Architectures For Internal Web Services Deployment

Oded Shmueli

Dealigence
8 Hayozma St.
Tirat Carmel
Israel
oded@dealigence.com

Abstract

There is a new emerging world of web services. In this world, services will be combined in innovative ways to form elaborate services out of building blocks of other services. This is predicated on having a common ground of vocabulary and communication protocols operating in a secured environment. Currently, massive standardization efforts [UDDI, WSDL, ebXML, RosettaNet] are aiming at achieving this common ground.

We explore possible architectures for deploying computerized traders internal services. This encompasses both the structure and the functionalities of "traders" and "services" and the form in which these functionalities could be realized in actual implementations. We consider elaborate services such as negotiation services and marketplaces and their role in this new emerging environment

1. Web Services and Traders

The term *web service* is used to refer to functionalities that a web entity (the service provider) makes available to other web entities (the service consumer, or more generally, the *service trader* as the consumer may, in turn, be a provider as well). By making a web service available, sufficient information regarding the service is disclosed. This information includes meta-information regarding the service and its provider as well as specific technical data on how to programmatically activate the service. For a comprehensive exposition on the subject, see [IBM, UDDI]. This ``soft definition'' of web services and their traders leaves many issues to explore: how ``large'' are the services? What constitutes a trader? Is service combination to be done in real time?

Traders may be both producers and consumers of such services. As producers, they need be able to advertise their service, and receive and respond to service requests. As consumers they need to discover service providers, choose among them and make service requests. As traders, they need perform both functions simultaneously. In responding to service requests and in choosing a provider, there is a process that in real life involves negotiations on the terms of the deal (such as dates and payment). In actually providing a service, issues such as delivery, quality, monitoring and dispute handling must be considered.

One can differentiate between *common services* and *special services*. Common services are well-understood standardized services that are universally, or at least community wide, known and understood. Special services either apply to a relatively small community, or alternatively, are complex enough so as to defy a common standard. Business processes may comprise services and their flow may be integrated with other business processes. Of course, the division between common services and special services is not strict.

2. Computerized Traders and Internal Services

2.1 Computerized traders

A *computerized trader* is a software entity with (some or all of) the following functionalities: discovering, negotiating, contracting, activating and tracking the execution of web services. We make the observation that these functions themselves can be viewed as (self provided) *trader internal services*. A critical issue is whether these services themselves should be outsourced. For example, suppose a computerized server decides to

Proceedings of the 27th VLDB Conference,
Roma, Italy, 2001

run an auction on a particular business deal. Should the computerized trader arrange the auction? Solicit participation? Run the auction? Notify participants?

An advantage of being self-sufficient in terms of an internal service is having total control over the way this internal service is performed. In particular, this deployment method lends itself well to adaptation of the service to organization specific needs and integration with local systems.

A disadvantage is the need to develop in-house expertise in providing this internal service. Another disadvantage is that this internal service, which interacts with other parties' internal services, will tend to become non-standardized. Computerized self-sufficient trader products will tend to be bulky systems. Product, maintenance and training costs will tend to be high.

2.2 Outsourced internal services

A non-exhaustive list of such services includes:

- Discovery services
- Personalization services
- Negotiation services
- Payment services
- Trust and mediation services
- Delivery services
- Deal tracking services

The first service, discovery of appropriate providers and services, seems natural to outsource to a registry service. In fact, this is one of the prime motives for UDDI [UDDI]. The main reason is that the alternatives are not economically feasible. A push registry service, in which suppliers broadcast information to potential consumers, or a poll oriented solution in which a consumer exhaustively searches for potential providers, are both inefficient. The registry service is viewed as a foundational service that enables the deployment of other services. The other services listed above may be outsourced as well. For example, Microsoft's Hailstorm [HAILSTORM] initiative is an example of outsourcing personalization services.

Outsourced services tend to be generic, which makes it more difficult to specialize them to particular trader needs. Since these services are not local, they are subject to network ailments such as congestion, delays and unavailability. They also raise issues of trust and impartiality of the remotely supplied service.

The fact that a service is local makes the deployment of the major software components easier. Still, even when services are outsourced, they need be accessed. The local integration of the outsourced services will display some of the weaknesses of local self-sufficient services.

3. Example: automatic negotiation services

3.1 Automatic negotiations

Negotiations are an essential part of obtaining products and services. Computerized traders need be able to handle both one-to-one (1-1) negotiations as well as one-to-many (1-n) negotiations. Furthermore, if negotiations are to be carried out simultaneously with many parties, in different geographic locations and time zones, they need be automated (partially or fully) [ECONTRACTS].

Let us first consider 1-1 automatic negotiations. A typical scenario is as follows. A computerized trader is looking for products or services. The trader is handed (or constructs) a description of the deal, which includes constraints, preferences and tradeoffs. The trader needs to discover appropriate providers, negotiate with them and choose one or more providers. We assume that initial discovery is done via registry or prior knowledge.

