



Blue Cross & Blue Shield of Rhode Island

BCBSRI 27x Real-Time API Connection Procedures

Version 3.3
Status: Published

October 29, 2014

This document may be revised and republished if and when Blue Cross & Blue Shield of Rhode Island makes improvements and/or changes to any referenced product, process or program.

The information and contents of this document and any notes or handouts, if any (together “document”), contain confidential and proprietary information, and are not to be disseminated, reproduced, printed, translated or transmitted in any form, in whole or in part, without the prior written consent or express permission of Blue Cross & Blue Shield of Rhode Island. Use and distribution limited solely to authorized personnel.

2013 © Blue Cross & Blue Shield of Rhode Island

All Rights Reserved.

PREFACE

The *BCBSRI 27x Real-Time API* document supplements the *BCBSRI Companion Guides*. Its purpose is to provide trading partners with instructions on transmitting electronic data to Blue Cross & Blue Shield of Rhode Island (hereinafter "BCBSRI").

DISCLAIMER

This document is considered a living document, and as such, the information provided herein will be subject to change after July 1, 2013 in the event that BCBSRI revises its policies or HIPAA Transactions and Code Sets law is updated or amended.

Table of Contents

1.0	Introduction	1
2.0	Scope	1
3.0	Contact Information	1
4.0	Using the Web Service	1
4.1	Logging On	1
4.2	Security	2
4.3	Web Service Description (WSDL)	2
4.4	Error Conditions	5
4.5	Sample Input XML	5
4.6	Test Mode	6
5.0	Document Version Control	6

1.0 Introduction

This document provides instructions for connecting to and using the 27x Real-Time API, a Web-based application utilizing Internet browser technology (IP Protocol). It allows the programmatic submission and receipt of individual HIPAA transaction responses. Use this document in conjunction with the *BCBSRI EDI Companion Guides* for each transaction.

2.0 Scope

The procedures in *BCBSRI 27x Real-Time API* apply to all of the following transactions:

270/271 Health Care Eligibility Benefit Inquiry and Response
276/277 Health Care Claim Status Request and Response
278 Health Care Services Review – Request for Review and Response

3.0 Contact Information

BCBSRI will work closely with its trading partners to establish effective communication protocols and to resolve any connectivity issues that may arise regarding the exchange of HIPAA-related electronic transactions.

The following contact information is provided to assist in the process of implementing all transactions:

HIPAA EDI Testing Support business hours are Monday through Friday,
8:00 AM to 4:30 PM.

Email Address: HIPAA.EDI.Support@bcbsri.org

For Production:

**Call the Information Technology Call Center, which supports BCBSRI, at
401- 751-1673 or 1-855-721-4211.**

4.0 Using the Web Service

4.1 Logging On

1. Using your Internet browser, enter the Web address provided to you upon completion of the registration process. This address will connect to the BCBSRI 27x Real-Time API.
2. The Web Service has been implemented in SOAP, Simple Object Access Protocol, as a single conduit for all transactions and as such there is only one operation in the API, **Real-Time X12 Transaction**. There is a parameter in the web service operation to denote the transaction requested (270, 276, and 278). For the 5010 transactions, the parameter names are 270-5010, 276-5010 and 278-5010. You may also use a T at end of parameter names to Test connectivity only.

3. To eliminate the need to use CDATA sections in order for the X12 not to trip up XML parsers, binary hexadecimal encoding is applied to the X12 content. This allows the X12 to be a defined parameter to the web service. All development platforms should allow for the simple conversion of text to binary hex. This does roughly double the size of the text but these transactions are small since they are Real-Time which only allows one inquiry per transaction (one ST segment).

NOTE: Both the User ID and password are case sensitive. An alpha-leading User ID and a randomly assigned password will be provided by BCBSRI. The User ID prefixes will change from "T" (for Test) to "P" (for Production). The unique password assigned for testing will be replaced by a new production password.

4.2 Security

Security is accomplished via the use of SSL and an authentication to the service. There are two parameters, username and password to pass the credentials to the service. Since all submittals are direct to BCBSRI.COM servers and no intermediaries are used SSL will accomplish confidentiality of all data included.

4.3 Web Service Description (WSDL)

The Web Service Operation is **Real-TimeX12Transaction** and this takes 4 parameters as input.

