Component Service Providers: An Evolution in Component Management

Russ Bunting, Objectools.com Ltd., Toronto, Canada
russ.bunting@objectools.com

Keywords:
component service provider, component management, web services

Research Problem

In consideration of the questions of how components are assembled, how components are managed, how system level properties are ensured (e.g. safety, responsiveness, availability), and how systems comprising hundreds of components are managed, the notion of a Component Service Provider (CSP) supplying component management and provisioning is desirable given the challenges and complexities of the tasks at hand.

During the last decade of so, the migration of application functionality from centralized server to numerous and heterogeneous nodes across a network has resulted in an evolution in the way applications are perceived, designed implemented and updated [1]. This more modular approach, in which an application is considered to be an interacting set of services in which each service is as self-contained as possible and accessed through a well-defined interface relies upon a number of elements to be in place [1]:

- Design approaches, methods and tools that support service based software architecture
- A shared infrastructure that facilitates separation of service interfaces, allows potential service providers to be identified, and brokers interconnections among service providers
- A marketplace of services from third party suppliers in the form of components that can readily be connected through the shared infrastructure

As well, supporting technology must be available to allow application developers to browse collections of components, select those of interest, and assemble those components to create the desired functionality [1].

Analogous to these increasingly common conceptions of component-based development and component-based systems is the notion of Web Services. A Web Service knows its attributes, how to perform its discrete task and how to work with other Web Services seamlessly thus shielding the user from the low-level details involved in locating the data and using the applications. Furthermore, Web Services are self-contained, self-describing and location independent [2]. Some typical examples of Web Services include Enterprise Resource Planning system vendors providing discrete functions over the internet by "XML-izing" their applications.

In order to offer Web Services, a Web Services Automation System must build on the Web Service basics of being highly dynamic, capable of cross boundary communications, location independence, and being manageable with granular security [2]. Similar service challenges have been addressed in distributed application architectures of particular products like Microsoft's Windows Distributed interNet Applications (DNA) architecture and also in the creation of standards such as the Object Management Group's Common Object Request Broker Architecture (CORBA) and Interface Definition Language (IDL). For example, Windows DNA provides a comprehensive and integrated set of component services including just-in-time activation, object pooling, load balancing, in-memory database, queued components, automatic transactions, role based security, and events [3]. CORBA's services include naming, events, life cycle, object trader, transactions and concurrency, object security, persistence and externalization, query and collections, object relationships and time, and licensing [4]. It is anticipated these service descriptions will aid in identifying the necessary service offering of the Component Service Provider.

Relationship to Application Service Providers

Along with identifying what a Component Service Provisioning model might offer it is also interesting to investigate how a CSP might actually be implemented. In both tasks, a study of existing Application Service Providers (ASP)

and their similarities and differences to the Component Service Providers is warranted.

The term ASP is a generic term to describe a company that hosts entire applications on centralized servers. Customers pay a hosting fee or monthly rental fee to access these applications either via the internet or a private network. A complete application service consists of application hosting, application delivery and application technical support [5]. One may conjecture that a Component Service Provider is nothing but a fine-grained Application Service Provider and therefore of marginal interest. We propose, however, that a CSP, while sharing a number of attributes with an ASP, has additional attributes with regards to both development and deployment. While an application service provider supplies an entire application over the internet, a Component Service Provider offers a service, through a defined interface, to applications running on a network [6]. Furthermore, key differences include the heterogeneous nature of a CSP with regards to the elements, i.e. components versus applications. As resulting applications may each use a different subset of components from perhaps a number of different CSPs there are required tasks in the assembly and management of the components, as well as how system level properties are ensured. Also for consideration is how existing component development tools and technologies will be integrated with the CSP, a challenge not evident for a typical ASP.

One item that is definitely similar in both ASPs and CSPs is that fundamentally the ASP challenge is to develop an efficient, i.e. where application performance does not depend on network bandwidth, internet-based architecture, which will efficiently provide access to software applications over the internet [7], or components in the case of a CSP. In this regard, advances in architectures for Application Service Providers will be examined in detail.

Next Steps

This project is in is inception phase in that we are formalizing the goal of the research project and identifying possible deliverables. Objectools.com Limited has built an open component marketplace and has commissioned this research project to understand the applicability, technical challenges, adoption criteria and other issues prior to undertaking development of a Component Service Provisioning pilot project. It is expected that this research project will examine the Component Service Provider approach in the context of modern component-based development. As a result, challenges and risks are to be identified as well as a critical path to the realization of a Component Service Provider model.

The research will be driven forward by addressing the following questions:

- Is there a minimal set of the commonly described distributed application services (i.e. those identified in the Windows DNA architecture and CORBA specification) necessary to realize Component Service Provisioning?
- Are there other distributed application services (i.e. not identified in either existing products or existing standards) required to realize Component Service Provisioning?
- Are certain component types better suited to a CSP approach than others?
- Are certain application types better suited to a CSP approach than others?
- Are certain approaches to CBD better suited to a CSP approach than others?
- In regards to the relationship between component-based systems and the evolving offerings of Web Services, what role does XML (eXtensible Markup Language) play in component service provisioning?

References

[1] A. Brown, "CASE in the 21st Century: Challenges Facing Existing CASE Vendors", IEEE Computer, 1997
[2] F. Moss, "Web Services: Computing's Fourth Wave", http://www.bowstreet.com
[3] G. Eddon, "COM+: The Evolution of Component Services", IEEE Computer, July 1999
[4] C. Syperski, "Component Software: Beyond Object Oriented Programming", Addison-Wesley, 1998
[5] http://www.aspisland.com/services/model.asp
[6] http://tbtf.com/jargon-scout.html#csp
[7] Furht et al., "An Innovative Internet Architecture for Application Service Providers", Proceedings of the 33rd Hawaii International Conference on System Sciences, 2000