Consider the process of 1-1 negotiations. Such negotiations are influenced by many factors:

- Compatibility of supply and demand. Generally, this leads to an *offer matching process* whose complexity depends on the expressiveness of the offer specification language [ECONTRACTS].
- Enterprise data (e.g., product, or personnel availability). Observe that such data may necessitate communication with enterprise systems.
- Company policy (e.g., specifying preferred customers, volume discounts) as may be expressed by business rules.
- External events (e.g., competitors' offers).
- Negotiation profile. This factor is influenced by temporal constraints on the deal (e.g., a strict deadline for obtaining raw material), by the party's eagerness to conclude a deal (is it a must deal or an optional one), and, more generally, by the negotiation tactics. Such tactics need address the issues of negotiation format, starting offer, how to end the negotiations and more.

As can be seen by the above description, 1-1 negotiation is a complex process. Such a process may form the core of a complex system (see [DEALIGENCE]). This complexity makes it a candidate for outsourcing (completely or partially).

We next outline possibilities for how such an internal service may be deployed (variations are possible).

3.2 Self-sufficient traders

Here, each trader is equipped with a subsystem for handling 1-1 negotiations. The subsystem receives as input a description of the required products or services and negotiates on behalf of the trader. In doing so, it

utilizes the constraints, preferences and tradeoffs, as well as the trader's negotiation profile, external data and business rules. The subsystem manages the negotiations (including offer and counter offer generation, timing decisions, accept/reject decisions etc.).

3.3 A negotiation consultant

Similarly to a registry service, there is a negotiation service. The service operates on behalf of a party to a negotiation. This service's input includes the party's profile, its constraints, preferences, tradeoffs and other relevant data, as well as the current state of the negotiations. The output is a recommendation as to how to proceed in the negotiation. One can think about this service as a negotiation consultant.

3.4 A negotiation service

A negotiation service receives as input the general specifications of a deal, as already agreed by the parties, and relevant data describing the two parties and their profile characteristics. The service performs the negotiations on behalf of the parties. For that, it may need to interact with the parties' local enterprise systems, be aware of external events and be privy to parties' business rules. As it services both parties, it should enjoy a high degree of trust. Conceptually, an *agent* faithfully represents a trader and negotiates on the trader's behalf according to negotiation directives.

3.5 A marketplace service

The idea is to bundle a number of related services into a *marketplace service*. Such services may include matching of business intentions, splitting and merging of deals, deal tracking, payment and dispute resolution as well as the negotiation services. Conceptually, an *agent* represents a trader and it faithfully negotiates on the trader's behalf.

A major advantage of bundling together these services is that it saves greatly on local integration. It also provides for a natural meeting place, which encourages the offering of additional services. One disadvantage is the need for choosing a marketplace and having trust in its participants and management. Marketplaces can be private (selected participants) or public (open to all).

We briefly discuss outsourcing 1-n negotiations (auctions). While auction mechanisms and rules can be quite complex, their execution is simpler as compared to 1-1 negotiations. However, administration of the process and ensuring trust in its integrity can be non-trivial. Another important factor is that counterparts naturally populate marketplaces. The consultancy and service models do not seem to add much value. The above observations suggest outsourcing auctions to marketplaces.

4. Conclusions

Web services are becoming a reality. Some services are considered foundational due to their widespread use and due to economic considerations. We consider internal computerized trader services which are on the border between foundational and specialized services. We observed four possible basic deployment styles: local, outsourced consultant, outsourced service and bundled (with other services). All four deployment styles have pros and cons and we expect to that various internal services will be provided in all these modes.

Acknowledgement

I would like to thank Dick Tsur and Dan Fishman for helpful discussions.

References

[ECONTRACTS] Leiba Lior, Konopnicki, D., Shmueli O., and Y. Sagiv. A Formal Yet Practical Approach to Electronic Commerce. Proc. COOPIS '99, Edinburgh, Scotland, September 1999.

[DEALIGENCE] www.dealigence.com

[ebXML]
http://www.ebxml.org/white_papers/whitepaper.htm
http://www.ebxml.org/

[HAILSTORM] Building User-Centric Experiences
An Introduction to Microsoft HailStorm, a Microsoft white paper, March 2001.
http://www.microsoft.com/net/hailstorm.asp

[IBM] Web Services architecture overview. The next stage of evolution for e-business. IBM Web Services Architecture Team. September 2000,
http://www106.ibm.com/developerworks/webservices/library/w-ovr/?dwzone=webservices

[RosettaNet]
http://www.rosettanet.org/rosettanet/Rooms/DisplayPages/LayoutInitial

[UDDI] UDDI Technical White Paper, September 2000. Copyright by Ariba, IBM and Microsoft.
http://www.uddi.org/whitepapers.html
UDDI Programmer's API Specification, UDDI Data Structure Reference, UDDI XML schema.
http://www.uddi.org/specification.html

[WSDL] Erik Christensen, Francisco Curbera, F., Greg Meredith, G., and S. Weerawarana: Web Services Description Language (WSDL) 1.1, W3C Note 15, March 2001.
http://www.w3.org/TR/2001/NOTE-wsdl-20010315