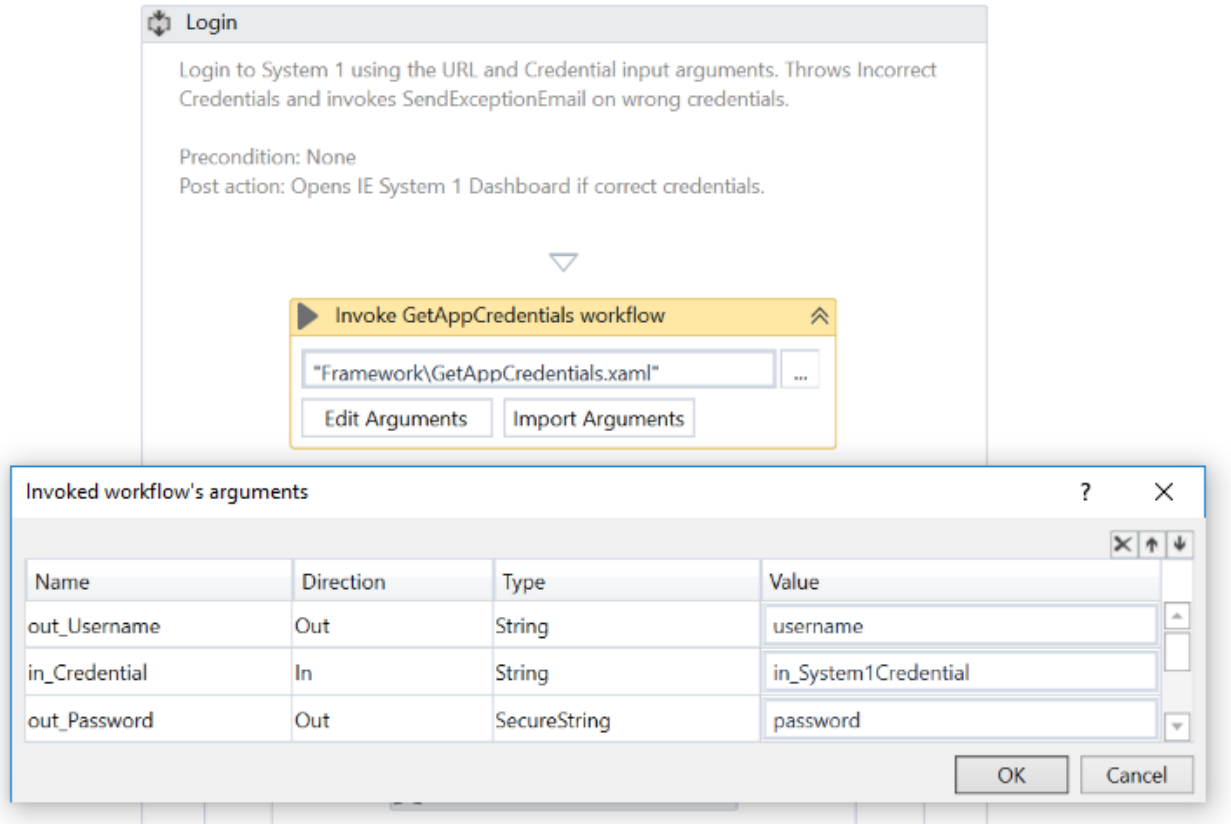
Itinerario – Cálculo del *hash* de seguridad del cliente

Itinerario – Cálculo del código *hash* de seguridad del cliente

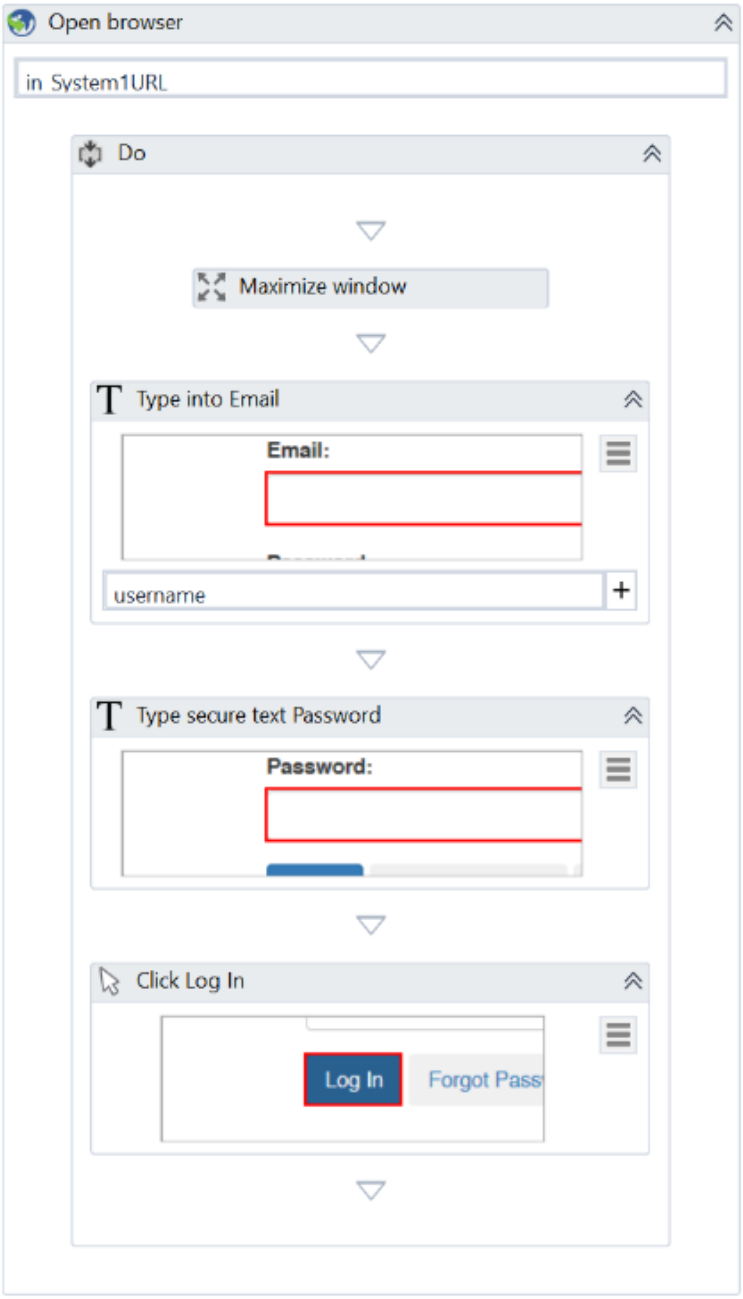
- Comience por la plantilla REFramework.
    - Empezamos con una implementación simple para hacer una demostración del REFramework sin usar Orchestrator Queues (colas).
    - Eche un vistazo a los datos de entrada (lista de elementos en cola); podemos extraer toda la tabla con el asistente Data Scraping.
    - El elemento de transacción es una línea de datos de la lista. El mismo enfoque se aplica cuando la entrada es una tabla de datos extraída de hojas de cálculo de Excel, archivos CSV y bases de datos.
- Es recomendable mantener los valores que tienden a cambiar en el archivo de configuración. Abra el cuaderno de trabajo Data\Config.xlsx y cambie a la hoja Settings.
    - Añada los parámetros de configuración para **System1 URL** y **SHA1 Online URL**.
    - La aplicación System1 precisa autenticación. Utilizaremos Orchestrator Assets (activos) para guardar las credenciales de System1. Añada otro parámetro de configuración, System1_Credential, para guardar el nombre de la credencial.
    - Añada un Asset de tipo credencial y escriba el nombre de usuario y la contraseña de System1. Asegúrese de que el nombre del Asset coincida con el valor del parámetro de configuración System1_Credential.
- Pase a la hoja **Constants** del cuaderno de trabajo **Config** y fije el valor de MaxRetryNumber en 2. Este parámetro controla cuántas veces el marco de trabajo intenta procesar un elemento de trabajo cuando falla con una excepción de aplicación, antes de pasar al siguiente.
- Realice los siguientes cambios en el marco de trabajo.
    - La variable **TransactionItem** en el archivo **Main** debe ser de tipo System.Data.DataRow, ya que estamos extrayendo toda la tabla para procesar una fila cada vez. También debe modificar el tipo de argumento en los flujos de trabajo **GetTransactionData**, **Process** y **SetTransactionStatus** para que coincida con el tipo de TransactionItem.
    - Elimine las tres actividades **SetTransactionStatus** del flujo de trabajo SetTransactionStatus, ya que no vamos a utilizar la función de transacción que ofrece Orchestrator.
- En este ejercicio utilizaremos dos aplicaciones, **ACME System1** y **SHA1-Online.com**. Cree dos carpetas, «System1» y «SHA1Online», en el directorio raíz de la solución y utilícelas para los flujos de trabajo creados para las dos aplicaciones.

