Sabre Web Services
Advanced Practices

John MacDonald, Implementation
Michael Kato, Product Technical Consultant
Session Management
John MacDonald, Implementation
Session Management

Session Management is:

- Managing *Sabre Web Services* connections
- Efficient
- Reliable

Session Management is not:

- *State* maintenance
  - Sabre PNR state (work area or AAA)
  - A client-side “shopping cart”

Service State

- *Sabre Web Services* are *stateless*
- Sabre *functionality* exposed via *Sabre Web Services* may be stateful or stateless
Three Approaches

Basic Connection
- Conversational
- Appropriate for low volume

Connection Pooling
- Persistent connections
- Efficient use of multiple sessions

Connection Management
- Protected connections
- Maximum reliability
- Highly scalable
Basic Connection

Conversational Strategy

1. Open connection (*SessionCreateRQ*)
2. Issue *Sabre Web Services* call(s)
3. Close connection (*SessionCloseRQ*)
Basic Connection

Pros
- Simple architecture
- Lowest cost
- Adequate for low transaction volume

Cons
- Little or no failover / session recovery
- Longer recovery after failure
- Limited scalability
Connection Pooling

Persist & Share Connections:
1. Send *SessionCreateRQ* to establish a pool of open connections
2. Client(s) connect and send *Sabre Web Services* calls
3. Send *SessionValidateRQ* or *OTA_PingRQ* as needed
4. Clear Sabre work area (AAA) between workflows
5. Send *SessionCloseRQ* to end connection
Connection Pooling

Pros
- Supports a larger number of clients
- Fewer web service calls
- Multiple simultaneous service calls

Cons
- Higher complexity
- Requires additional hardware
Connection Management Case Study

A Sabre Web Services customer that had been using the Basic Connection strategy implemented Connection Pooling following a sharp increase in volume

Before: Basic Connection
- Average volume >1 million SWS calls per day
- Averaging 7,000+ SessionOpenRQ and SessionCloseRQ calls per hour

After: Connection Pooling
- Averaging 20 SessionOpenRQ and SessionValidateRQ calls per hour
- Savings
  - Average 200 MS response for each Sabre Web Services open and close call
  - System resource savings of 23 minutes per hour no longer opening/closing sessions
  - 9.3 hours of system time per day no longer spent on connection overhead
Connection Management

Horizontally Scalable Implementation
Connection Management

Fully Redundant Implementation (HOT/HOT)
Connection Management

Pros

• Highest reliability
• Fastest failover recovery
• Maximum connection efficiency
• Highly scalable
• No single point of failure

Cons

• Cost
• Complexity
More Information
Session Housekeeping

- Default connection timeout is 15 minutes
- Activity resets the timeout clock
- Services to keep connections alive
  - SessionValidateRQ
  - OTA_PingRQ
- Session Manager is a proxy for clients
  - Client transaction updates session ‘last used’ timestamp
  - Session Manager sends keep alive on behalf of client
- Security
  - Only the Session Manager has user ID and password
  - Clients cannot open connections
- Weekly housekeeping
  - Close and re-open all persisted sessions
  - Sessions opened after 09:00 US CT are not subject to NORM OAA
Session Housekeeping

A Sabre host application known as NORM OAA is run system wide every Sunday morning at 00:15 U.S. Central Time. NORM OAA takes about 5 minutes to run and clears every AAA in Sabre including those used by Sabre® Web Services customers. All active Sabre users are required to log out and log in again. Binary security tokens used by SWS are invalidated when NORM OAA runs.

NORM OAA begins at 00:15 CT with a 3-minute warning period during which any Sabre entry returns a notice that system housekeeping is about to run. If action is not taken the USG_INVALID_SECURITY_TOKEN error will result the next time an SWS service call is executed.

This development pattern describes NORM OAA in detail and suggests remediation strategies for Sabre® Web Services customers using session pooling to manage connections.

Last Update: April 26, 2011
Sabre Web Services
Advanced Practices
Michael Kato, Product Technical Consultant
Parallel Threads

Practice:
• Use parallel pooled sessions rather than multi-threaded requests using a single security token

Benefits:
• Avoids message collisions
• Increased performance
• Simplified error handling
• Simplified troubleshooting (using ConversationID)
Weekly Session Housekeeping

Practice:

- System maintenance requires every active Sabre user to log out and log in again once a week
- Sabre Web Service (SWS) subscribers that keep inactive SWS sessions alive must close and re-open the session between 09:00 Saturday and 00:00 (midnight) Sunday (US Central Time)
- Sessions opened after 09:00 US Central on Saturday are not subject to the weekly housekeeping (NORM OAA)

Benefits:

- Reduced session errors following Sabre house keeping
- Booking in progress will be preserved
- Pattern document *NORM OAA Guide* available on the DRC
Data Compression

Practice:
• Useful for services with large response payloads such as Bargain Finder Max, Hotel Services, Fare Services.
• Can be enabled for individual or for all messages
• Supported by all HTTP/1.1 compliant servers

Benefits:
• Reduced response message size
• Additional network bandwidth
• Faster response time (depending on network)
Use The Most Current Version

Practice:
• Plan migration to latest service version
• Begin integrating 2.x versions
• Test when latest version deployed in CERT

Benefits:
• Enhancements and fixes
• No impact from service version deprecation
• 2.x simplifies migration to the latest version
Use SabreCommandLLSRQ Only As Necessary

Practice:
- Use versioned service rather than SabreCommandLLSRQ
- Use only when there is no SWS for a Sabre Host command

Benefits:
- XML response easier to develop, troubleshoot, maintain
- Access to non-Host functionality (e.g., PassengerDetailsRQ)
- Updates for both SWS and underlying functionality
- SWS handles multiple host responses
Orchestrated Services

Practice:
- Use Orchestrated Services in lieu of or in addition to individual Sabre Web Services calls

Benefits:
- Fewer calls required to complete a transaction
- Improved application performance
- Reduced network bandwidth
- Improved error handling
- Better segment status management (*Enhanced_AirBookRQ*)
- Orchestrated Services Include:
  - PassengerDetailsRQ
  - EnhancedAirBookRQ
Request What You Need

Practice:

• Tailor your requests to what you really need
• Get to know the options available in a service
• In some cases, making a full request with minimal qualifiers will return more information and reduce additional requests
• In other cases, asking for only what is required will reduce the size of the response improving performance and reducing overhead

Benefits:

• Scan reductions and overall performance improvements are possible when many requests can be consolidated into one request
• Tailored requests will reduce latency and overhead, and provide just the information required
Event Notification Services (ENS)

Practice:
- Use ENS rather than polling to detect changes to PNR, Queues, and Profiles

Benefits:
- Efficiency in client applications
- Reduced scan cost
Monitor Communication from Sabre

Practice:

- Read and save communication from Sabre
  - Webservices broadcast announcements
  - SWS Release notes
  - SAN *(Sabre Advance Notification)*

Benefits:

- Advance notice of fixes, updates, enhancements
- Leverage new features and business opportunities
- Help budget maintenance capital and resource
Stay Connected

facebook.com/SabreTravel

twitter.com/SabreTN

youtube.com/SabreTravelNetwork

agentstream.com

sabretravelnetwork.com/blog