

---

Article

[Zhong Li](#) · Sep 25, 2017 5m read

# HealthConnect (Ensemble) Integration with PKB Service

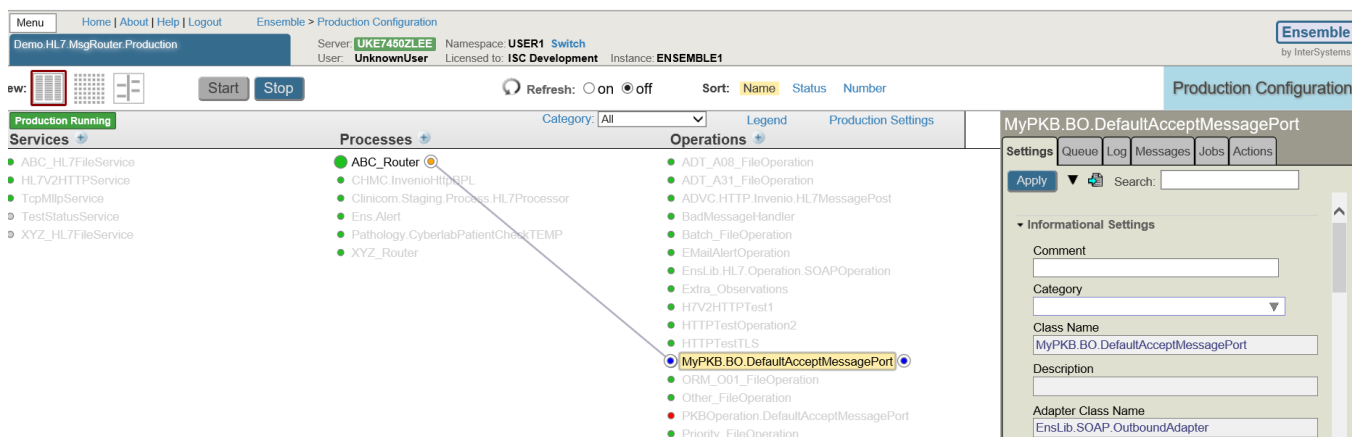
## 1. Scope and Objective:

Recently we supported a few NHS cases that required TIE (Trust Integration Engine) integration with the PKB service. Hence this article is meant to be a 10-minute quick guide to describe a demo solution (simple configurations and end-2-end implementation steps) for Health Connect (Ensemble) Integration with PKB (Patient-Knows-Best) service.

[PKB service](#) is a patient centric information service. It has a set of well defined API [interfaces \(documentation here\)](#) based on HL7 V2 over HTTP/SOAP, REST and FHIR. This article only touches its "HL7 V2 over HTTP/SOAP" service interface.

## 2. Demo Solution:

The following end-2-end implementation steps are done in a testing Ensemble instance. We will use Ensemble Studio's "[SOAP Wizard](#)" tool to create a new PKB SOAP Client (Business Operation in Ensemble terms) on an existing HL7 V2 demo production, then we will simply link an existing BP(Business Process) to this PKB BO via a standard EnsLib.HL7.Message type request. Here below is a screen capture highlighting the related Ensemble ConfigItems:



### 3. Detailed Implementation Steps

#### 3.1 Prerequisites:

##### 3.1.1 Create a basic SSL/TLS Configuration called e.g. "PKB\_Client":

Menu

Home | About | Help | Logout

System > Security Management > SSL/TLS Configurations > Edit SSL/TLS Configuration

Edit: PKB\_Client

Server: UKE7450ZLEE Namespace: %SYS  
User: UnknownUser Licensed to: ISC Development Instance: ENSEMBLE1

Save

Cancel

Test

Use the form below to edit an existing SSL/TLS configuration:

Configuration Name

PKB\_Client

Required.

Description

SSL client for PKB BO

Enabled

☒

Type

☒ Client ☐ Server

Server certificate verification

☒ None ☐ Require

File containing trusted Certificate Authority certificate(s)

Browse...

File containing Certificate Revocation List

Browse...

This client's credentials

Note: Only necessary if this client will be asked to authenticate itself to servers.

