Article
Mihoko Iijima · Mar 5, 2021 6m read

[InterSystems IRIS for the First Time] Interoperability: What a Production is

This article is a continuation of this post**.**

In the previous article, how the Interoperability menu works for system integration was explained.

In this article, I would like to explain how to develop a system integration using the Interoperability menu.

To begin with, what kind of process do you want to create? While thinking about this, make the following content.

- **Production**
- Message
- Components
    - Business Services
    - Business Processes
    - Business Operations

Production is **a definition used to specify the components required for system integration and to store the component settings,** which are configured using the Management Portal (internally stored as a class definition for Production).

For example, suppose you are creating a business service that processes files placed in a specified directory at regular intervals. In that case, it is necessary to set up exactly which directories to monitor and which files to process. A **Production** is prepared to store these settings.

The settings depend on the **adapter** used by the component that sends and receives data.

Adapters are classes to simplify the connection to external systems, some are protocol-specific such as Mail/File/SOAP/FTP/HTTP/SQL/TCP, and some are standards specific HL7.

Please refer to the documentation (protocol-specific adapters and adapters related to EDI documentation for more information on adapters.

Since we will define the necessary components for the **Production**, **"Start Production"** will start the system integration, and **"Stop Production"** will stop the system integration.

The development required to complete the **Production** is the creation of the components necessary for system integration, specifically the following contents:

- Message
- Components (Business Services, Business Processes, Business Operations)
- Data conversion, etc.

The content above will be explained slowly in the articles coming after this one.

First of all, let's start **Production** using the sample **Production** and check the message process by processing data while checking the settings.

Sample can be downloaded from https://github.com/Intersystems-jp/selflearning-interoperability.

To use a Container, download the sample code using the git clone, navigate the clone's directory, and run **docker-compose up -d** It's that easy!

See here for the procedure (it will take some time to create a container).

If you do not use containers, create a new namespace after downloading the sample, and import all class definition files (extension .cls) under the src folder into the created namespace.
For more information on the process of creating a namespace, please refer to the video after 07:03 of this article.

Please refer to the README for more details on the sample code.

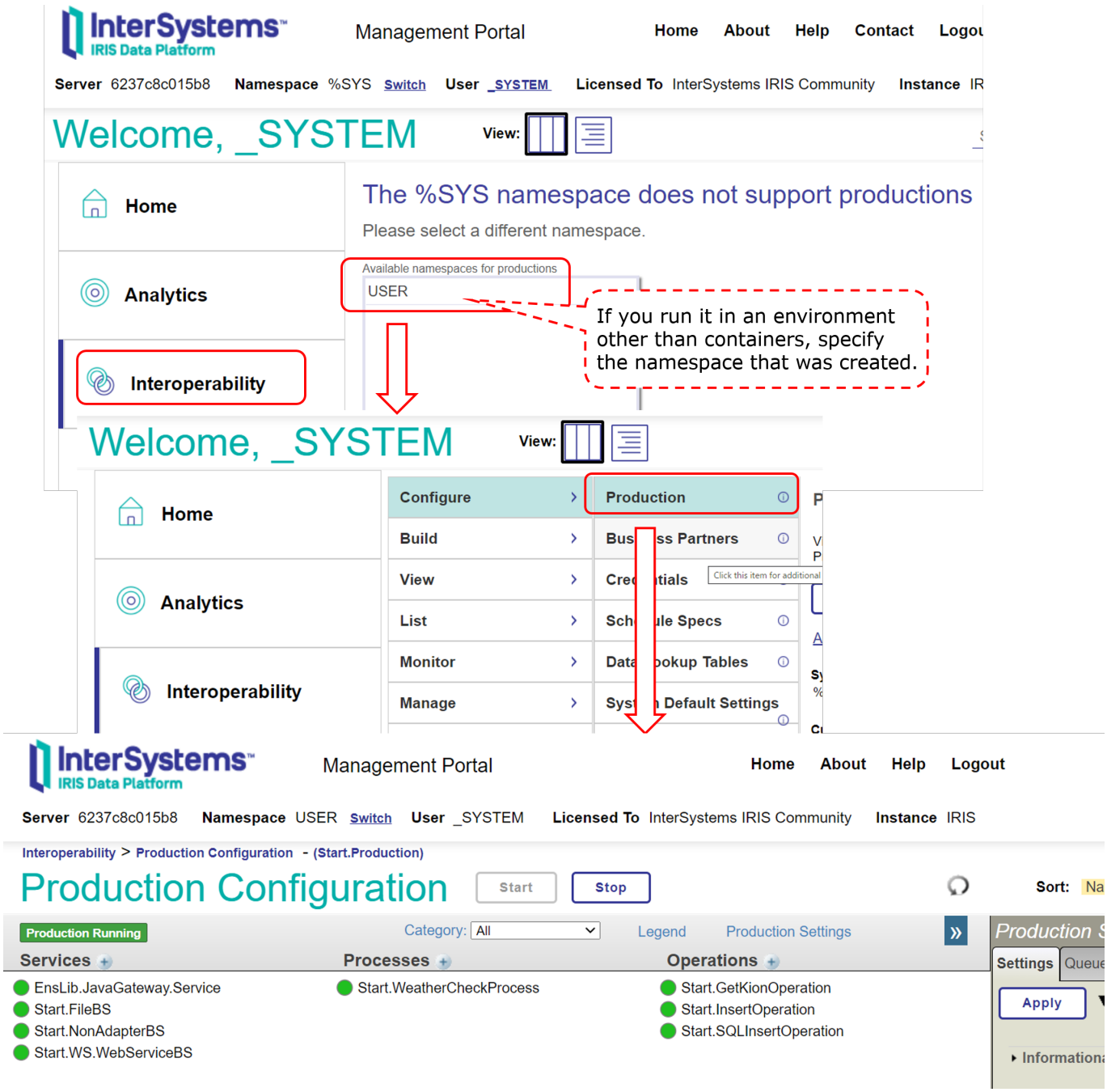
When you are ready, access the management portal (change the web server's port number to match your environment).

