Service Gateway Manual

Quarter 3: February 2016
After reviewing this manual, you should be ready to start development.

**Expectations**

You should be familiar with the following:

- Ability to read XML and SOAP documents
- Bindings
- Microsoft® Windows Communication Foundation (WCF)
- Operations
- Schemas
- WCFs
- Web Services
- Web Services standards as defined by W3C ([www.w3.org](http://www.w3.org))

**jXchange Interface Manual**

New and innovative tools are required to keep pace with the rapidly changing requirements in today's financial industry. The exchange of data between multiple types of products must be flexible and immediate. jXchange is designed to provide a unified interface for the entire suite of JHA products, enabling easier integration with consumer applications.

jXchange has been developed as the integration tool for the Jack Henry core products:

- SilverLake
- CIF 20/20
- Core Director

jXchange has also been developed as the integration tool for these Jack Henry complementary products:

- 4|sight
- Argo
- Broadcast Event
- iPay
- NetTeller Host
- NetTeller IS
- PassPort
- Synergy

jXchange reduces the effort and cost of creating interfaces by maintaining a backward compatibility and uses non-proprietary technologies and accepted industry standards to communicate over an extensive web services network. The result is an interoperable atmosphere prime for financial software application working in unison with Jack Henry products.
The information in this manual is instructional in nature and designed for consumers to evaluate the interface using a web-based messaging system for interfacing with Jack Henry & Associates products through jXchange.

**Purpose**

The purpose of this document is to describe the implementation, standards, and operations of the current jXchange release as well as a general overview of the jXchange product.

This document is not designed to be a primer for consuming web services nor a programming teaching tool. JHA can provide direction to materials for training and education if needed.

**Permissions**

The information contained in this document is designed and created for the use of approved parties. To obtain permission for use of this document, contact Jack Henry & Associates at IDGProductAdoption@jackhenry.com.

**General Description**

**What is a Service Gateway?**

The Service Gateway is a set of .NET web services that reside on Windows servers that enable the exchange of information between the various software products. These web services provide the entry point for real-time integration to JHA's SilverLake, Core Director, and CIF 20/20 core banking systems. They also provide access to additional Jack Henry products that act as a jXchange Service Provider (e.g., products such as 4|sight, Synergy, and Argo). The jXchange product is modeled after the SOA architecture.

All the business services exchange their information through the Service Gateway. The information exchange is in the form of XML. Service Gateway provides for the designation of configured providers for all business services that are used by the institution. It allows mapping of all operations for the service providers. It also provides security for the exchange of information by designating authorized users.

**Why was jXchange created?**

jXchange provides a single, real-time interface between JHA products and consumer products. Before jXchange, financial software applications from Jack Henry Banking and consumers required complex interfaces to move data to and from core and complimentary products.

Using web services technology, jXchange employs an XML-based messaging system to accept requests from our complementary products and consumers and return the requested information.

Once the message is submitted, jXchange processes/coordinates the request and forwards it to the appropriate service provider (JHA complementary or core product). A response with the requested information is then returned to the consumer after the message has been processed.

**Technical Recommendations**

**Web Service Description Language (WSDL)**

WSDLs are provided with the Data Contracts (XSDs) at the appropriate time whether it is before the initial development cycle or at the time a new point release becomes available. Once access to jXchange servers and download services have been established, contracts can be generated from the WSDL files that are delivered to you.
**XML/SOAP Toolkit/Parsing**

XML is a self-describing language and parsing, or syntactic analysis of the sequence of tokens, and is a prerequisite for consuming web services. Without the programmatic ability to parse the tokens or elements, applications are not able to keep pace with the technology occupied by jXchange. Processing XML as plaintext hinders the ability to consume the next release of jXchange and its additional services and operations without a development effort.

**Certificate Processing**

SSL certificates are issued by Chosen Security, which is a Certificate Authority, and are provided by JHA. If your system does not trust this Certificate Authority by default, you may obtain the root certificate from [www.TCTrustCenter.de](http://www.TCTrustCenter.de).

**jXchange Release Cycle**

jXchange Web Services are upgraded to the current release at least once annually.

**Multiple Point Releases**

jXchange can have multiple releases annually. Any point release may represent either minor or major updates or upgrades.

