# ExtendedREFramework

## 1.1 Resumen

Documentación del Framework Extended UIPath.

NOTA: Esta documentación suplementa a la documentación original del REFramework de UIPath (REFramework Documentation-EN.pdf, incluida por defecto al generar un proceso).

ExtendedREFramework es una extensión del REFramework Original de UIPath. El funcionamiento general y arquitectura es idéntica al REFramework, pero incluimos una serie de extensiones:

* Envío de mails al inicio, fin, y excepciones de negocio del proceso
* Conexión a BBDD común
* Integración con Data Service
* Inicialización de Credenciales
* Gestión de códigos de error
* Inicialización de rutas compartidas
* Inicialización (concatenación) de URLs
* Traspaso de queue ítem a la siguiente cola

A continuación, vemos cada uno de los puntos

## 1.2 Config

El ReFramework incluye un Config.xlsx con una serie de claves predefinidas. Las claves que hay definidas (por ejemplo las de mail, ItemDataTable etc) **NO** se deben de borrar, sólo cambiar el valor si es aplicable.

El ExtendedReFramework hace uso de una BBDD adicional o el Data Service de uipath para almacenar los datos relativos a transacciones que puedan cambiar su estado en el tiempo. El uso de uno u otro se decide con las variables del config.

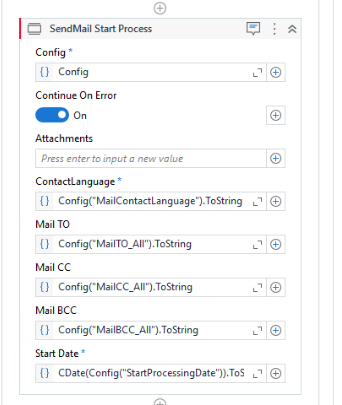
## 1.3 Envío de mails

El Framework hace uso de una librería customizada para envío de mails. Los mails se envían con unos templates previamente definidos. Dichos templates están guardados (en .zip, para contener texto e imágenes) dentro del Storage Bucket RPA CoE/Configs



Los templates además están configurados por Idioma, hay para español, inglés, y portugués.

Ejemplo de actividad de envío de mail



Es importante tener en cuenta que hay variables del config necesarias para que esto funcione (en Constants) y que algunas de estas se deben de cambiar para incluir el nombre del proceso:

MailProcessFullName

MailProcessShortName

MailProcessNickName

MailSenderName

MailCountry

MailBusinessUser

(…)

## 1.4 Conexión a BBDD Común

El Framework hace la conexión a BBDD común en InitAllSettings, y guarda la variable de conexión. Esta BBDD No es la de orchestrator, sino una auxiliar sólo para los procesos y códigos de error

Esta conexión a BBDD se usará en el resto de actividades de BBDD. La credencial que contiene los datos de conexión a BBDD es un Asset (tipo Credencial) en la carpeta RPA CoE. NOTA: La credencial NO es la contraseña del usuario, es el string de conexión entero.

## 1.5 Integración con Data Service

Con la misma filosofía que en el punto anterior, el framework está preparado para hacer uso del Data Service de UIPath. Para usarlo hay que seguir siempre los mismos pasos:

* Crear la entidad en Data Service
* Importar la entidad desde UiPath Studio
* Añadir la actividad de Query Entity Records en GetTransactionData
* Añadir la actividad de Create Entity Record en AddTransactionItem y AddFailedItem (solo si es aplicable para el proceso en desarrollo)
* OPCIONAL: Sacar como output la variable de entidad en el GetTransactionData, y añadirla como variable en Main, y como argumento de entrada en Process.xaml

## 1.6 Inicialización de Credenciales

Extensión para guardar los assets de tipo “Credential”. En la hoja de Credenciales del Config.xlsx se incluyen los nombres de las credenciales (exactamente igual que si fueran assets). El framework las guardará en la variable Config pero al tener dos valores se guardan dichos valores. Por ejemplo:



Esta credencial se guardará como Config(“CredMail\_User”) (String) y Config(“CredMail\_Pass”) (SecureString)

## 1.7 Gestión de códigos de error

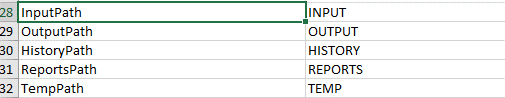
Los errores controlados que lancemos en el proceso deben de tener un código de error asociado (tanto si es System Exception como Business Exception). La nomenclatura y estructura de los códigos viene detallado en el documento de buenas prácticas. Resumen: el único contenido del mensaje de excepción debe ser el código de error: new Exception(“0-0001-0003-0001”)