1. Transaction – the HIPAA transaction code (i.e. 270-5010, 276-5010, 278-5010)
2. UserName - user ID assigned to you
3. Password - password assigned to you
4. InputX12Transaction - X12 Real-Time HIPAA transaction encoded in hexadecimal binary form

It is shown below:

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions targetNamespace="http://webservice.realtime"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:intf="http://webservice.realtime"
xmlns:impl="http://webservice.realtime"
xmlns:tns2="http://fault.webservice.realtime">
  <wsdl:types>
    <schema elementFormDefault="qualified"
targetNamespace="http://webservice.realtime"
xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:impl="http://webservice.realtime"
xmlns:intf="http://webservice.realtime"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
      <complexType name="RealTimeX12Request">
        <sequence>
          <element name="userName" nillable="true" type="xsd:string"/>
          <element name="password" nillable="true" type="xsd:string"/>
          <element name="transaction" nillable="true" type="xsd:string"/>
          <element name="inputX12Transaction" type="xsd:hexBinary"/>
        </sequence>
      </complexType>
    <element name="realTimeX12Transaction">
      <complexType>
        <sequence>
```

```
        <element name="parameters" nillable="true"
type="impl:RealTimeX12Request"/>
    </sequence>
</complexType>
</element>
<complexType name="RealTimeX12Response">
    <sequence>
        <element name="outputX12Transaction" type="xsd:hexBinary"/>
        <element name="status" nillable="true" type="xsd:string"/>
    </sequence>
</complexType>
<element name="realTimeX12TransactionResponse">
    <complexType>
        <sequence>
            <element name="realTimeX12TransactionReturn" nillable="true"
type="impl:RealTimeX12Response"/>
        </sequence>
    </complexType>
</element>
</schema>
<schema elementFormDefault="qualified"
targetNamespace="http://fault.webservice.realtime"
xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:impl="http://webservice.realtime"
xmlns:intf="http://webservice.realtime"
xmlns:wSDL="http://schemas.xmlsoap.org/wSDL/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <complexType name="AuthenticationFailedFault">
        <complexContent>
            <extension base="tns2:RealTimeFault">
                <sequence/>
            </extension>
        </complexContent>
    </complexType>
    <complexType name="RealTimeFault">
        <sequence>
            <element name="message" nillable="true" type="xsd:string"/>
        </sequence>
    </complexType>
    <element name="AuthenticationFailedFault" nillable="true"
type="tns2:AuthenticationFailedFault"/>
    <complexType name="SystemUnavailableFault">
        <complexContent>
            <extension base="tns2:RealTimeFault">
                <sequence/>
            </extension>
        </complexContent>
    </complexType>
    <element name="SystemUnavailableFault" nillable="true"
type="tns2:SystemUnavailableFault"/>
</schema>
</wSDL:types>
<wSDL:message name="realTimeX12TransactionRequest">
    <wSDL:part name="parameters" element="intf:realTimeX12Transaction"/>
</wSDL:message>
<wSDL:message name="SystemUnavailableFault">
    <wSDL:part name="fault" element="tns2:SystemUnavailableFault"/>
</wSDL:message>
<wSDL:message name="realTimeX12TransactionResponse">
```

```
<wsdl:part name="parameters"
element="intf:realTimeX12TransactionResponse"/>
</wsdl:message>
<wsdl:message name="AuthenticationFailedFault">
  <wsdl:part name="fault" element="tns2:AuthenticationFailedFault"/>
</wsdl:message>
<wsdl:portType name="X12RequestHandlerService">
  <wsdl:operation name="realTimeX12Transaction">
    <wsdl:input name="realTimeX12TransactionRequest"
message="intf:realTimeX12TransactionRequest"/>
    <wsdl:output name="realTimeX12TransactionResponse"
message="intf:realTimeX12TransactionResponse"/>
    <wsdl:fault name="AuthenticationFailedFault"
message="intf:AuthenticationFailedFault"/>
    <wsdl:fault name="SystemUnavailableFault"
message="intf:SystemUnavailableFault"/>
  </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="X12RequestHandlerServiceSoapBinding"
type="intf:X12RequestHandlerService">
  <wsdlsoap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="realTimeX12Transaction">
    <wsdlsoap:operation soapAction=""/>
    <wsdl:input name="realTimeX12TransactionRequest">
      <wsdlsoap:body use="literal"/>
    </wsdl:input>
    <wsdl:output name="realTimeX12TransactionResponse">
      <wsdlsoap:body use="literal"/>
    </wsdl:output>
    <wsdl:fault name="AuthenticationFailedFault">
      <wsdlsoap:fault name="AuthenticationFailedFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="SystemUnavailableFault">
      <wsdlsoap:fault name="SystemUnavailableFault" use="literal"/>
    </wsdl:fault>
  </wsdl:operation>
</wsdl:binding>
<wsdl:service name="X12RequestHandlerServiceService">
  <wsdl:port name="X12RequestHandlerService"
binding="intf:X12RequestHandlerServiceSoapBinding">
    <wsdlsoap:address
location="http://localhost:9081/RealTimeServicesWeb/services/X12RequestHandle
rService"/>
  </wsdl:port>
</wsdl:service>
</wsdl:definitions>
```