**Compatibility**

jXchange end-to-end services are designed to allow both forward and backward compatibility from release to release. For example, jXchange may be at the 2009.1 release while SilverLake is at the 2010 release, or jXchange may be at the 2010 release while SilverLake is at the 2009 release. Not all potential functionality is available, but current functionality is available and unhindered.

**jXchange Release Classification Types**

**Transparent**

jXchange server updates are transparent to the consumer of the web services. After a scheduled server update, connectivity can be resumed without any changes to the application or correlating configuration files.

**Consumer Configurable**

Changes that can be made through configuration files to resume web service connectivity after an update has been made to the jXchange server.

Example: URL Address endpoint extension change from .ashx to .svc.

**Non-backward Compatible**

Release is not compatible with previous releases. The application that consumes the web service must be modified to match the new contracts to resume connectivity. Support for the release before the breaking change has a sunset date and is not enhanced with new operations or services.
Consumer Relations

Conseller Relations with JHA

Products cannot have the term or branding of jXchange in the user interface. jXchange is copyrighted and must not be advertised to the end user as the product itself nor give the impression that the application using the interface is jXchange.

All consumers are required to participate in the Integration Development Group Adoption team adoption process. If it is not a JHA product, membership in the Vendor Integration Program is required. Contact the IDG Adoption team by emailing IDGProductAdoption@jackhenry.com.

Standards

The standards jXchange follows are:

- HTTPS: http://en.wikipedia.org/wiki/HTTP_Secure
- SOAP: http://www.w3.org/TR/soap/
- W3C: http://www.w3.org/
- WSDL: http://www.w3.org/TR/wsdl
- XML: http://www.w3.org/XML/

Web Service Standards

The following Web Services standards are being used in the design and construction of jXchange. Messaging with jXchange from third party vendors must comply with the standards for effective communication:

- SOAP 1.1 for standard and error messaging.
- XML Schema 2001 (W3C).
- WS-I Basic Profile 1.0 for basic architecture built against compliance with the standard. The system follows WSI usage scenarios and synchronous messaging scenario.
- WS-I BP 1.0 for security model in compliance with WS-I Basic Security Profile 1.0.
- OASIS WS-Security 1.0 specification using a hybrid solution of transport level security (SSL) and WS-Security in compliance with WS-I Usage Scenarios 1.0.
- WS-I Usage Scenarios 1.01.

Best Practices

XML Toolkit

Highly recommended due to ease of limitation and modification.

Unit of Work Connections

When requests can be grouped, group them on a single, established session. For example, when making multiple jXchange calls for a customer, open the connection, make all the requests, and then close the connection. This improves performance by reducing the number of times you must spend time establishing a TCP and SSL session to the jXchange server before submitting a request.
Multi-Thread

Use this option whenever possible to increase your performance gains.

Error Message Handling

There are two types of error messages:

- System Fault: Designates a failure of one or more parts of the system.
- Request Faults: Designates a failure regarding to the request for data or information or the related response.

Request Fault Error messages are returned as a SOAP fault. The detail section contains a standard JHA fault structure, which is described in the WSDL file for a service. The structure contains an error code and a description of the specific request fault.

Further information about SOAP protocol can be obtained at the W3C link: http://www.w3.org/TR/2003/REC-soap12-part1-20030624/.

Create a Proxy using the SVCUTIL.EXE

Here is an example of creating a .NET web service proxy using SvcUtil.exe. This utility is included with the Microsoft® Windows SDK. http://msdn.microsoft.com/en-us/windowsserver/bb980924.aspx

The assumption is that the XSDs and WSDLs are in the same directory as the SvcUtil.exe, but paths can be used if they are not in the same directory.

The text Namespace.Name should be replaced with the namespace used in the related project file.

- C# (the default): svcutil/n:*,"[namespace]" "[path to wsdl]" "[path to xsd]" /mc /importXmlTypes "[path to this xsd]oasis-200401-wss-wssecurity-secext-1.0.xsd" "[path to this xsd]oasis-200401-wss-wssecurity-utility-1.0.xsd" /serializer:XmlSerializer /useSerializerForFaults /d:[location of class and config file creation]

Additional information can be found at http://msdn.microsoft.com/en-us/library/aa347733(v=vs.110).aspx

Sample .NET App.config File

- The following is an example of an app.config file used with .NET projects. In this example, the URL Endpoint Address is in the config file. The security mode attribute TransportWithMessageCredential must be used and indicates that the connection is over Secured Socket Layer (SSL).