- Cree una secuencia en blanco en la carpeta System1 para el proceso de inicio de sesión en System1. Queremos crear un componente reutilizable que se pueda emplear con muchas credenciales diferentes.
    - Es una práctica recomendable comenzar la nueva secuencia con una breve nota para explicar la finalidad del flujo de trabajo. Esto es lo que haremos en todos los archivos futuros. La nota debe comenzar con una descripción que incluya los argumentos que se han utilizado, una condición previa y una acción posterior.
    - Cree dos argumentos de entrada, uno para **System1 URL** y otro para **System1 Credential**.
    - Haga una llamada al archivo de flujo de trabajo Framework\GetAppCredential. Utilice el argumento System1 Credential como entrada. Cree dos variables para guardar los dos valores de salida: nombre de usuario y contraseña.
    - La secuencia debería quedar así:

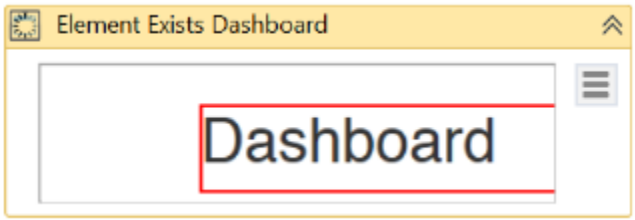


    - Utilice una actividad **Open Browser** para acceder a System1 URL. Por defecto se utilizará Internet Explorer salvo que se modifique la propiedad **BrowserType**.

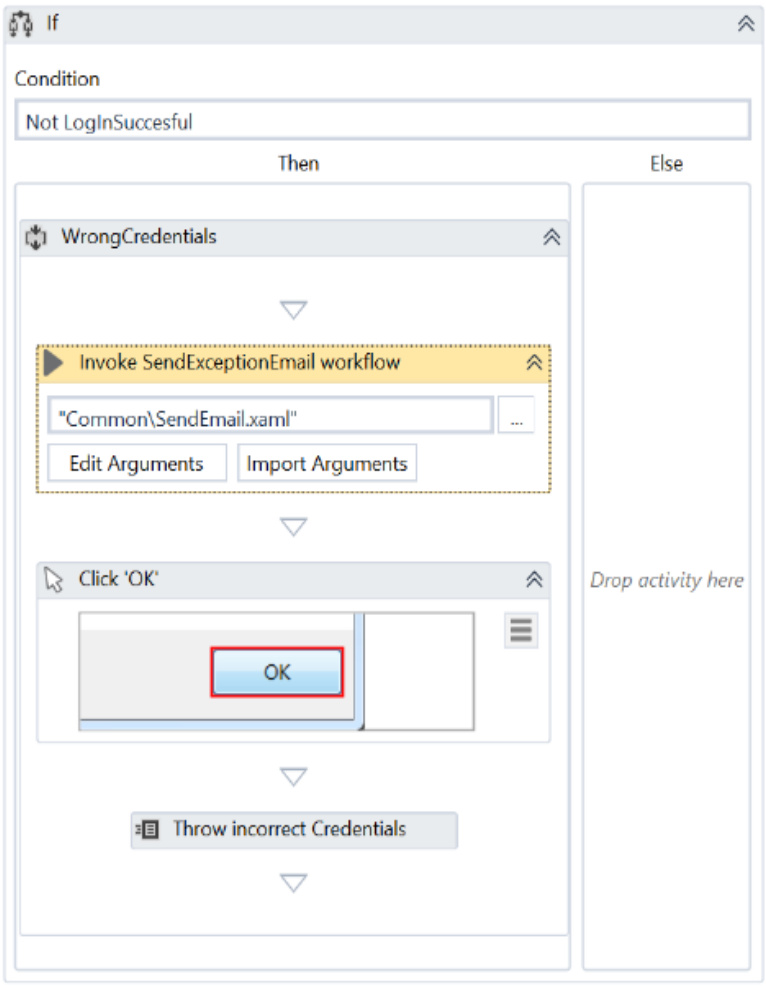
- Opcionalmente puede añadir en la secuencia **Do** de **Open Browser** una actividad **Maximize Window** para una mejor visibilidad al realizar las tres siguientes actividades.
- Utilice una actividad **Type Into** para indicar el nombre de usuario. Asegúrese de que está marcada la casilla **SimulateType** en el panel **Properties**.
- Utilice una actividad **Type Secure Text** para introducir la contraseña. Asegúrese de que está marcada la casilla **SimulateType** en el panel **Properties**.
- Utilice una actividad **Click** para seleccionar el botón Log In. Asegúrese de que está marcada la casilla **SimulateClick** en **Properties**.
- La actividad Open Browser debería quedar así:



- Siempre es recomendable pensar en las excepciones que pueden darse durante la ejecución de nuestro proceso. En este caso debemos comprobar si se realizó correctamente el inicio de sesión. Este requisito también se presenta en el archivo PDD. Intente iniciar sesión con unos datos incorrectos y observe la diferencia.
- Utilice una actividad **Element Exist** para comprobar si se inició sesión correctamente buscando un elemento que aparece solo en tal caso.