File containing this client's certificate

Browse...

File containing associated private key

Browse...

Private key type ☒ RSA ☐ DSA

Password: ☐ Enter new password ☐ Clear password ☒ Leave as is

Cryptographic settings

Protocols ☐ SSLv2 ☐ SSLv3 ☒ TLSv1.0 ☒ TLSv1.1 ☒ TLSv1.2

Enabled ciphersuites ALL:!aNULL:!eNULL:!EXP:!SSLv2

##### 3.1.2 To create a SOAP Credential in Ensemble

Note: You can contact PKB Service to get a SOAP credential (plain-text username/password) to its Test Environment, by following the [documentation](#) here.

Here below is a sample configuration of the Test Credential: (by clicking in Management Portal -> Ensemble -> Configure -> Credentials )

**Credentials Viewer**

**ID**  
org\_isc  
Identifier for this item

**User Name**  
org\_isc  
User name used to connect to external system

**Password**  
Password hidden. Click to enter new password  
Password used to connect to external system

☐ Show typing

**Business Partner**  
[Empty dropdown menu]  
Name of Business Partner Profile associated with this item

[Business Partners Configuration Page](#)

**New** **Save** **Remove**


### 3.2 Use "SOAP Wizard" to create PKB SOAP Client BO Package:


#### 3.2.1 Start "SOAP Wizard" from Ensemble Studio:

Start "Studio"; go to right namespace such as "TEST1"; Click menu "Tools" -> "Add-Ins" -> "SOAP Wizard"

#### 3.2.2 Steps through "SOAP Wizard" pages for PKB Service's Endpoint:

Note: You can see the PKB service's Test Endpoint in its [online documentation here.](#)

 SOAP Wizard

 Studio Template  
**SOAP Wizard**

User: UnknownUser  
Namespace: ENSDEMO

The SOAP Wizard reads a WSDL (*Web Services Description Language*) document and creates one or more SOAP client or service classes. Each SOAP Client class contains one or more methods that, when invoked, remotely call the corresponding Web Method of the Web Service. Each SOAP Service class contains one or more methods that may be remotely invoked.

The Wizard will also create any additional classes needed to represent any complex types (objects) used by the Web Service. These additional classes are placed within a package named after the Service Name.

To start, enter the location of the WSDL document that describes the Web Service and then press **Next**.

---

**Select a WSDL file or URL: \***

☒ URL ☐ FILE

**Enter a WSDL URL: \***

Required. Enter address of a WSDL document.

SSL Configuration:

SSL Configuration to use when connecting to the server that is hosting this WSDL.

Recently Used URLs:

SOAP Wizard

Studio Template

SOAP Wizard

User: UnknownUser  
Namespace: ENSDEMO

Step 2 - The WSDL you have selected is displayed below. If this is correct, select options as wanted and press **Next**.

Options to control class generation and compilation: \*

☒ Create Client for Web Service  
☐ Create Web Service  
☒ Compile generated classes    Compile flags:

Class Type: ☐ Persistent  
☐ Persistent using one-many relationships  
☐ Persistent using indexed one-many relationships  
☐ Persistent using parent-child relationships  
☒ Serial

☒ Create Business Operation  
Business Operation Package   
Request Object Package   
Response Object Package  x

Proxy Class Package   
SSL Configuration name

WSDL Document: [https://sandbox.patientsknowbest.com/hl7\\_wsdl.xml](https://sandbox.patientsknowbest.com/hl7_wsdl.xml)