  This is for a .NET developer only. JHA is a .NET developer.
XML Request/Response Capture

One of the most important parts of jXchange communications are the XML request and response messages. The Support Department requests these messages from you to help identify several aspects, which include the raw request submitted to our services for reproduction of the issue or identification of improper formed messages. When the Support Department receives wrapping errors or application side errors that are not the real error returned from jXchange, it causes delays in troubleshooting the issue. As a result, it is highly recommended that you code your software to be able to capture both the raw request and response XML that is transmitted and received from our services.

For information on performing similar function in .NET, you may review this link: http://bit.ly/WSF8YZ. We do not have any examples for languages other than C#.

jXchange Log Tracking Id

The jXLogTrackingId is a Globally Unique Identifier (GUID) that can be specified in the request to our services, carried through all components, and returned on the response.

The jXLogTrackingId is recommended to be specified by the consumer so that the id can be associated to the specific request from the consumers side and match to any log information that may be available. If one is not specified, jXchange returns a log tracking id for you. During troubleshooting, knowing the jXLogTrackingId up front allows the Support Department to isolate our logs to that specific request from receipt to response, including access to the cores. The jXLogTrackingId is the single unifying ID for a particular message into our system that is carried to all components in the jXchange system.

jXchange Development Environment

Jack Henry offers a jXchange development environment for consumers to access during their development cycle. It is commonly referred to as the DMZ, demilitarized zone, due to the manner in which it is deployed in the infrastructure.

This environment has a strict URL Path Policy applied, in that you must hit the jXchange DMZ Services on a URL Path that is valid for jXchange Web Services or your communication attempt is discarded. The URL paths are given to you in the credentials email after access is granted.

Access to the DMZ requires:

- Signed Confidential and Certification Agreement
- Signed Third Party Access Agreement
- Vendor Integration Program participation
- Certification Fee
This environment provides all of the supported business providers to which the Service Gateway integrates. This environment is always kept at the current GA release of jXchange. Previous releases are not supported or available in this environment. Access to this environment allows a close working relationship with the jXchange support staff and expediting the process and development of the third party products.

Consumers are required to test all operations applicable to their specific application to verify that the XSD contract schema requirements are being met.

Maintenance periods on this environment are scheduled every day from 12:00 a.m. Central Standard Time through 2:30 a.m. Central Standard Time. System maintenance may not be performed every day, but when it is needed jXchange Support attempts to isolate it to these hours. Maintenance can include but is not limited to:

- Server patching and upgrades.
- Hardware upgrades.
- Core processing hours for test banks.
- Core component upgrades.

**jXchange Security**

**Authentication**

Authentication in the jXchange product family now relies on Active Directory as our account store. It allows a single repository for accounts that falls into the financial institution's existing infrastructure, allowing easier, more familiar management of accounts by bank personnel. It allows the bank to enforce their policies on jXchange product accounts to satisfy their internal regulatory requirements.

The drawback to this method is that the financial institution can enact changes on product accounts that can affect the availability of services to consumers. For instance, if a bank implements a new password policy on accounts and the jXchange product accounts are included, then accounts could have passwords or complexity requirements change that a consumer is unaware of. As a result, that consumer can lose access to services until the issue is resolved. However, with education and understanding of the impact on jXchange services, these risks can be mitigated.

**General Description**

The security model for the Client - Third Party Gateway boundary provides authentication and authorization with the WS-Security specification's Username Token Profile. The cryptographic protocol Secure Socket Layer (SSL) is used for transport-level security which provides both confidentiality and integrity. Messages destined for the Third Party Gateway Web Service must include at least two elements in the WS-Security SOAP header: one element for the user name and the second element for the password.
SSL is required using 128-bit encryption for server authentication. Using these security principles, jXchange provides a secure boundary between the client and Third Party Gateway.

**Authorization**

The jXchange system restricts access to its interfaces using a Role Based Access Control (RBAC) system. A third party system's identity must have been associated with the required role for a given operation on a service.