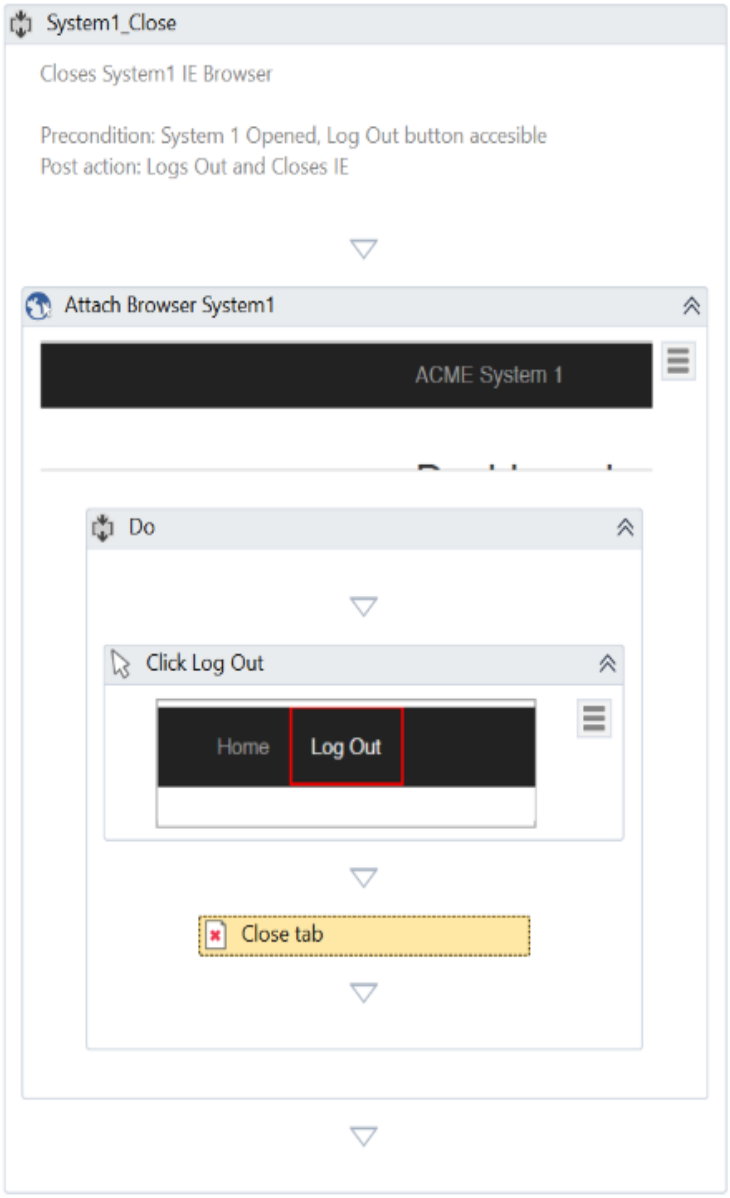
- Utilice una actividad **If** para comprobar si falló el intento de inicio de sesión. Si falló, proceda como se indica a continuación.
  - Envíe un correo electrónico con la excepción. Se recomienda crear aparte un flujo de trabajo reutilizable y hacer una llamada a él desde la sección **Then**.
  - Cierre el mensaje de error con una actividad **Click**.
  - Emita una excepción por credenciales incorrectas introducidas en System1 para detener el proceso.
- La actividad **If** debería quedar así:



- Realice una prueba unitaria del archivo creado usando los valores por defecto de los argumentos.

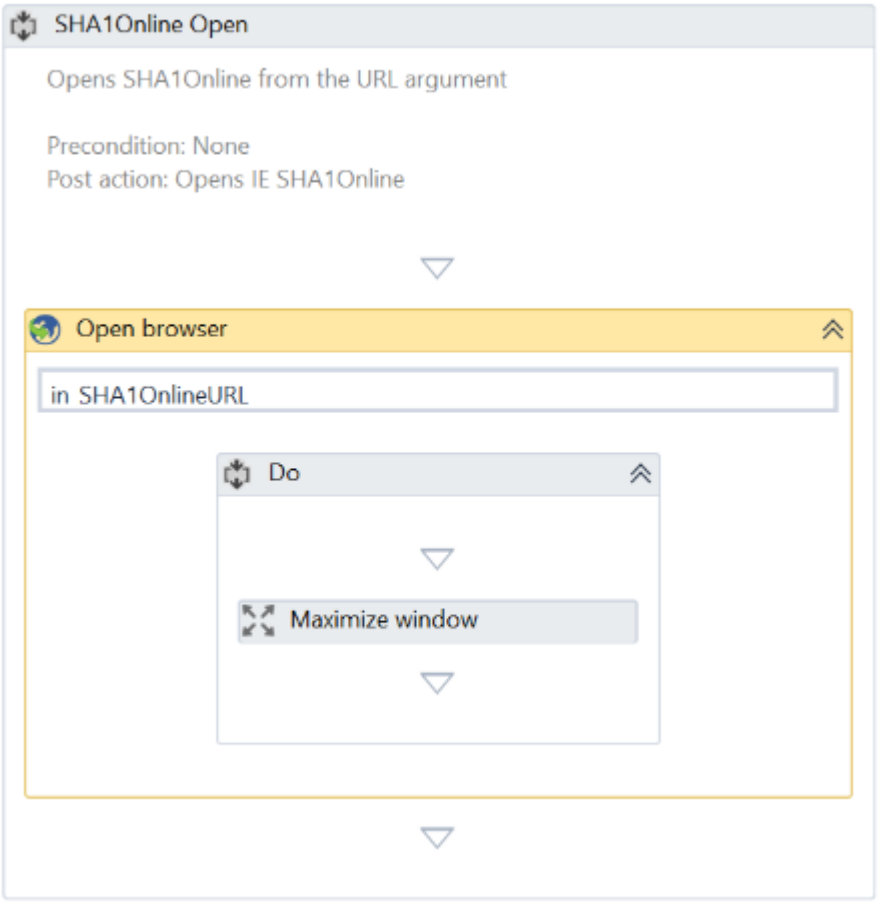
- A continuación crearemos el flujo de trabajo para cerrar sesión y salir de System1. Cree una secuencia en blanco en la carpeta System1.
  - La secuencia nueva debe comenzar con una nota.
  - Utilice la actividad **Attach Browser** para adjuntar a la ventana del navegador que contenga la aplicación System1. Podemos extraer de aquí el primer requisito previo para este flujo de trabajo: la aplicación System1 debe estar abierta.
  - Dentro de la actividad de navegador adjunta, arrastre y suelte una actividad **Click**. Establezca el objetivo en el botón Log Out. Este es el segundo requisito previo para este flujo de trabajo: el botón de cierre de sesión debe ser accesible.
  - Utilice una actividad Close Tab para cerrar la ventana del navegador de System1.