Expand All

```
<?xml version="1.0" ?>
<definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tns="http://wsssl.hl7api.patientsknowbest.com/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://schemas.xmlsoap.org/wsdl/" targetNamespace="http://wsssl.hl7api.patientsknowbest.com/"
name="DefaultAcceptMessageService">
  <types>
    <xsd:schema>
      <xsd:import namespace="http://wsssl.hl7api.patientsknowbest.com/"
schemaLocation="https://sandbox.patientsknowbest.com/hl7_xsd.xml"></xsd:import>
    </xsd:schema>
  </types>
  <message name="acceptMessage">
    <part name="parameters" element="tns:acceptMessage"></part>
  </message>
  <message name="acceptMessageResponse">
    <part name="parameters" element="tns:acceptMessageResponse"></part>
  </message>
  <portType name="DefaultAcceptMessage">
    <operation name="acceptMessage">
      <input message="tns:acceptMessage"></input>
      <output message="tns:acceptMessageResponse"></output>
    </operation>
  </portType>
  <binding name="DefaultAcceptMessagePortBinding" type="tns:DefaultAcceptMessage">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"></soap:binding>
    <operation name="acceptMessage">
      <soap:operation soapAction=""></soap:operation>
      <input>
        <soap:body use="literal"></soap:body>
      </input>
      <output>
        <soap:body use="literal"></soap:body>
      </output>
    </operation>
  </binding>
  <service name="DefaultAcceptMessageService">
    <port name="DefaultAcceptMessagePort" binding="tns:DefaultAcceptMessagePortBinding">
      <soap:address location="https://sandbox.patientsknowbest.com:7443/services/hl7"></soap:address>
    </port>
  </service>
</definitions>
```

Back

Next

Finish

Cancel

Help

SOAP Wizard

Studio Template

SOAP Wizard

User: UnknownUser  
Namespace: ENSDEMO

Step 3 - The SOAP Wizard has determined that that your WSDL uses the following XML namespaces.

The Wizard will generate all the classes for an XML namespace within the same class package. You can specify the package name used for each XML namespace using the fields below.  
Afterwards, press **Next**.

☒ Add NAMESPACE class parameter

☐ Use unwrapped message format for document style web methods

☐ Do not create array properties

☐ Generate XMLNIL property parameter for nillable elements

☐ Generate XMLNILNOOBJECT property parameter for nillable elements

☐ Set XMLSEQUENCE parameter to 0

☐ Generate XMLIGNORENULL parameter set to 1

☐ Use Streams for Binary

Specify SECURITYIN class parameter 

ALLOW

	Package Name
Web Client Package	<div>MyPKB.Proxy</div>
Configuration Sub-Package. If not specified, append 'Config' to class name in same package.	<div></div>

XML Namespace	Package Name
<a href="http://schemas.xmlsoap.org/wsdl/soap/">http://schemas.xmlsoap.org/wsdl/soap/</a>	<div>MyPKB.Proxy.soap</div>
<a href="http://wsssl.hl7api.patientsknowbest.com/">http://wsssl.hl7api.patientsknowbest.com/</a>	<div>MyPKB.Proxy.tns</div>
<a href="http://schemas.xmlsoap.org/wsdl/">http://schemas.xmlsoap.org/wsdl/</a>	<div>MyPKB.Proxy.wsdl</div>