El framework resuelve y transforma el código de error en un mensaje, añadiéndolo al ítem de cola (si es aplicable). Para el buen funcionamiento del proceso (en especial al usar el data service) es importante que los códigos de error sigan dicha estructura (X-XXXX-XXXX-XXXX). En caso contrario, se debe de modificar los workflow de resolución de errores

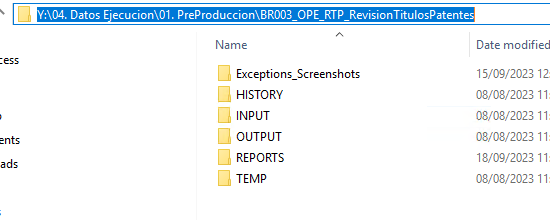
Si el error no tiene un código asociado (es decir, es un error no controlado) se le asigna el código de error genérico de no controlado (1-9999-9999-9999). No es necesario intentar capturar todos los errores y cambiarles el código.

## 1.8 Inicialización de rutas compartidas

El framework expone rutas compartidas para utilizarlas durante el proceso:



Estas rutas se calculan y generan (si no existen) durante la inicialización. La ruta se genera a partir de este nombre, y el nombre del proceso, por ejemplo:



El framework generará estas rutas por defecto, aunque no sea necesario usarlas.

Se genera siempre en función a una ruta base (la cual a su vez depende del Entorno, pre o pro indicado)

## 1.9 Specific Content de Queue Items

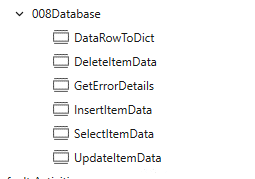
Para mejorar la funcionalidad de datos de ítem de cola, se hace uso de tablas en BBDD. Los ítems de cola siguen llevando un specific content con los datos necesarios, pero además, se guardan los mismos datos en la tabla de BBDD del proceso (o en Data Service). Esos datos sí se pueden actualizar, filtrar etc con facilidad, a diferencia de los del ítem de cola.

Además, se hace control de duplicados en base a la Referencia del ítem

NOTA: En el caso de usar Data Service, es necesario añadir manualmente las actividades relativas al data service e importar la entidad

Para ello es esencial NO usar la actividad de Add Queue Item sin más, sino que hay que usar el Workflow de AddTransactionItem. A este workflow le tenemos que pasar un Dict(String,Object) que es el Specific Content

La librería RPA008.Database contiene las actividades para gestión de estos elementos en BBDD:



NOTA: Aunque es posible tener más columnas en la tabla de BBDD que keys en el ítem de cola (por ejemplo porque se añadan más tarde a lo largo del proceso), el caso contrario NO es posible.

## 1.10 Inicializar Paths

Para concatenar dos componentes de URLs (URL base + Endpoint) se hace uso de la hoja de Paths y el workflow correspondiente. Durante la inicialización, se hace esta concatenación y se guarda en Config. Esta concatenación vale tanto para URLs (páginas web) como para rutas de archivos

# Resumen de ficheros que se cambian respecto a REFramework

**InitAllSettings**: Añadido el flujo de credenciales y conexión a BBDD. Inicializada fecha de ejecución de proceso

**Main**: Añadidas actividades de envío de mail al principio y fin de proceso. Adaptadas variables globales para hacer uso del ítem data de BBDD y otras variables necesarias. Añadidos invokes para InitAllPaths. Añadidas alertas y gestión de códigos de error en inicialización y get transaction data

**InitAllPaths**: Nuevo workflow que inicializa los paths y URLs (1.9)

**InitSharedFolders**: Nuevo workflow que inicializa las carpetas (1.7)

**SetTransactionStatus**: Añadida la lógica de conversión de códigos de error.

# Extensiones REFramework

Basado en el ExtendedREFramework, existen las siguientes extensiones. Son frameworks ya customizados para ciertas tareas, la arquitectura general es idéntica al original, pero se deja preparado el código correspondiente para dichas tareas:

* **ExtendedREFramework\_ColectorExcel**: Para procesos de tipo colector en los que se obtiene un Datatable. El transaction ítem es un DataRow de dicho Datatable. La lógica de obtención de dicho datatable se tiene que hacer en GetTransactionData, dentro de GetDataTableData. La misma lógica también se puede usar si el transaction data es, por ejemplo, un List<Dictionary> o JArray (muy común cuando los datos se cogen de una API REST)
* **ExtendedREFramework\_Reporter:** Para procesos de reporte. Incluye la gran mayoría del código necesario para generar el reporte, sólo hay que customizar el XLSX de template (si es necesario), las variables del config, y el workflow Getdataforeachqueueitem (si es necesario)
* **ExtendedREFramework\_Dispatcher:** Miniproceso para monitorizar una cola y relanzar el proceso indicado si es necesario.