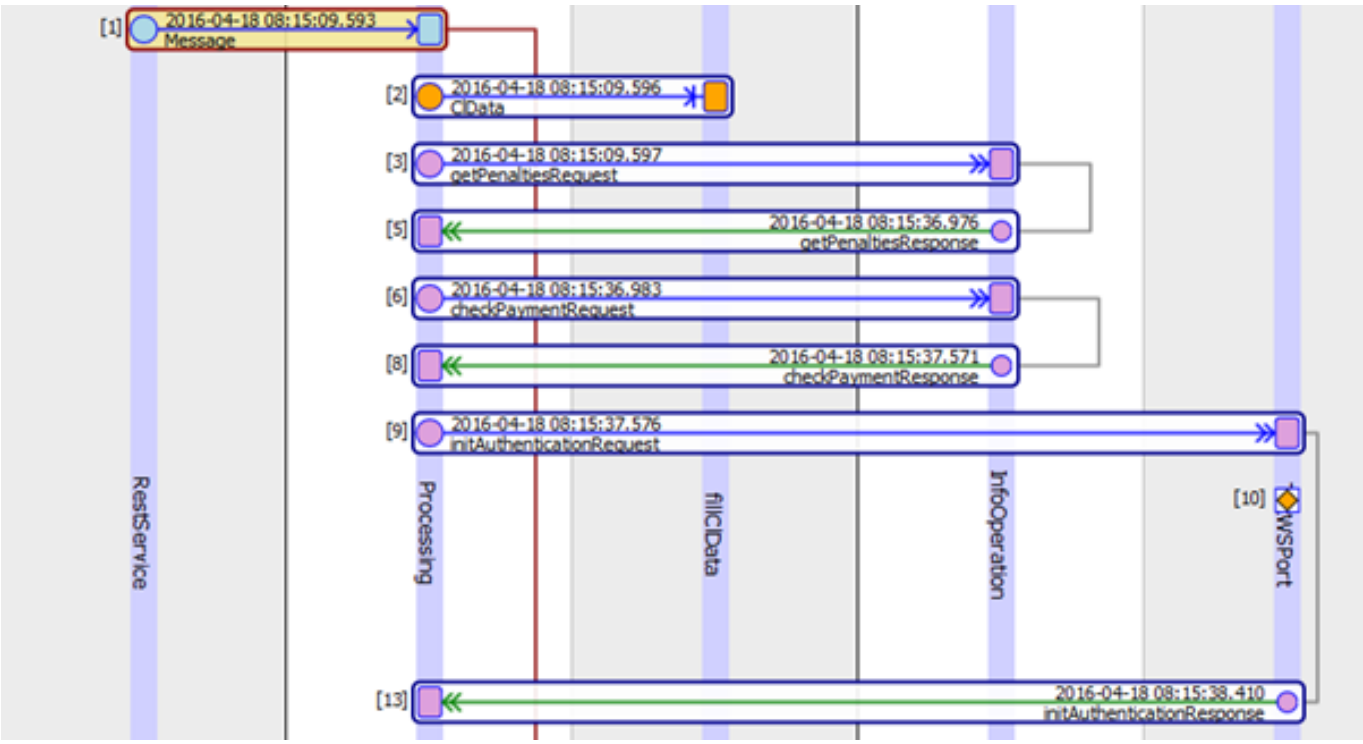
Artículo
Dani Fibla · 26 jun, 2020 Lectura de 6 min

Cómo aprendimos a dejar de preocuparnos y pasamos a amar InterSystems Ensemble



Introducción: nuestra pequeña pero muy ambiciosa empresa llamada " Black Mushroom Studio " tuvo una idea para desarrollar un proyecto de comercio electrónico, y una aplicación móvil que permitiría a los usuarios pagar por ciertos bienes/servicios mediante un agregador de pagos.

Lo que teníamos inicialmente: un esqueleto para la aplicación en Android que, por supuesto, prefería la comunicación mediante HTTP y JSON, y un sistema de pago con una API, es decir, servicios web con contenido SOAP.

Objetivo: hacer que todo funcionara de manera conjunta.

