Chapter XV

Case Study Implementing SOA: Methodology and Best Practices

Gokul Seshadri
Architect, TIBCO Software Inc., USA

ABSTRACT

SOA or service-oriented architecture is an innovative approach to enterprise application integration that increases the benefits of EAI by means of standardizing the application interfaces. This chapter explores the gradual evolution of SOA through various phases and highlights some of the approaches and best practices that have evolved out of real-world implementations in regional retail banks. Starting from a step-by-step approach to embrace SOA, the chapter details some typical challenges that creep up as the usage of the SOA platform becomes more and more mature. Also, certain tips and techniques that will help the institutions maximize the benefits of enterprise-wide SOA are discussed.

INTRODUCTION

Service-oriented architecture or simply SOA is seen as the new face of enterprise application integration (EAI). By covering application-specific touch points with business-oriented interfaces, SOA is able to provide better design, agility, reusability, and maintenance savings, and has become the choice for the EAI approach.

This chapter details various steps involved in embracing the technology at an enterprise level, right from conception to enterprise-wide implementation. Throughout the chapter, we will be highlighting the approach using a standard example: A regional financial institution that attempts to embrace SOA in a methodical manner to realign its IT architecture with the business vision and goals.

International financial institutions have always been plagued with EAI problems. They are one of the earliest breed of business communities to quickly embrace SOA concepts and theology, and hence we chose this example. However, readers can quickly relate these examples and situations to any business community or sector they are associated with.
PROBLEMS WITH TRADITIONAL APPROACHES

Traditionally, financial institutions such as banks have been part of the earliest business institutions to adopt computerization because of the obvious benefits a mechanical computation machine brings to financial transactions. It is common to find traditional monolithic applications and programs built using procedural languages like COBOL and C in use in such institutions, even today.

Since these business applications could not function as independent silos, they had to communicate with each other. One required the data and semantics from another system in order to complete the desired business function. So, point-to-point communications interfaces were established between the applications. By point-to-point interface, we mean Application A directly hooking onto application B by means of whatever communication protocol that can be adopted.

With more and more applications seeking to communicate across one another throughout the enterprise, the number of point-to-point interfaces also increased dramatically.

The last decade saw the growth of professional prepackaged vendor applications meant for specific lines of business like loans, trade, or treasury. These boxed applications, once again, had to connect to existing applications and programs in order to achieve their functionality. The choice was, once again, point-to-point interfaces.

Substantial numbers of these point-to-point interfaces were batch programs, running at the end of the business day or end of the business week or month. Thus, when Application A updated a particular customer record, this change was available to Applications B and C only the next day or next week.

As banks expanded their business horizons into other related areas like selling credit cards and insurance products, the back-end applications and systems also grew in number. With the internationalization of business, every bank

Figure 1. Point-to-point interfaces for integration