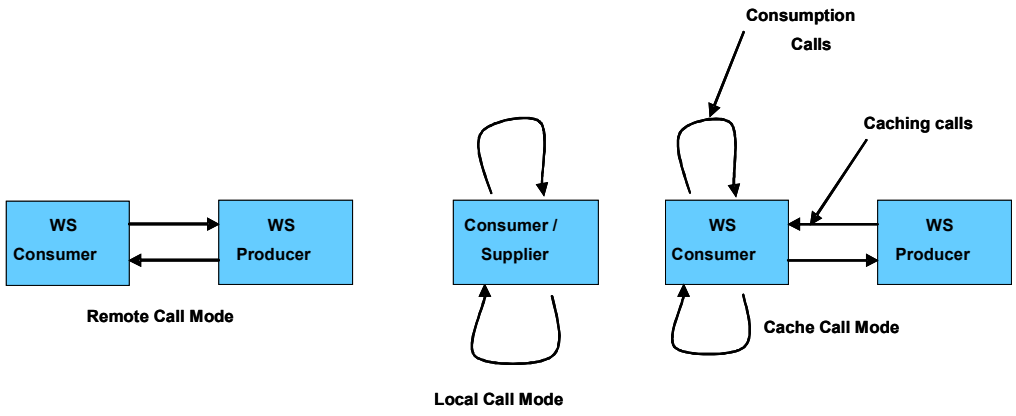

Figure 3. WS Invocation Modes

We argue that the metric for the performance and scalability of the system can be of the number of messages exchanged across the WS clients and servers to complete a transaction. This argument is illustrated with the help of the following model:

Cn = Cost of a WS message between a WS client and a server
Cf = Cost of fetching a message from a local cache
Cc = Cost of pre-fetching a message to a local cache
Cpl = Cost of processing a WS request on the local machine
Cpr = Cost of processing a WS request on the remote machine
Ct = Total cost of a WS transaction, where,
Ct = {min{Cn+Cpr}, {Cf+Cc+Cpl}}

We make the following simplifying assumptions: the processing cost at local and remote servers is equal, and the cache hit rate is significant. Under these assumptions, the above equation reduces to Ct = Cn. Therefore a WS architecture, such as that proposed in this paper, that facilitates multiple invocation mechanisms is likely to result in better performance and scalability.

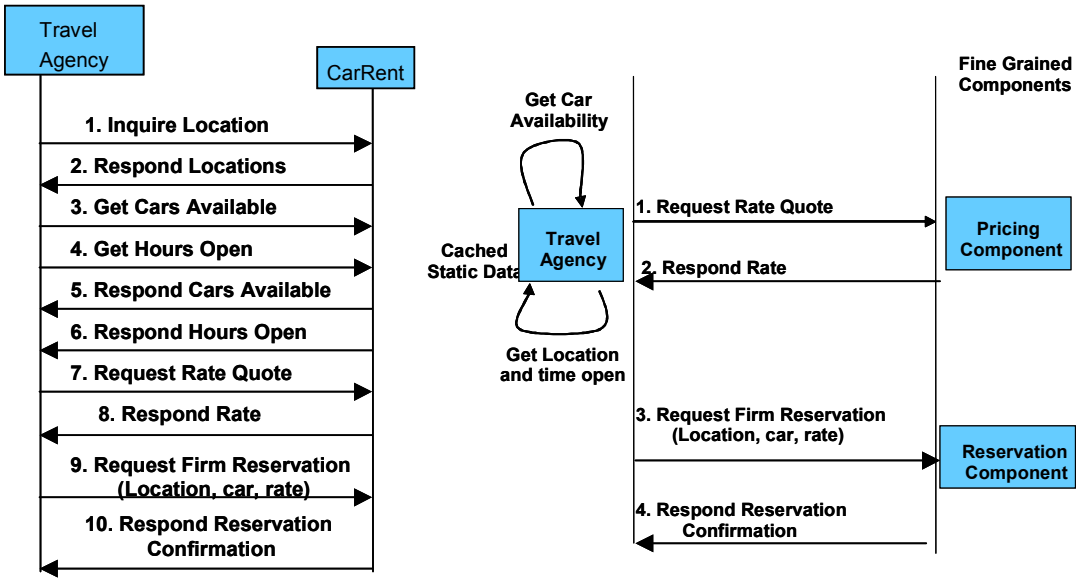

Figure 4 (A and B): Message Invocation Examples

Consider Figure 4 (A) which shows that the current system architecture supports only one mechanism for WS invocation; namely, a request is always processed remotely. As shown in Figure 4 (B), the proposed architecture supports all three mechanisms which may be needed to execute a complex transaction. This model shows that it is necessary to have the flexibility to support all invocation mechanisms but it does not address the decision about optimal hosting strategies for components.