Los siguientes factores influyeron para elegir la tecnología del stack: la velocidad de desarrollo y la capacidad para reaccionar rápidamente ante los cambios. Se suponía que el producto tendría un éxito inmediato. Mientras los competidores seguían haciendo estimaciones, nosotros queríamos lanzar el producto. Mientras nuestros competidores buscaban a los desarrolladores adecuados, se suponía que nosotros estaríamos contando nuestras primeras ganancias. Con este factor restrictivo en curso, todavía necesitábamos una estrategia formal para trabajar, pues todo giraba en torno al dinero de los

inversores y eso es algo que requiere una atención especial.

Podríamos pasar mucho tiempo hablando de las ventajas y las desventajas de ciertas tecnologías de proveedores específicos y de los beneficios que aportan los productos de código abierto. Después de analizar varios productos (este tema por sí mismo merece que redactemos otro artículo), concluimos que InterSystems Ensemble era la mejor opción para satisfacer nuestras necesidades.

Solo uno de nuestros desarrolladores tenía experiencia desarrollando con Caché ObjectScript, pero nadie sabía cómo utilizar Ensemble. Sin embargo, logramos implementar la compleja lógica empresarial de nuestro producto con Ensemble en solo un par de semanas.

Qué fue lo que nos ayudó:

1. Ensemble es un producto completo, que combina un Sistema de gestión de bases de datos (DBMS), un servidor para aplicaciones, un bus de servicios empresariales, un sistema para la gestión de procesos empresariales (BMP) y una tecnología para el desarrollo de aplicaciones analíticas con Inteligencia empresarial (BI). No es necesario conocer varias soluciones e integrarlas

2. Un modelo de almacenamiento basado en objetos. Si queremos guardar un objeto en una base de datos, simplemente lo guardamos en una base de datos

3. Un método muy sencillo de integración con sistemas externos/internos basado en diversos protocolos, gracias a que cuenta con una biblioteca de adaptadores que puede ampliarse

Una solución de primer nivel

Un cliente envía una solicitud con contenido JSON a través del protocolo HTTP hacia un puerto del servidor. Este puerto recibe la información mediante la " caja negra " de Ensemble. El cliente recibe una respuesta sincronizada cuando finaliza el proceso.

¿Qué hay en su interior?

Entre los beneficios de utilizar Ensemble se encuentra la implementación de una producción (una solución de integración en Ensemble) que consiste en tres partes: los servicios empresariales, los procesos empresariales y las operaciones empresariales (de ahora en adelante, utilizaré estos términos sin el sufijo " empresarial " para facilitar la lectura).

Un servicio en Ensemble es un componente que permite recibir solicitudes mediante diversos

protocolos; un proceso es la lógica de las aplicaciones; y una operación es un componente que permite enviar solicitudes salientes hacia sistemas externos.

Toda la interacción dentro de Ensemble se basa en filas de mensajes. Un mensaje es el objeto de una clase que se heredó desde Ens.Message, la cual permite utilizar y transmitir datos desde una parte de la producción hacia otra.

En nuestro caso, el servicio utiliza un adaptador HTTP que se heredó desde EnsLib.HTTP.InboundAdapter, el cual recibe una solicitud enviada por un cliente, la convierte en un " mensaje " y la envía hacia el proceso. El proceso empresarial responde con un " mensaje " que contiene los resultados del proceso y los convierte en una respuesta para el cliente.

Así es como se ve en nuestra solución:

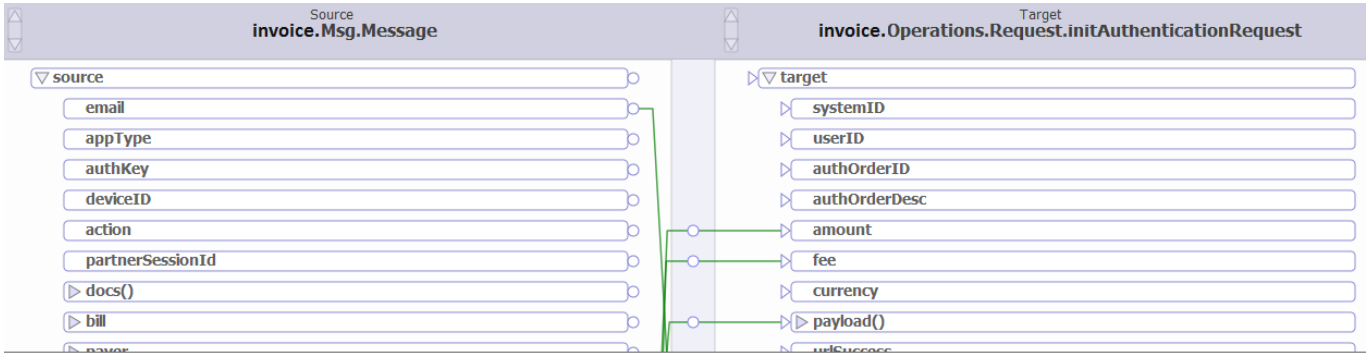
```
Method OnProcessInput(pInput As %Library.AbstractStream, Output pOutput As %Stream.Ob
ject) As %Status
{
   Set jsonstr = pInput.Read()

   // Convirtámoslo en un mensaje
   Set st = ##class(%ZEN.Auxiliary.jsonProvider).%ConvertJSONToObject(jsonstr,"invoic
es.Msg.Message",.tApplication)
   Throw:$$$ISERR(st) ##class(%Exception.StatusException).CreateFromStatus(st)
   // Algo de lógica para el llenado de mensajes con los datos,
   // característica de nuestras consultas

   // Instanciamos un proceso de negocio
   Set outApp=##class(invoices.Msg.Resp).%New()
   Set st =..SendRequestSync("Processing",tApplication,.outApp)
   Quit:$$$ISERR(st) st

   // Convertimos la respuesta a JSON
   Set json=""
   Do ##class(invoices.Utils).ObjectToJSON(outApp,,,"aeloqu",.json)

   // Ponemos el JSON en la respuesta
   Set pOutput=##class(%GlobalBinaryStream).%New()
   Do pOutput.SetAttribute("Content-Type","application/json")
   Do pOutput.Write(json)
   Quit st
}
```

Un proceso empresarial es una implementación de la lógica empresarial de nuestra aplicación. Contiene una secuencia con las acciones que deben realizarse para enviar una respuesta al cliente. Por ejemplo:

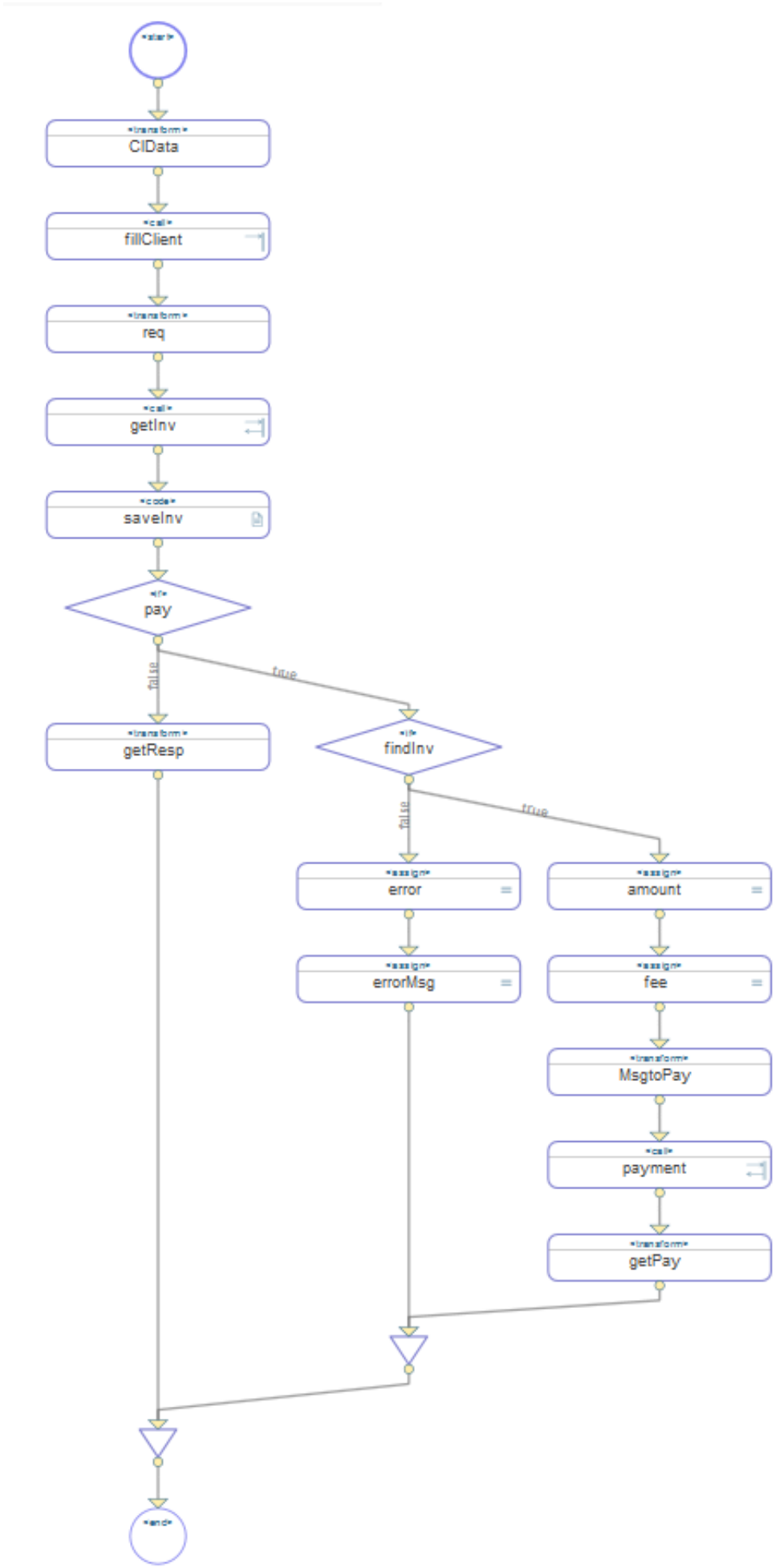
guardar/actualizar los datos del cliente, añadir un nuevo servicio que pueda rastrearse, solicitar el estado de un servicio, pagar por un servicio. Sin una plataforma integrada, tendríamos que escribir una gran cantidad de código.

Qué necesitamos realizar en Ensemble: construir un proceso empresarial en un editor gráfico a partir de diferentes elementos. Los procesos empresariales se describen en el Lenguaje del Proceso Empresarial (BPL, por sus siglas en inglés). Los elementos incluyen asignaciones simples (y no tan simples), conversión de datos, llamadas al código, procesos síncronos y asíncronos y llamadas a las operaciones (hablaremos de esto más adelante).

La conversión de datos también es muy conveniente y es compatible con el mapeo de datos sin la necesidad de escribir líneas de código:

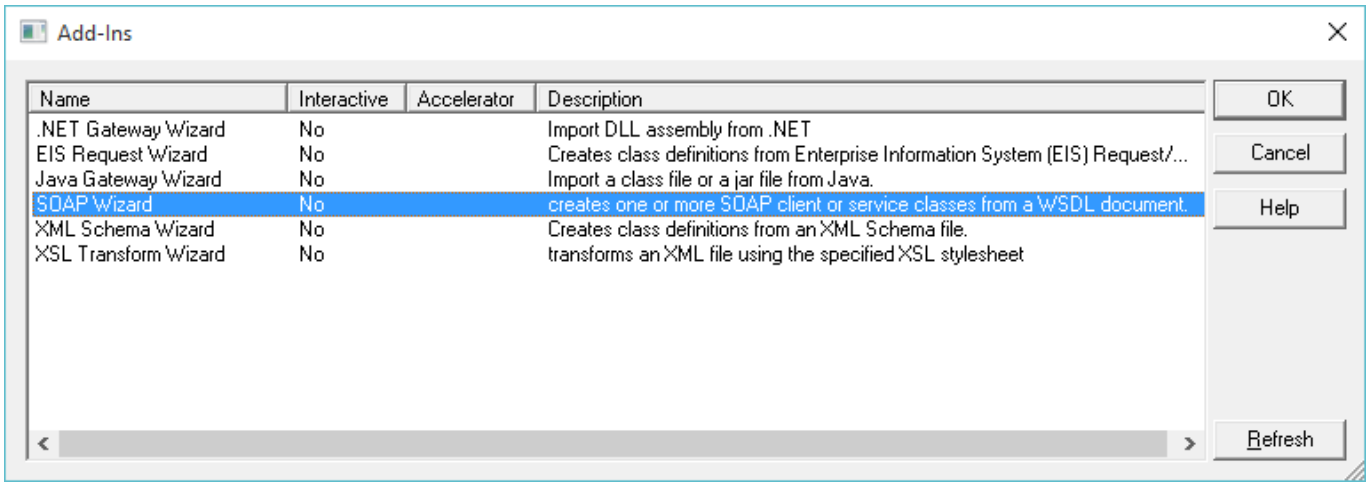


Como resultado, obtuvimos lo siguiente en lugar de un montón de código en cierto momento:

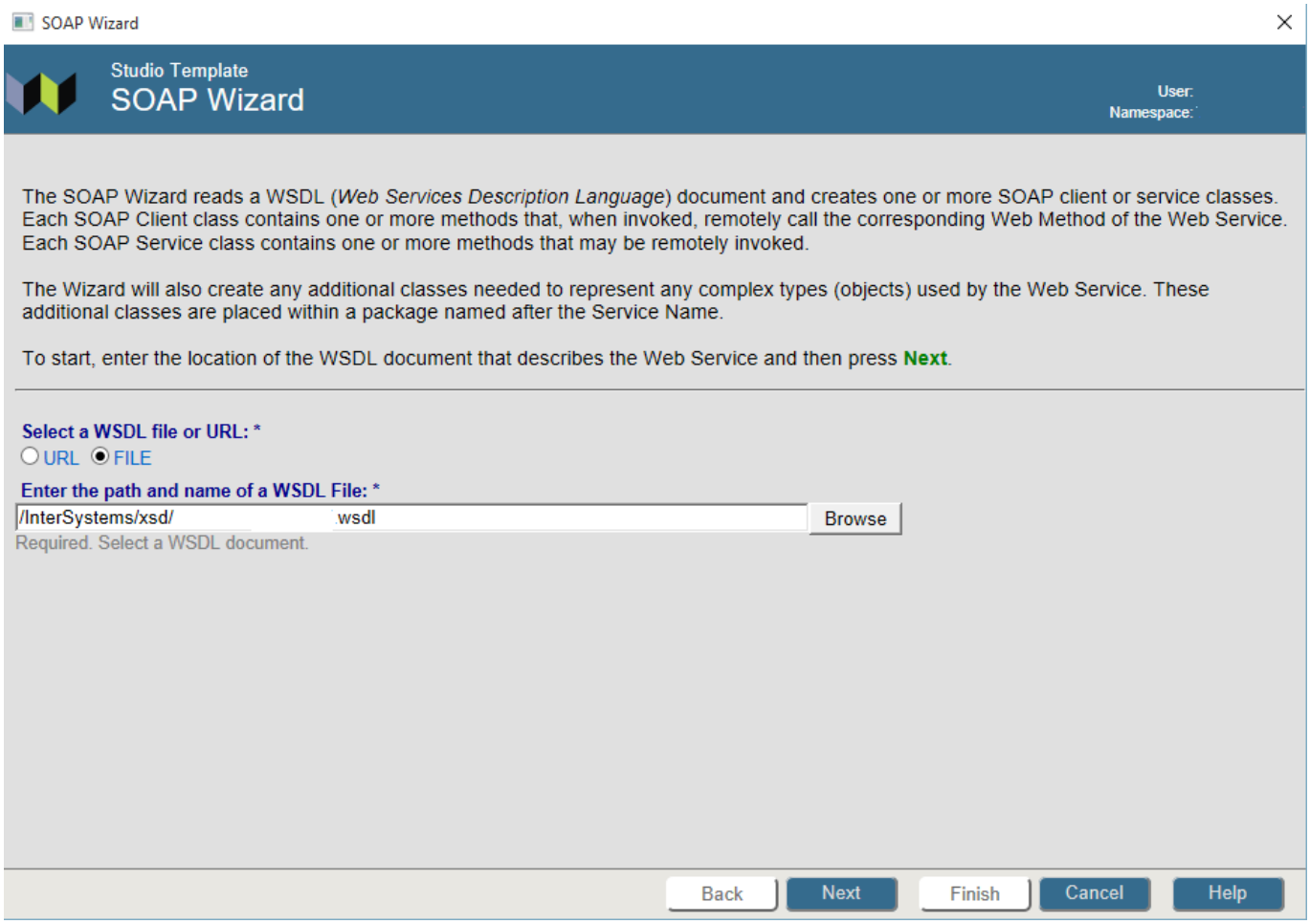

start
transform
ClData
call
fillClient
transform
req
call
getInv
code
saveInv
if
pay
false
true
transform
getResp
if
findInv
false
true
assign
error
assign
errorMsg
assign
amount
assign
fee
transform
MsgtoPay
call
payment
transform
getPay
end

Ahora os comento algunas cosas sobre las operaciones. Esta entidad nos permite enviar una solicitud hacia un sistema externo mediante algún protocolo.

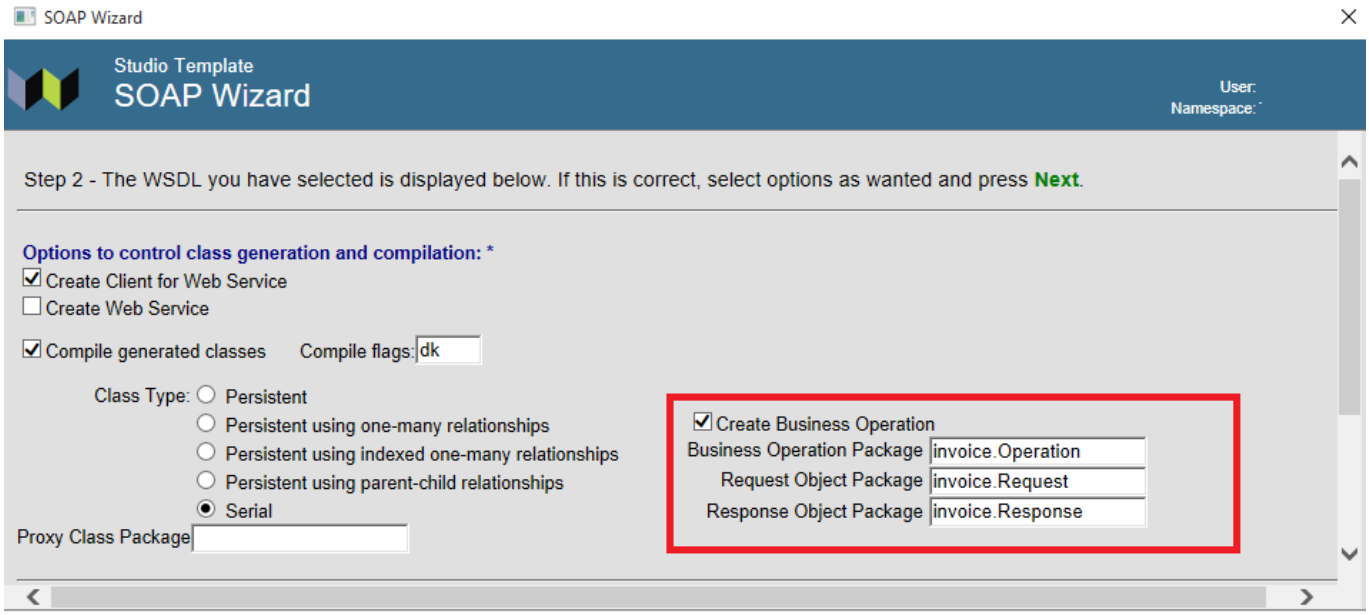
Cómo lo hicimos: utilizamos una extensión incorporada en studio para importar el WSDL proporcionado por el sistema de pago desde el WSDL de Caché Studio:



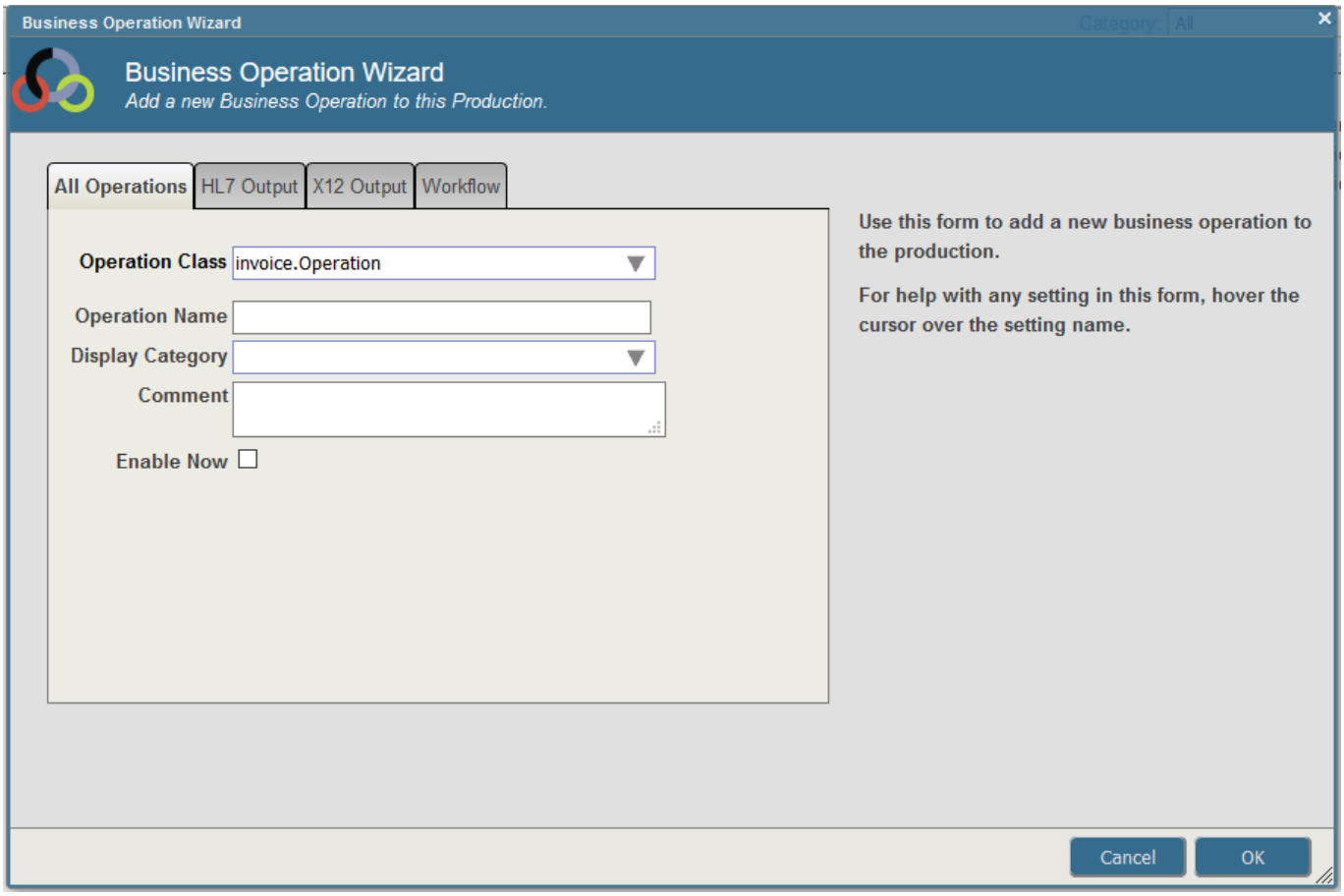
Especificamos la ubicación del WSDL:



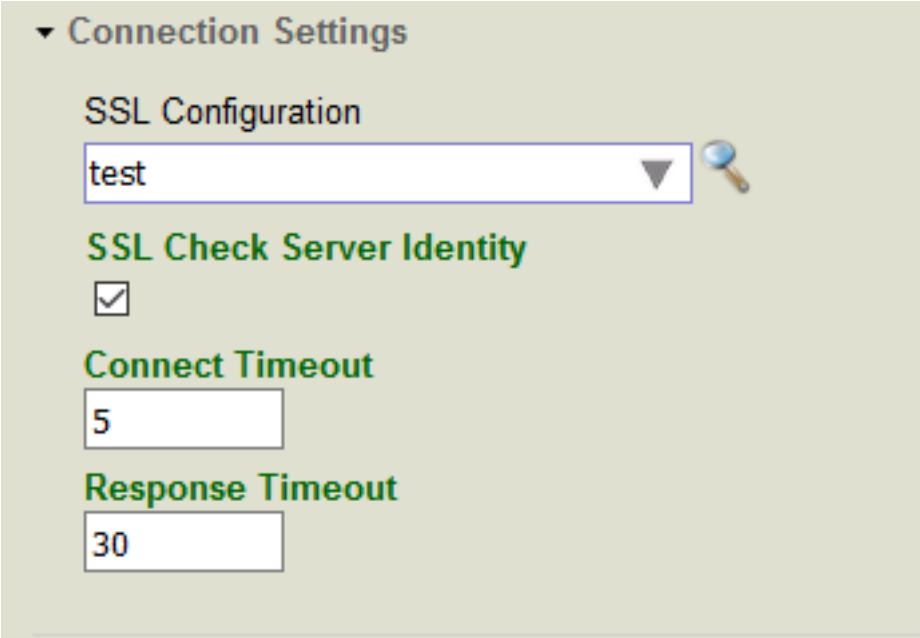
Durante la importación, marcamos la casilla “ Create an operation ” :



Como resultado, obtuvimos un código que ya está listo para realizar solicitudes y procesar las respuestas desde nuestro sistema de pago. Vamos a configurar una operación en el portal y especificaremos su clase:



Ahora instalaremos la configuración SSL (debe crearse previamente mediante el Portal de Administración del Sistema desde la ruta: Administración - Seguridad- Configuraciones SSL/TLS):



¡Hecho! Lo único que debemos hacer ahora es llamar a la operación desde un proceso empresarial. En nuestro caso, tenemos dos de estas operaciones: la parte de la información y la parte correspondiente al pago. Al final, no tuvimos que escribir ni una sola línea de código para interactuar con el sistema de pago.

Listo, se completó la integración.

Sin embargo, también existe un proceso distinto que se utiliza para enviar notificaciones PUSH mediante las herramientas incorporadas en Ensemble, un proceso diferente para obtener registros SFTP desde el sistema de pago para generar recibos, y otro un proceso para generar recibos PDF, pero todos ellos merecen la redacción de otro artículo.

Como resultado, solo dedicamos un par de semanas para la implementación de todo esto (incluido el tiempo necesario para familiarizarnos con la nueva tecnología).

Por supuesto, este producto de InterSystems no es perfecto (nada es perfecto). Cuando trabajamos en nuestras implementaciones, nos enfrentamos a muchas dificultades, y la falta de documentación de apoyo para Ensemble no nos ayudó en absoluto. Sin embargo, en nuestro caso, la tecnología demostró ser eficiente y funcionó bastante bien para nuestros objetivos. Me gustaría felicitar a la empresa por apoyar a los desarrolladores jóvenes y ambiciosos, y por su buena disposición para ayudarnos. Definitivamente planeamos lanzar nuevos productos basados en esta tecnología.

Ya lanzamos una aplicación con esta tecnología y la versión web está en preparación.

#Interoperabilidad #Studio #Ensemble

URL de fuente:https://es.community.intersystems.com/post/c%C3%B3mo-aprendimos-dejar-de-preocuparnos-y-pasamos-amar-intersystems-ensemble