Back

Next

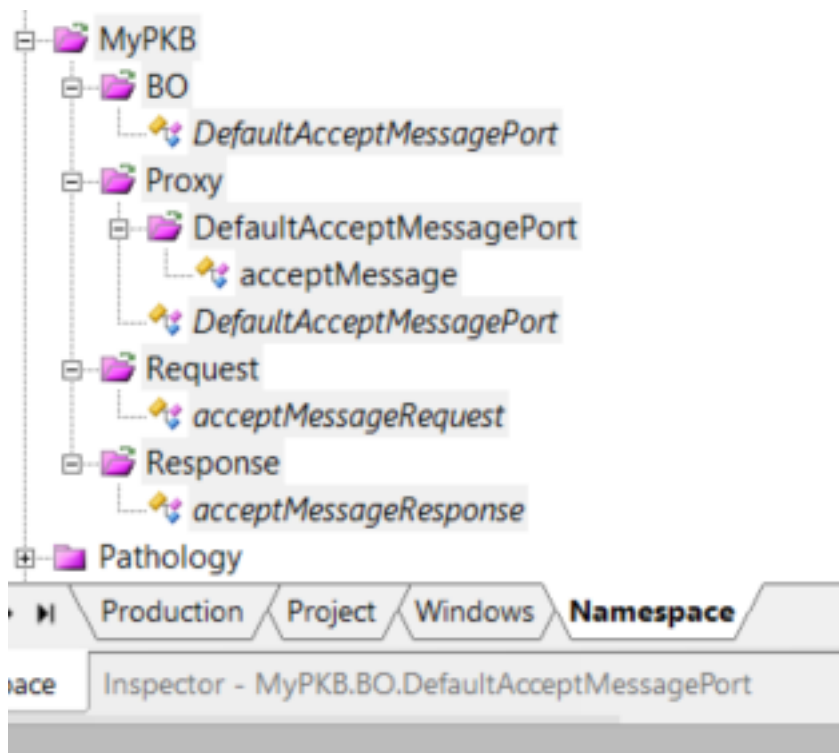
Finish

Cancel

Help

3.2.3 Inspect the generate "MyPKB" SOAP Client Package in "Studio":

Now within the Studio, you can see the following classes have been automatically generated by "SOAP Wizard":



Note: For detailed introductions, click on the product documentation ["how Ensemble SOAP Wizard works"](#) here.

### 3.3 Customise the MyPKB.BO class slightly to accept "EnsLib.HL7.Message" request from BP:

1. Open " [MyPKB.BO.DefaultAcceptMessagePort.cls](#)" in Studio
2. Add in "[EnsLib.HL7.Message](#)" in its "MessageMap"
3. Define its handler as e.g. "HandleHL7Messages" Method.
4. Recompile the MyPKB package.

So the resulted source code of this class file could be something like this sample below:

```
Class MyPKB.BO.DefaultAcceptMessagePort Extends Ens.BusinessOperation [ ProcedureBlock ]
{
    Parameter ADAPTER = "EnsLib.SOAP.OutboundAdapter";

    Method acceptMessage(pRequest As
MyPKB.Request.acceptMessageRequest, Output pResponse As
MyPKB.Response.acceptMessageResponse) As %Library.Status
    {
        Set ..Adapter.WebServiceClientClass = "MyPKB.Proxy.DefaultAcceptMessagePort"
        Set tSC = ..Adapter.InvokeMethod("acceptMessage",.return,pRequest.arg0) Quit:$$$ISERR(tSC) tSC
        Set tSC = pRequest.NewResponse(.pResponse) Quit:$$$ISERR(tSC) tSC
        Set pResponse.return=$get(return)
        Quit $$$OK
    }
}
```

```

Method HandleHL7Messages(pRequest As EnsLib.HL7.Message, Output pResponse As
MyPKB.Response.acceptMessageResponse) As %Library.Status
{
  //From PKB
  Set PKBReq = ##class(MyPKB.Request.acceptMessageRequest).%New()
  Set PKBReq.arg0 = pRequest.RawContent

  Set ..Adapter.WebServiceClientClass = "MyPKB.Proxy.DefaultAcceptMessagePort"
  Set tSC = ..Adapter.InvokeMethod("acceptMessage",.return,PKBReq.arg0) Quit:$$$ISERR(tSC) tSC
  Set tSC = PKBReq.NewResponse(pResponse) Quit:$$$ISERR(tSC) tSC
  Set pResponse.return=$get(return)
  Quit $$$OK
}
XData MessageMap
{
  <MapItems>
    <MapItem MessageType="MyPKB.Request.acceptMessageRequest">
      <Method>acceptMessage</Method>
    </MapItem>
    <MapItem MessageType="EnsLib.HL7.Message">
      <Method>HandleHL7Messages</Method>
    </MapItem>
  </MapItems>
}

```

### 3.4 Customise property "return" within "MyPKB.Response.acceptMessageResponse" class

So the "return" property can accept a long string returned from PKB service of more than 50 chars:

```
Property return As %String(MAXLEN = 5000);
```


Recompile the class again.