- La secuencia debería quedar así:

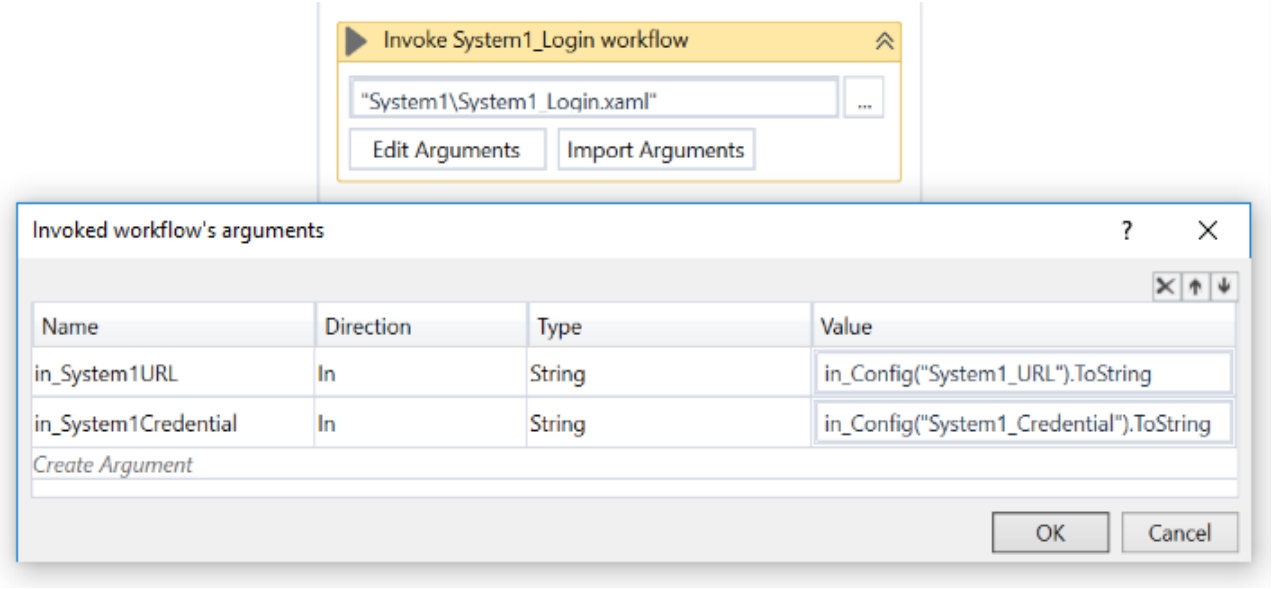


- A continuación, haremos lo mismo con la otra aplicación utilizada en el proceso SHA1 Online.
- Cree un flujo de trabajo con secuencia en blanco en la carpeta SHA1Online para abrir la aplicación. Esta secuencia es más sencilla, puesto que la aplicación no precisa autenticación.
  - Naturalmente, comenzaremos con una nota.
  - La URL de la aplicación debe pasarse al flujo de trabajo con un argumento, para que el proyecto sea más fácil de mantener. El valor de la URL se guarda en el archivo de configuración del proceso.

- Utilice una actividad **Open Browser** con el argumento creado como URL de entrada. Por defecto se utilizará Internet Explorer salvo que se modifique la propiedad **BrowserType**.
- Opcionalmente puede utilizar una actividad **Maximize Window** en la ventana del navegador.
- La secuencia debería quedar así:

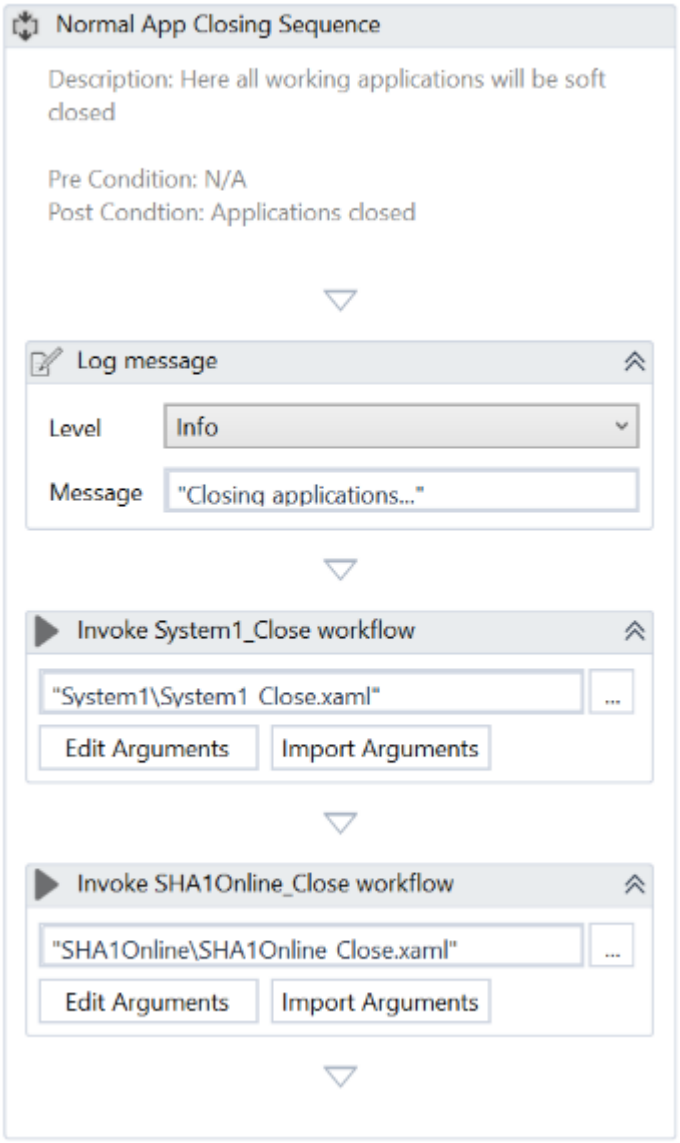


- Cree una secuencia en blanco para cerrar la aplicación SHA1Online. Esto debería ser sencillo.
- Ahora que tenemos los flujos de trabajo para abrir y cerrar ambas aplicaciones, podemos realizar los cambios en las partes de inicio y cierre del marco de trabajo.
- Abra el flujo de trabajo **Framework\InitAllApplications**.
  - Arrastre y suelte el archivo System1_Login. Se creará automáticamente una actividad **Invoke Workflow File**.
  - Haga clic en **Import Arguments** y enlace los valores a los del archivo Config.
  - La actividad **Invoke Workflow File** debería quedar así:



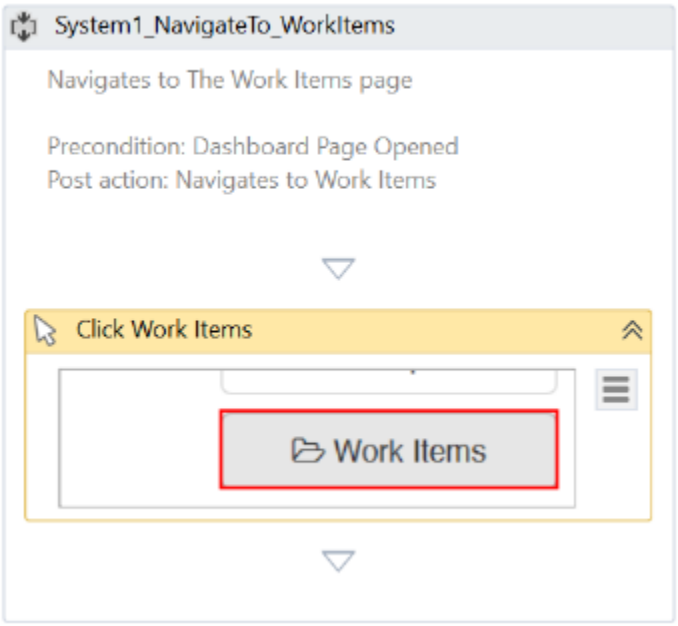
- Arrastre y suelte el flujo de trabajo SHA1Online_Login.
- Haga clic en **Import Arguments** y enlace el argumento URL al valor del archivo Config.

