

SOA and Payments Integration

SOA, Web Services and Payments Integration

SOA is sweeping the IT world as institutions strive to increase business agility and competitive edge. Major vendors such as IBM, BEA, Microsoft, Information Builders, Software A.G., Oracle, HP and SAP have been on the acquisition trail and are gearing up to satisfy the burgeoning market demand for SOA (and in particular, SOA using Web Services) solutions.

Yet despite this intense activity, SOA has yet to achieve serious penetration in the payments sector. The main reasons for this are concerns that an SOA/Web Services approach would have a negative impact on throughput (due to the underlying http/XML/SOAP concepts used) and due to the complexity of the payments message transformations necessary for a Web Services Adaptor to achieve integration with legacy payment systems.

Web services adaptors

A Web Services Adaptor is the wrapper which envelopes an existing system and exposes its functionality in the form of a Web Service which can be requested from, and provided to, other Web Services using SOAP/XML within an SOA environment.

A multi-application existing system may have multiple Web Services Adaptors, each exposing a particular facet of the existing legacy functionality in Web Service form.

In the case of transactional payment systems, as well as performing other functions, one of the key roles of a Web Services Adaptor is message transformation, taking messages in existing system format (often

some variety of ISO 8583 in a national or international format or in a proprietary format) and transforming them into XML.

Payments system interfaces

Historically, creation, adaptation and maintenance of new interfaces for payment systems has been a persistently high cost, continually recurring issue.

Unfortunately, Web Services Adaptors from the generalist SOA/Web Services vendors still require substantial custom coding to handle the foibles of payment system interfaces. So the message interfacing issues do not go away with the advent of Web Services, they persist.

Added to this is the fact that using Web Services is synonymous with sending and receiving XML messages in a SOAP envelope over http – http is not a high performance protocol and the use of XML implies the need for an XML parsing layer.

Hence, the very concepts which are fundamental to Web Services may compromise the ability to achieve the high transaction throughput rates demanded in the payments space.

The way forward

For SOA/Web Services to gain acceptance in the payments market, the issues of performance and flexible message integration with legacy systems need to be resolved. Fortunately, Alaric's products provide such resolution.

Alaric's payments integration middleware

Alaric's payments integration middleware products, Message Mapper and Authentic Gateway are ideal for organizations which are already deploying an SOA or which wish to make initial,

incremental moves in the direction of SOA.

Written in Java and designed with the payments industry in mind, these products are capable of ultra high performance payments message transformation while enabling interfaces to be easily created and managed via a point-and-click GUI, without programming.

With their ability to readily transform payments message formats into XML, Alaric's products can be used as the kernel of high performance Web Services adaptors or may be deployed in their own right, with a view to moving to an SOA in the future.

A key attribute of SOA is the emphasis on preserving historic investments in legacy systems by exposing their functionality for use as Web Services via a Web Services Adaptor. Authentic Gateway is ideal for deployment in this mode, shielding legacy systems from future change while leaving the door open to move to a full Web Services implementation at a later stage.

Alaric's products help customers make quick wins, enabling them to reap many of the benefits of SOA (e.g. efficiency of development, service reuse, smooth evolution, increased agility, increased return from existing IT investments, reduced integration costs) without necessarily having to embark on a full blown enterprise-wide SOA initiative.