### 3.5 Add the newly created MyPKB SOAP Client into a demo HL7 Production

1. Create or re-use a demo HL7 Production.
2. Add the newly created "MyPKB.BO.DefaultAcceptMessagePort" BO class into the production (click "+" beside "Operations")
3. Configure it as follows, for example:

### MyPKB.BO.DefaultAcceptMessagePort

**Settings** Queue Log Messages Jobs Actions

Apply ▼  Search:

**▼ Informational Settings**

**Comment**


**Category**

**Class Name**

**Description**

**Adapter Class Name**

**Adapter Description**


**Business Partner**  
 


**▼ Basic Settings**

**Enabled**  
☒

**Web Service URL**

**Web Service Client Class**

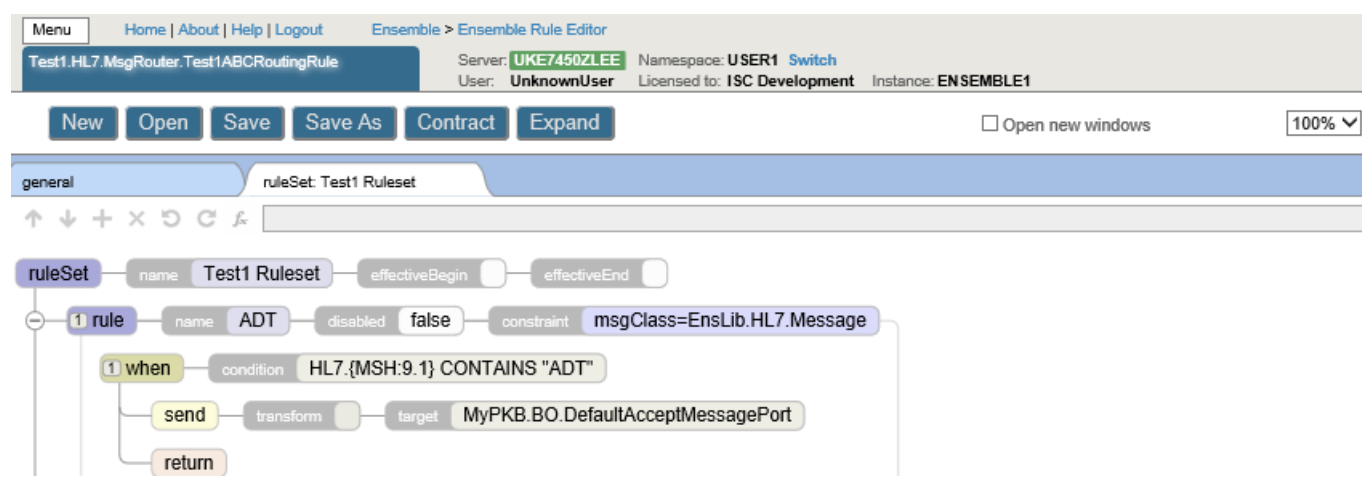
**SOAP Credentials**  
 

**Credentials**  
 

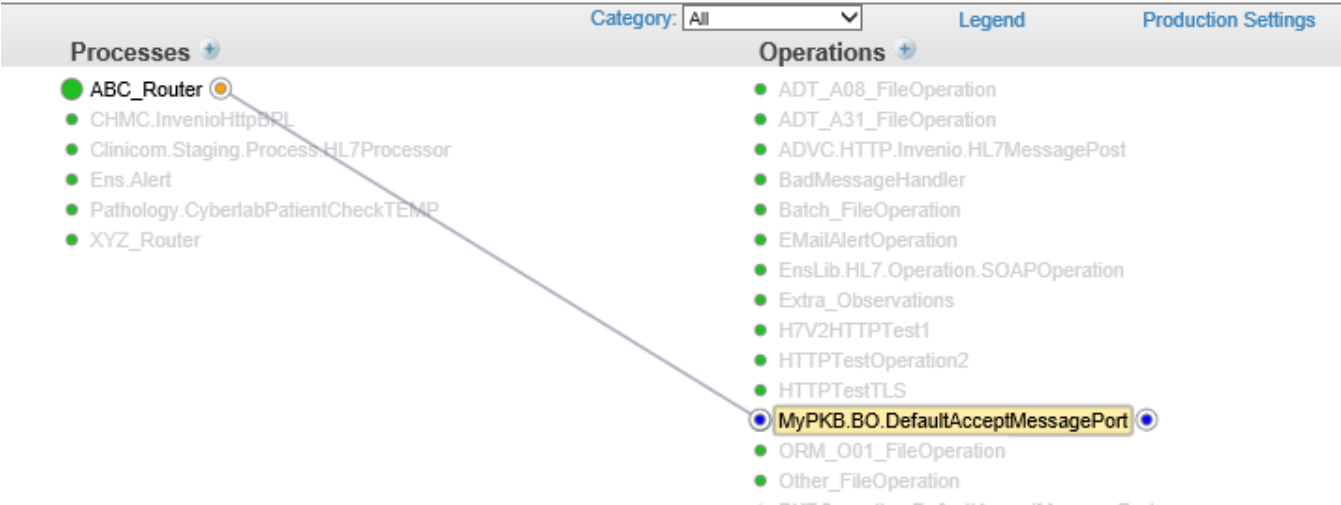
The screenshot shows the configuration window for 'MyPKB.BO.DefaultAcceptMessagePort'. The window has tabs for Settings, Queue, Log, Messages, Jobs, and Actions. The 'Settings' tab is active, showing various configuration options:

- Connection Settings:**
  - SSL Configuration: PKB\_Client
  - SSL Check Server Identity: ☒
  - Connect Timeout: 15
  - Response Timeout: 45
- Proxy Settings:**
  - Proxy Server: [Empty text box]
  - Proxy Port: 80
  - Proxy HTTPS: ☐
  - ProxyHttpTunnel: ☐
  - ProxyHttpSSLConnect: ☐
- Additional Settings:**
  - Schedule: [Empty dropdown menu]
  - Pool Size: 1
  - Reply Code Actions: [Empty text box]
  - Retry Interval: 5
  - Failure Timeout: 15
  - Throttle Delay: 0
- Alerting Control:** [Empty text box]
- Development and Debugging:** [Empty text box]

3.6 Optionally, we can link a HL7 V2 BP to this PKB Client BO via "EnsLib.HL7.Message" request:



So in the demo production, it will be shown as below, for example:




#### 4. Testing

Now let's use the Ensemble Production's "Test" facility to do a quick test on the demo "ABCRouter":

ABC\_Router

Clinicom Staging Process.HL7Processor

 **ABC Router**  
Production Demo.HL7.MsgRouter.Production

Request Type: EnsLib.HL7.Message

Request Details

HL7 document content:

```
MSH|^~\&|IHCS-PAS|NHS TIE1|||20160120083519||ADT^A01|252728|P|2.4|||AL
EVN|A01|20160120083519
PID||Q1ob|1007700^HOS~0501230^KOR|Adhoc16204^Test^REG^MR||19660101|M||1 THE STREET^NEW
TOWN^TORQUAY^TQ1 1XX^P~1 THE STREET^NEW TOWN^TORQUAY^TQ1 1XX^M|01234
122122~07654221221|||U|C1^Clinician|||A^WHITE - BRITISH|||N|03
PD1||STOKENHAM SUR^L83148^Q2483~THE SURGERY^L83148^Q2483|G8813253^TEST^Dr~ABC
PV1||I|CRO^Bay C Bed 18^TORBAY
HOSPITAL^CROMIE|C|||G8813253^TEST^A.^MR|BRAD^BRADBROOK^R.A.^MR^13|13|||1|N|N|29080|||
|||||201601200834
PV2||3|||13
AL1|^FREE TEXT|PENICILLIN||20160119
```

HL7 document object properties:

ParentId:

CacheSegsGotten:

☒

TimeCreated:

2017-08-01 12:33:22.141

Source:

IsMutable:

☒

OriginalDocId:

AutoBuildMap:

☐

DocType:

2.4:ADT\_A01

Envelope:

Separators:

|^~\&

Invoke Testing Service

Test Results


Cancel

OK

And we would expect to get the response such as this, then it means the integration with PKB Service (Test environment) has been working end-2-end:

ABC\_Router

Clinicom Staging



**ABC\_Router**  
*Production Demo.HL7.MsgRouter.Production*

Request Type: EnsLib.HL7.Message

☒ Request Details

☐ Test Results

**Session Id:** 522 [Visual Trace](#)  
**Request Sent:** 2017-08-01 12:37:19.613  
**Response Received:** 2017-08-01 12:37:19.915

EnsLib.HL7.Message

**HL7 ACK\_A01 Message - Id = 118, DocType = '2.4:ACK', Message Type Category = '2.4'**  
 'General acknowledgment message - Admit / visit notification', 2 Segments

1	MSH ^~\& EnsembleHL7 ISC IHCS-PAS NHS TIE1 201708011237 . ACK^A01 252728 P 2.4
2	MSA AA 252728

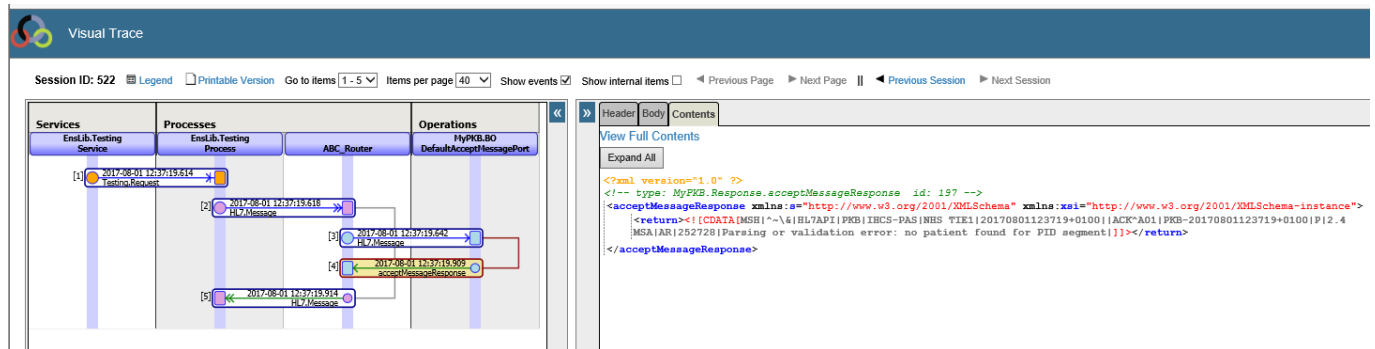
**HL7 document object properties:**

<ObjectId>	118
ParentId	
TopParentId	
Cache Segs Gotten	1
TimeCreated	2017-08-01 12:37:19.913
Source	117
IsMutable	0
OriginalDocId	
BuildMapStatus	OK
AutoBuildMap	0
DocType	2.4:ACK
Envelope	
Separators	^~\&
FS	
CS	^
RS	~
ESC	\
SS	&
SegmentTerminator	
MessageTypeCategory	2.4
Name	ACK_A01
TypeVersion	2.4
Identifier	252728
SegCount	2
ChildCount	0
ParentIds	
RawContent	MSH ^~\& EnsembleHL7 ISC IHCS-PAS NHS TIE1 2017080...
DocTypeCategory	2.4
DocTypeSecondary	ACK
DocTypeName	ACK
Full Size	92

Cancel

OK

And its Ensemble message viewer for this transaction would be such as this:



## 5. Follow-up questions?

Again, this article only describes a demo integration between Health Connect (Ensemble) and PKB Service in its Test environment. If you have any further requirements including production grade functional and non-functional requirement, please do not hesitate to contact an Intersystems Sales Engineer, or Intersystems WRC at [support@intersystems.com](mailto:support@intersystems.com) - we are always happy to support you anytime.

## 6. Acknowledgement

Special thanks to Abdul Kazi and Alice Shrestha, of Chelsea and Westminster Hospital NHS Trust, who kindly organised an additional review with PKB service team on this article.

[#API](#) [#Business Operation](#) [#SOAP](#) [#Health Connect](#) [#HealthShare](#)

---

Source URL: <https://community.intersystems.com/post/healthconnect-ensemble-integration-pkb-service>