- Abra **Framework\CloseAllApplications**.
  - Arrastre y suelte los dos flujos de trabajo creados para cerrar las aplicaciones System1 y SHA1Online.
  - La secuencia resultante debería quedar así:



- Abra **Framework\KillAllProcesses.**
    - Ambas aplicaciones están alojadas dentro de un navegador web. Si el valor predeterminado de la propiedad **BrowserType** en las actividades **Open Browser** no se ha modificado, se utilizará Internet Explorer.
    - Utilice una actividad **Kill Process** y defina «iexplore» como nombre del proceso.
- Cree un flujo de trabajo con secuencia en blanco en la carpeta System1 para navegar a los **Work Items** en la aplicación System1. Queremos incluir esta acción en un flujo de trabajo aparte, puesto que navegaremos a los Work Items también en otros proyectos.
    - Añada la nota correspondiente.

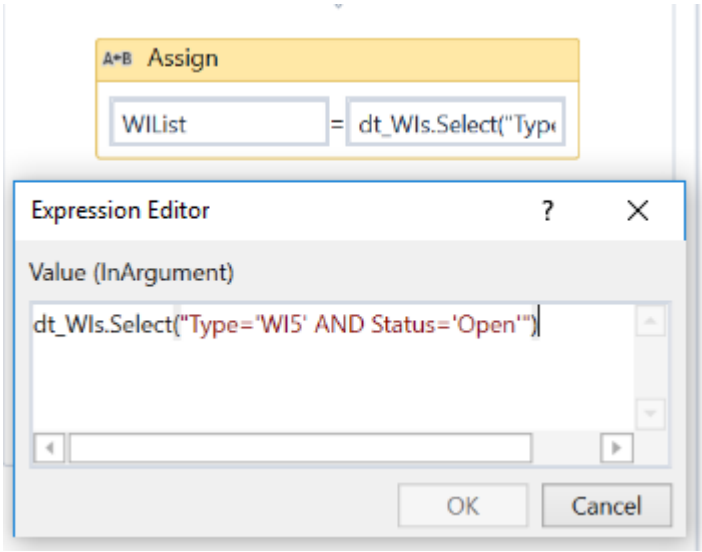
- Utilice una actividad **Click** para seleccionar el botón Work Items. Asegúrese se utilizar un selector completo, puesto que esta actividad no está dentro del ámbito de Attach Browser.
- El proyecto debería quedar así:



- Antes de empezar a procesar las transacciones necesitamos añadir las actividades necesarias para leer los datos de entrada en el estado Init. Normalmente, esto es muy sencillo. Puede utilizar Read Range o Read CSV. Sin embargo, en nuestro proceso, los datos de entrada están almacenados en un sitio web, por lo que se precisan más pasos.
- Recuerde que el objetivo de este proceso es recuperar el código *hash* de cada elemento de tipo WI5. Para ello, necesitamos en primer lugar una lista de todos los elementos WI5.
- Cree un flujo de trabajo con secuencia en blanco en la carpeta System1 para extraer una variable de tabla de datos que contenga todos los elementos de trabajo de la aplicación System1. Extraeremos todos los elementos de trabajo disponibles y luego filtraremos los de tipo WI5.
  - Utilice el asistente Data Scraping para extraer toda la tabla en formato HTML. Cuando se le pregunte si los datos abarcan varias páginas, responda Sí e indique el botón de la página siguiente.
  - Asigne el valor 0 a la opción **Maximum number of results** para que todos los elementos identificados se puedan extraer como resultados.
  - Cree un argumento de salida y asígnele el valor de la tabla de datos extraída.

- Si había añadido una nota, ya puede dar por concluido este flujo de trabajo. Si no lo hizo, añada ahora la nota.

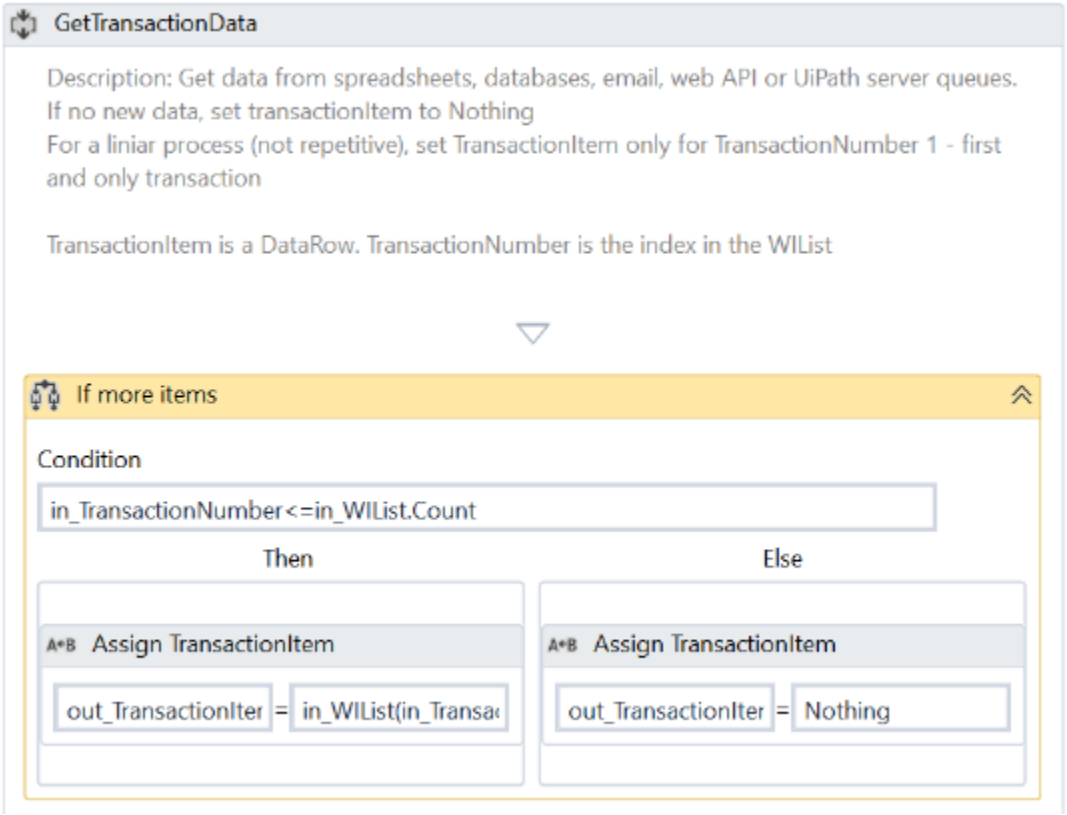
- Abra el flujo de trabajo **Main** y expanda el estado **Init** haciendo doble clic sobre él.
  - En la zona de entrada, localice dónde se hace la llamada al flujo de trabajo KillAllProcesses.
  - Añada una nueva secuencia tras la actividad **Invoke KillAllProcesses** para leer la tabla de datos de transacciones de entrada.
  - Dentro de esta secuencia, llame a cuatro de los flujos de trabajo creados previamente del modo siguiente:
    - Inicio de sesión en System1.
    - Navegar al elemento de trabajo en System1.
    - Extraer la tabla de datos de elementos de trabajo en System1.
    - Cerrar System1.
  - Importe y enlace argumentos donde sea necesario.
  - De la lista de elementos de trabajo, extraiga únicamente los elementos necesarios para el proceso actual (los de tipo WI5) que tengan el estado «Open».
    - Utilice el método DataTable.Select para filtrar los elementos que no cumplan los criterios anteriores. Asigne el resultado a una variable nueva.
    - La actividad **Assign** debería quedar así:



  - Vamos a procesar cada elemento de la matriz al mismo tiempo. De este modo, todos los objetos DataRow se convertirán en elementos de transacción. El índice de transacción comienza con el 1, pero las matrices

tienen base cero, por lo tanto el índice de elemento en la matriz es 1 menos que el número de transacción.

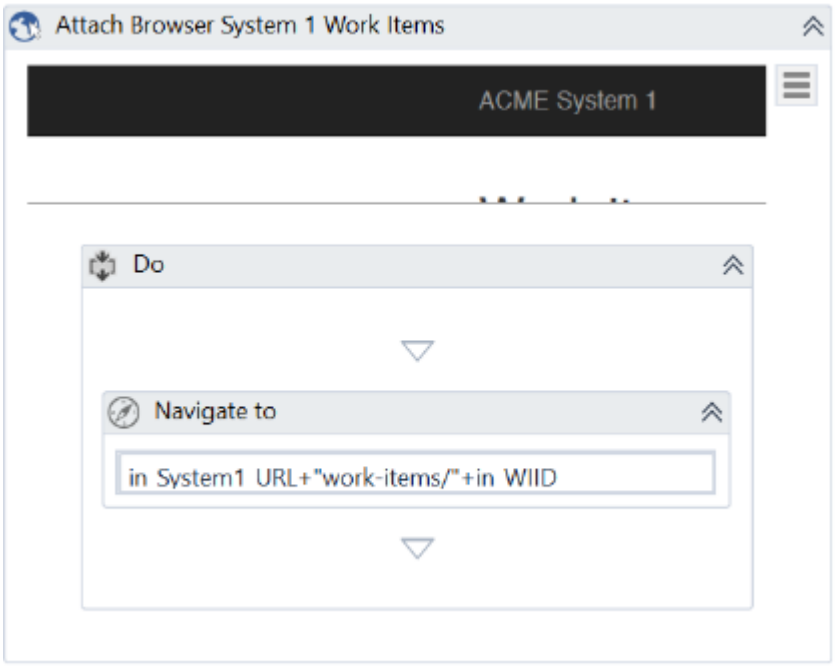
- Abra el flujo de trabajo **Framework\GetTransactionData**.
  - Actualice la nota para el proceso actual.
  - Añada un argumento de entrada para la matriz de los elementos de trabajo.
  - Recuerde que nuestro elemento de transacción es un objeto de la matriz de elementos de trabajo. Solo podemos tener una nueva transacción si el número de transacción es menor o igual al índice del elemento de la matriz.
  - Añada una actividad **If** para comprobar si hay alguna transacción nueva para procesar.
    - En la sección **Then**, utilice una actividad **Assign** para definir el valor del argumento del elemento de transacción de salida conforme a su índice.
    - En la sección **Else**, defina el valor del elemento de transacción a Nothing.
  - La secuencia Get Transaction Data debería quedar así:



  - Además, cuando haya elementos de transacción pendientes de procesar debemos ajustar el ID de la transacción para que coincida con el valor del ID del elemento de trabajo. Las actividades ya están en su lugar, así que solo

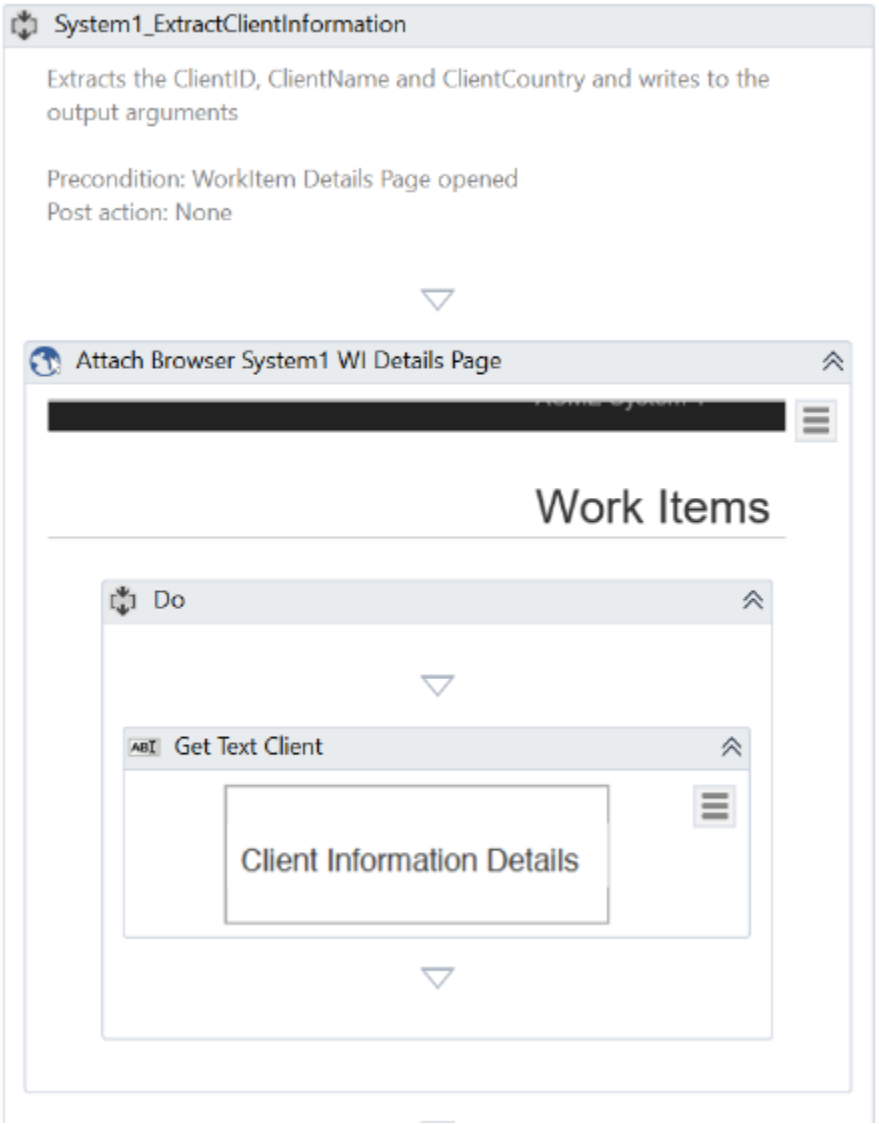
nos queda cambiar el valor de TransactionID en la actividad **Assign** a «out_TransactionItem("WIID").ToString».