Conclusions and Research Issues

We have described a pioneering interorganizational web service implementation in this case study. The primary business imperative for this company was to enable a large number of channel partners. The review concludes that the web services technology has fulfilled the initial promise of enhancing the flexibility of corporate IT infrastructure. We further evaluated the implementation to see if it can support complex interorganizational business processes. This led to the conclusion that significant enhancements are needed in the architecture of web services enabled enterprise systems. The DN-BPM enterprise architecture was presented towards achieving this objective.

Our current work focuses on validating the proposed architecture as well as the detailed design of the critical components such as process flow engine. The new business models that are enabled through web services technology also need to be researched. Other area of importance for further research is in-depth study of performance and scalability of interorganizational web services implementations.

References

Aissi, S., P. Malu, and K. Srinivasan. "E-business process modeling: the next big step." *IEEE Computer*, 35(5):55-62, 2002.

Aoyama, Mikio "A Business Driven Web Service Creation Methodology", *Proceedings of the 2002 Symposium on Applications and the Internet (Saint'02w).*

Cendant Corporation Apollo Web Site (2003). Online at **http://www.apollo.com.**

Business Process Execution Language for Web Services standard web site. Online at **http://www-106.ibm.com/developerworks/library/ws-bpel/**, 2003.

Cerami E. *Web Services Essentials.* O'Reilly & Associates, 2002.

Choi, Y and Zhao, J.L., "Matrix based abstraction and verification of e-business processes", *WEB 2002: The First Workshop on e-Business, 2002*, pp 154-165.

Java Software Web Site of Sun Corporation. Online at **http://java.sun.com**, 2003.

Geng, Xianjun, Huang, Yun, Whinston, Andrew B., "Smart Marketplaces – A step beyond Web Services", *Information Systems and e-Business Management*, 2003, 1:1-21.

Gottschalk, K., Graham, S., Kreger, H., Snell, J., "Introduction to Web Services Architecture", *IBM Systems Journal, Special Issue on New Developments in Web Services and E-commerce,* Vol. 41, No. 2, 2002, pp. 198-211002E.

Lambros, P., Schmidt, M., Zentner, C. "Combine Business Process Management Technology and Business Services to Implement Complex Web Services", *IBM Web Services*, May 2001, **http://www3.ibm.com/software/solutions/webservices/pdf/BPM.pdf**.

Litoiu, Martin "Migrating to Web Services – Latency and Scalability", *Proceedings of the Fourth International Workshop on Web Site Evolution (WSE' 02), 2002*.

Microsoft Developers Network (MSDN) website of Microsoft Corporation. Online at **http://msdn.microsoft.com**, 2003.

The OpenTravel Alliance Organization Web Site. Online at **http://www.opentravel.org**, 2003.

Sabre Corporation Web Site. Online at **http://www.sabre.com**.

Segev, Arie and Patankar, Ajit "A Travel Industry Case Study of a Pioneering Web Services System", Fisher CITM, Haas School of Business, University of California, at Berkeley, Working Paper.

Segev, A., Patankar, A., Kim, J., Zhao, L., "Active collaboration requirements in enabling interleaved eBusiness processes", *Web 2002, The First Workshop on eBusiness*, Barcelona, Spain, 2002.

Stohr, E. A. and J. L. Zhao, "Workflow Automation: Overview and Research Issues", *Information Systems Frontiers: Special Issue on Workflow Automation*, Volume 3, Issue 3, September 2001.

The W3C standardization group on Web Services Choreography. Online at **http://www.w3.org/2003/01/wscwg-charter**, 2003.