SON/SOA Application Development Environments for Enterprise Communications

IEEE GlobeCom 2008

Sean Moore, Avaya
smoore@avaya.com
978-677-5232 (US)
Overview: SOA Application Development

• **Business Driver:** Today, communications platform providers and service providers depend on partners to develop applications which subscribers use to consume provider-hosted (SOA) services

• **Provider Response:** Must enable partners by
  1. Exposing rich services API’s using standard protocols (e.g. WS-* web services, Parlay X, CSTA, etc.)
  2. Providing Software Development Kits -- **SDK’s** -- that minimize partners’ \{time, cost\}-to-market for their applications and reduce provider’s support costs
     – Personal observation: Providers focus on (1) but often overlook (2)

• **Key Observation:** Effective SDK’s assist partners with developing applications with a given software architecture
     – vs. traditional SDK’s which often just document API’s
     – **Architecture-centric** SDK’s vs. **API-centric** SDK’s

• **This Presentation:** How do we design and deliver effective architecture-centric SDK’s for application development in the modern SOA-web-telephony context?
Step 1: Identify SOA Applications Software Architecture Segmentation

- Segment partner developer market and SDK portfolio by application architecture type (vs. which API’s they use):
  - Phone/Endpoint applications
    - Example: Phone XML scripts pushed to phone browser user interface
  - Web applications/Web Design patterns
    - i.e., applications delivered into/accessed by web browsers
    - Example: Desktop web browser application supplementing associated deskphone
  - Speech/Call Control/Contact Center applications (Service orchestrations)
    - Example: Customer self-service applications (1-800-401K-PLAN) executed by CCXML/VXML engines
  - SIP Applications
    - Example: Named application (1-800-RINGTONES) hosted by a (JSR 289) SIP A/S
  - Business process/vertical integrations (API-centric)
    - Example: Communications web services integrated with IBM Sametime or SAP
  - Product Integrations (API-centric)
    - Example: Call recorders integrated with IP PBXs

- Align SDK portfolio with this segmentation
  - Deliver best-in-class SDK’s for creating phone apps, web apps, speech apps, SIP apps
Unified Communications Access Model

- In Unified Communications, a user can access all services using any application type.
- Each Application type has access to all SOA communications services.
- SDK’s needed to build applications for each access channel type vs. SDK’s for each API.
SDK Case Study: Web Applications & SDK

Expected User:
Communications-enabled
Operations Mgr/Agent
w/real-time situational
awareness

Partner-developed web application:
Developed as extension of
Vendor-provided web application
framework

Web application framework:
JSP & AJAX-based framework for creating
Enterprise communications web
Applications.

Web App Development Environment:
Web libraries, Development Lab
configuration, IDE, sample application,
Documentation, tech support

Traditional API-centric SDK's typically
provide support for API layer only

SOA Communication Services
Accessed or consumed
by web applications

Native API’s Layer
Telephony  Messaging  Conf.
Recording  Presence  Other Svcs

JEE A/S
(IBM Websphere, Red Hat JBoss)
Components of a Web Applications SDK (1 of 2)

• Beyond API’s for the (SOA) Services, a web applications SDK includes:

• **Web libraries**: Estimate that 80-90% of the software for a communications web application is generic and may be provided by the vendor as a web application framework.

• **Web Development Laboratory**: Developer lab configuration consisting of SOA services platform, services emulators, web server with web app framework.
  
  – Provide a laboratory blueprint w/detailed instructions for installation and operation; or
  
  – Provide hosted development laboratory so that partners do not incur laboratory overhead cost.
Components of a Web Applications SDK (2 of 2)

• **IDE with web development plugins**: Recommended IDE framework -- Eclipse, NetBeans, Visual Studio, etc. -- configured with plugins for web development support (JSP, JSF, Javascript, XML, etc.)

• **Sample Application(s)**: Sample web application designed to fulfill three {roles, purposes}:
  
  – **Training tool**: For developers to learn the AJAX libraries and recommended architecture for telephony web applications
  
  – **Application template**: For use as a basis/starting point for specific application, because all of the AJAX client software for service and event consumption is already coded.
  
  – **Reference application**: For testing and debugging specific web applications by comparing to generic client web application which is known to be correct (hopefully!)

• **Documentation**: Concept of operations, installation and operational instructions for development lab, how to use recommended IDE configuration to make non-trivial modification to sample applications, how to extend the framework to include other (SOA) services, …
Recapitulation

• SOA Application Enablement:
  – Adopt Partner-centric view
  – Define communication application architecture segmentation
    (which likely maps 1-to-1 to a partner developer market segmentation)
  – Create Architecture-centric SDK’s for each segment
    • Frameworks and libraries
    • Low-cost development laboratory blueprint or hosted environment
    • IDE, plugins, and usage instructions
    • Sample applications
  – Target 10X reduction in partners’ {time, cost}-to-market for their applications (vs. API-centric SDK’s)
  – Target 10X increase in volume, scope, and functionality of partner-developed applications