- Al hacerlo, habremos finalizado la configuración del marco de trabajo. Podemos probar el flujo de trabajo **Main** ejecutándolo y comprobando la extracción de los datos de entrada. Utilice la ventana de registro de salida para comprobar si los datos son correctos.
- Ahora comenzaremos a desarrollar los flujos de trabajo para procesar los elementos de trabajo.
- Cree una secuencia en blanco en la carpeta **System1** para navegar a la página Work Item Dtails. Póngale el nombre **System1_NavigateTo_WIDetails**. Podemos utilizar el ID del elemento de trabajo para ir directamente a la página de detalles del elemento en cuestión. Abra un elemento de trabajo y observe el formato de la URL (consta de la URL de System1, la cadena «/work-item/» y el ID del elemento de trabajo).
    - El ID del elemento de trabajo y la URL de System1 son la secuencia de entrada requerida. Cree dos argumentos de entrada, uno de tipo Int32 y otro de tipo String.
    - Adjúntelos al panel de control de System1 y utilice una actividad **Navigate To** para ir a la página Work Item Details.
    - La secuencia debería quedar así:



- Cree un flujo de trabajo con una secuencia en blanco en la carpeta System1 con el nombre **System1_ExtractClientInformation**. Lo utilizaremos para recuperar la información sobre un elemento.

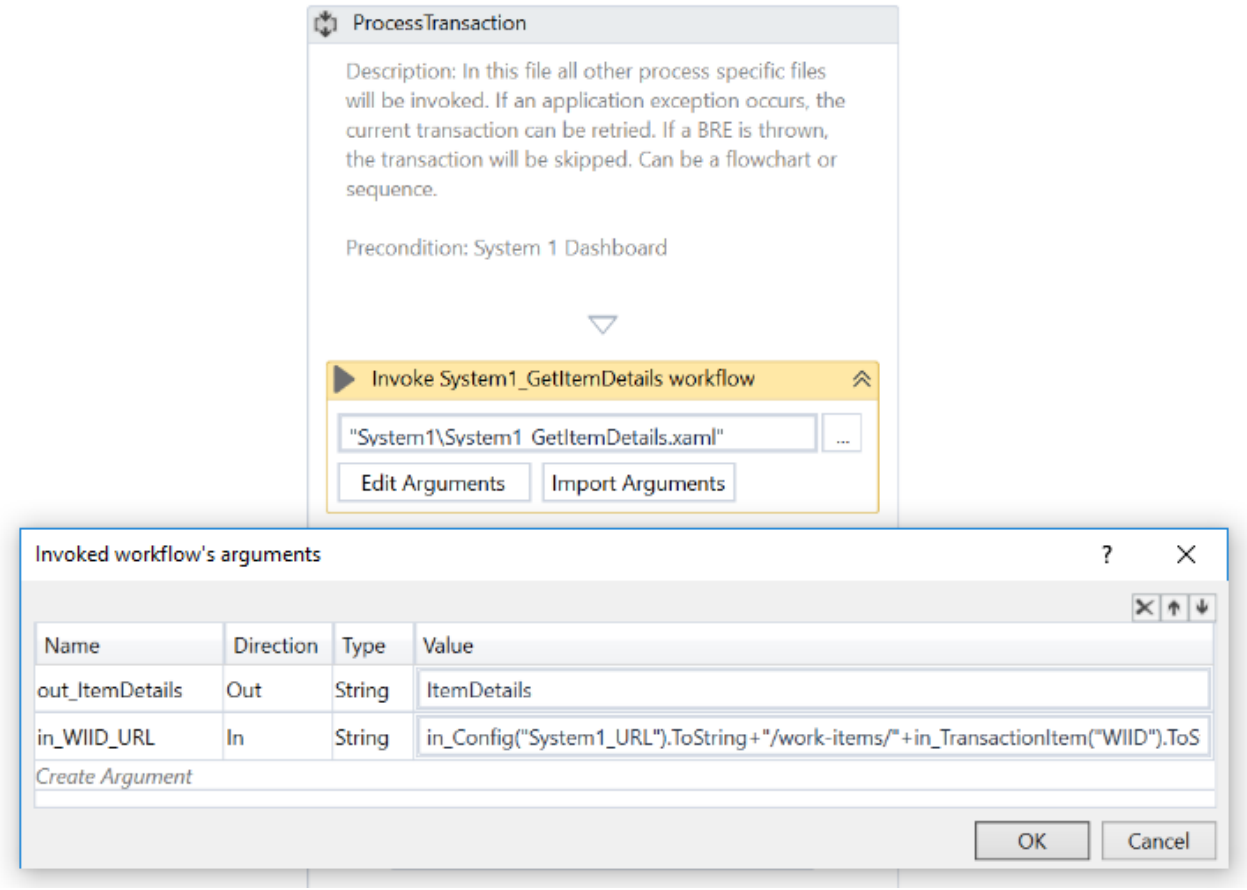
- No hay argumentos de entrada en este flujo de trabajo. Solo hay una condición previa: la página Work Item Details debe estar abierta.
- Necesitamos 3 argumentos de salida: ID de cliente, nombre de cliente y país del cliente.
- Antes de obtener el texto de la página web, tenemos que asegurarnos de que la página está cargada, y todos los elementos están disponibles. Añada una actividad **Attach Browser**. En el apartado **Do** utilizaremos una actividad **Get Text** con la propiedad **WaitForReady** definida como **Complete**.
- El flujo de trabajo debería quedar así:



- En el mismo flujo de trabajo debemos extraer el valor del ID de cliente, nombre de cliente y país del cliente.

- Añada una actividad **Assign**. Utilice el argumento de salida para ID de cliente en el campo **To**.
- En el campo **Value** utilizaremos los métodos Substring y Split para extraer únicamente la información necesaria. La expresión resultante es ClientInformation.Substring(ClientInformation.IndexOf("ID de cliente: ")+"ID de cliente: ".Length).Split(Environment.NewLine.ToCharArray)(0)
- Añada dos actividades **Assign** más para nombre de cliente y país del cliente. El parámetro introducido en el campo **Value** debe ser similar al utilizado en el argumento ID de cliente.
- Después de ello, se recomienda escribir los valores extraídos en el panel de salida con la actividad **Write Line**; esto puede ayudarnos a depurar nuestro flujo de trabajo más adelante.