http://localhost:52773/csp/sys/UtilHome.csp

Go to the **Management Portal > Interoperability > Configuration > Production**.

If you are using a method other than containers, connect to the namespace where you imported the source code, access **[Configuration] > [Production]**, click the **[Open]** button, select **[Start] > [Production]**, and then click the **[Start]** button.

※ If you are using something other than a container, you will need to make some initial settings. Please set up the contents described below before trying the following contents.



The production page will be displayed as **[Component Name]** for each of the "Service", "Process", and "Operation" components.

Click on the component name to change the contents of the "Settings" tab on the right side of the screen.

For example, when you click on **Start.GetKionOperation** (single click), the display is as follows.

This component has the [HTTP Server] and [URL] settings for connecting to the Web API.
There is a [appid] field at the bottom of the settings where you can enter the API key that you get it.
There is a [lang] field near [appid] and is set "ja" ("ja" = Japanese). [lang] set language of response from OpenWeather. For English, set "en".
When you finish to set these settings , click the "Apply" button.

If you are using a container, the setup is complete. For more information, please click here.

If you are experimenting with something other than containers

---

Please make the following two settings in advance:

1) Configure the SSL client.

Since the Web API to be connected to will be communicated using HTTPS, configure the SSL client on the IRIS side in advance.

To match the settings of the sample production, we will use the name **[openweather]**. The settings in the Production are as follows:

Click the Management Portal > [System Administration] > [Security] > [SSL/TLS Configuration] > [Create New Configuration] button, enter **"openweather"** in the "Configuration Name" field, and then click in the "Save" button to finish.

2) Create a base URL for REST

In the sample production, we have configured it so that the information can be entered via REST, and the base URL for REST needs to be configured on the IRIS side.

In the sample, we set /start as the base URL. Since the Start.REST class exists in the namespace where the sample was imported, we will specify this class as the dispatch class and add %All as the application role to omit authentication at the time of access.

Management Portal > System Administration > Security > Applications > Web Application Path > Click the "Create new web application" button.

In the Name field, specify **/start**; in the Namespace field, specify the namespace from which the sample was imported; in the Dispatch Class field, specify **Start.REST**; in the Allowed Authentication Method field, select **"Unauthenticated"**, and save the file.

After saving, add the **%All** role to the **application role** on the "Application Roles" tab.

Try to send data

Once you are all set up, try to use a business service to send information via REST and let it run.

http://localhost:52773/start/weather/Takoyaki/Osaka

The above example is a URL that supposes that someone has purchased "Takoyaki" in Osaka City.

The screen after execution is as follows.

Check the messages that have been sent to the **Production**.

In the **Management Portal > Interoperability > Configuration > Production**, click on the service below:

Select the **"Messages"** tab on the right side of the screen and click on any number below to the header field column. If you do not see it, reload your browser.

Using the Visual Trace page, you can see the information of **messages** sent and received between components. You can see that the weather information is retrieved from the Web API and sent back in the **light blue frame**.

In this way, you can use tracing to see what data was being sent and received at that time and in what order.

Throughout this article, we have confirmed that **Production** has defined the necessary components and their settings for system integration by referring to the sample code settings.

We also confirmed that we could refer to the messages flowing through the **Production** in chronological order by using the Visual Trace page.

In the next articles, we will discuss the concept behind creating the **"message"** shown in this trace and how actually to define it.

#Beginner #Interoperability #InterSystems IRIS #InterSystems IRIS for Health

---

Source URL:https://community.intersystems.com/post/intersystems-iris-first-time-interoperability-what-production