The institution, a JHA client, manages the association of roles to identities. A given identity can be associated with more than one role. An operation on a service may restrict access by requiring a role, may deny access to callers that are a member of a particular role, or may not restrict access to the operation at all.

The jXchange system restricts access to operations using RBAC. Bank association also restricts access. The caller must be associated with the bank the target service has been assigned to. The institution also controls the assignments of callers to banks.

**Certificates**

**Symantec®**

Certificates are issued by Symantec®, the company that purchased TC TrustCenter. Jack Henry has a business relationship with Symantec® and provides highly trusted and secure certificates.

jXchange only utilizes server-based certificates. It does not require client certificate processing. The publicly trusted root of the SSL certificate must be in the application's proper certificate store for processing.

**Appendix A**

Supported operations for each provider are in a separate document.

**Glossary**

**ABA**

An identification number consisting of a two-part code assigned to banks and savings associations. The first part shows the location and the second identifies the bank.

**Authentication**

The process of identifying an individual based on a user name and password. In security systems, authentication is distinct from authorization, which is the process of giving individuals access to system objects based on their identity. Authentication ensures that the individual is who he or she claims to be, but says nothing about the access rights of the individual. Windows Authentication is preferred, but SQL Authentication is available for use.

**Authorization**

The process of granting or denying access to a network resource. Most computer security systems are based on a two-step process. The first stage is authentication, which ensures that a user is who he or she
claims to be. The second stage is authorization, which allows the user access to various resources based on the user's identity.

**Business Service Provider**
A Business Services Provider (BSP) is an application service provider that focuses on providing and hosting applications related exclusively to business functions.

**Consumer**
A network application that uses internet protocols to access information and functionality provided by a Service Provider.

**DMZ**
Demilitarized zone. A computer or small subnetwork that sits between a trusted internal network, such as a corporate private LAN, and an untrusted external network, such as the public internet. Typically, the DMZ contains devices accessible to internet traffic, such as web (HTTP) servers, FTP servers, SMTP (email) servers, and DNS servers. The term comes from military use, meaning a buffer area between two enemies.

**HTTPS**
Hypertext Transfer Protocol Secure, see SSL for more information.

**NLB**
Network Load Balancing (NLB) is a clustering technology offered by Microsoft® as part of all Windows® 2000 Server and Windows Server® 2003 family operating systems. NLB uses a distributed algorithm to load balance network traffic across several hosts, helping to enhance the scalability and availability of IP-based services such as web, Virtual Private Networking, streaming media, terminal services, and proxy. It also provides high availability by detecting host failures and automatically redistributing traffic to operational hosts.

**SOA**
Abbreviated SOA, an application architecture in which all functions, or services, are defined using a description language and have invokable interfaces that are called to perform business processes. Each interaction is independent every other interaction and the interconnect protocols of the communicating devices (i.e., the infrastructure components that determine the communication system do not affect the interfaces). Because interfaces are platform-independent, a client from any device, using any operating system, in any language can use the service.

Though built on similar principles, SOA is not the same as web services, which indicate a collection of technologies, such as SOAP and XML. SOA is more than a set of technologies and runs independent of any specific technologies.

**SOAP**
Simple Object Access Protocol. It is a lightweight XML-based messaging protocol used to encode the information in web service request and response messages before sending them over a network. SOAP messages are independent of any operating system or protocol and may be transported using various internet protocols, including SMTP, MIME, and HTTP.

**SSL**
Secure Sockets Layer. It is a protocol developed by Netscape for transmitting private documents via the internet. SSL uses a cryptographic system that uses two keys to encrypt data, a public key known to everyone and a private or secret key known only to the recipient of the message. Both Netscape Navigator and Internet Explorer® support SSL, and many websites use the protocol to obtain confidential user information, such as credit card numbers. By convention, URLs that require an SSL connection start with https instead of http.

**W3C**
World Wide Web Consortium. It is an international consortium of companies involved with the internet and the web. The W3C was founded in 1994 by Tim Berners-Lee, the original design of the World Wide Web.
The organization's purpose is to develop open standards so that the web evolves in a single direction rather than being splintered among competing factions.

**XML**

Extensible Markup Language. It is a specification developed by the W3C. XML is a pared-down version of SGML, designed especially for web documents. It allows designers to create their own customized tags, enabling the definition, transmission, validation, and interpretation of data between applications and between organizations.