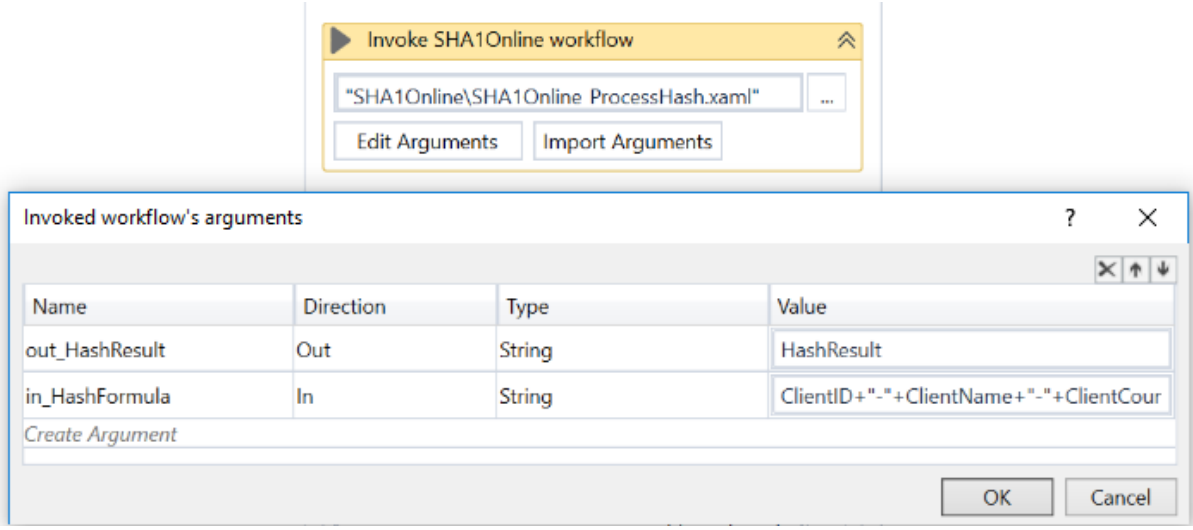
- Abra el flujo de trabajo **Process**.
  - Edite la nota.
  - Haga una llamada al flujo de trabajo **System1_NavigateTo_WIDetails**. Importe y enlace los argumentos.
  - Debería quedar así:



  - Haga una llamada al flujo de trabajo **System1_ExtractClientInformation**. Enlace los tres argumentos de salida a las variables locales.
- Ahora crearemos un flujo de trabajo para recuperar el valor del código *hash* de la aplicación SHA1Online.com. Cree un flujo de trabajo con secuencia en blanco en la carpeta SHA1Online. Póngale el nombre siguiente: SHA1Online_GetHashCode.
  - Necesitamos un argumento de entrada de tipo String. Su valor es el del código *hash* que hemos obtenido.
  - También necesitamos un argumento de salida de tipo String para el código *hash* calculado.
  - Añada una actividad **Type Into** y configúrela para introducir el argumento de entrada en el campo *hash* de la aplicación System1. De forma opcional, en el panel **Properties**, marque la casilla del campo **Simulate Type** si desea que esta actividad trabaje en segundo plano con precisión.
  - Añada una actividad **Click** para seleccionar el botón ***hash***. Alternativamente, vaya al panel **Properties** para marcar la casilla del campo **Simulate Click** si desea que esta actividad trabaje en segundo plano con precisión.

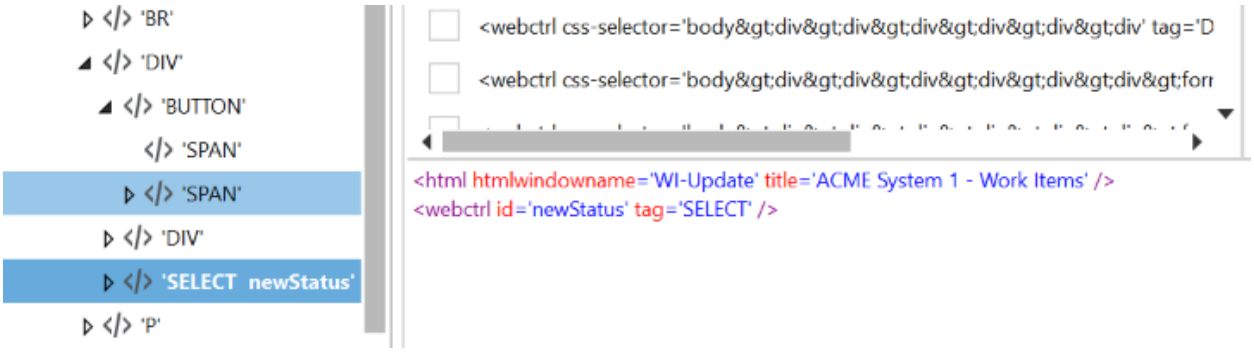
- Ahora podemos obtener el resultado utilizando una actividad Get Full Text; utilice el argumento de salida para guardar el texto de salida.
- Finalmente, tenemos que volver a la página de inicio de la aplicación para poder utilizar la misma secuencia para procesar el siguiente elemento. Para ello, añada una actividad **Go Back** para volver a la página de inicio.

- Recuerde que debemos calcular el código *hash* para [IDcliente]-[NombreCliente]-[PaísCliente], por lo que debemos componer dicha cadena. Utilizaremos el valor de salida del flujo de trabajo **System1_ExtractClientInformation** como argumento de entrada en el flujo de trabajo **SHA1Online_GetHashCode**.
- Vuelva al flujo de trabajo Process.
  - Añada una actividad **Invoke Workflow** y seleccione SHA1Online_GetHashCode.
  - Importe los argumentos. Use la fórmula de *hash* como argumento de entrada. Cree una variable para guardar el *hash* resultante.
  - El flujo de trabajo debería quedar así:



- Cree un flujo de trabajo con secuencia en blanco en la carpeta System1 para actualizar los elementos de trabajo con el código *hash* calculado. Es mejor crear un flujo de trabajo genérico que se pueda reutilizar en futuros proyectos. Para actualizar los elementos de trabajo solo hay que añadir un comentario, fijar el nuevo estado y enviar los cambios.
  - Inicie la nueva secuencia añadiendo una nota. La condición previa es que la página Work Item Details esté abierta para poder actualizar el elemento de trabajo.
  - Añada dos argumentos de entrada de tipo String, uno para el comentario y otro para el nuevo estado.

- Añada una actividad **Click** y establezca el botón **Update Work Item** como objetivo. En el panel **Properties**, marque la casilla del campo Simulate Click .
- Utilice una actividad **Type Into** para rellenar el campo de comentarios de la página **Update Work Item**. Además, seleccione la casilla **Simulate Type** en el panel **Properties**.
- Ahora debemos actualizar el estado de este elemento de trabajo. Para ello, arrastre y suelte una actividad **Select Item**. Haga clic en **Indicate on Screen** y seleccione el cuadro desplegable en el campo **New Status**. Probablemente aparecerá un mensaje de error diciendo que ese control no es compatible con el elemento seleccionado. Esto ocurre porque tenemos otros elementos UI por encima de la entrada Select.
    - Abra **UiExplorer** y haga clic en Select Target Element. El elemento rechazado puede ser de tipo Button o de tipo Span UI Element, según donde haya hecho clic.
    - Haga clic en el campo **New Status**. Seleccione un elemento debajo de Button o Span en el panel en árbol y utilice el selector generado para la actividad Select Item.
    - Este es el aspecto que debería tener en el UiExplorer:



    - Para terminar, defina la propiedad del elemento en el argumento in_Status.
- Añada una actividad **Click** para el botón Update Work Item. Asegúrese de que esté activada la opción **Simulate Click**.
- Aparecerá un mensaje de confirmación, por lo que deberemos utilizar otra actividad **Click** para seleccionar el botón OK. La opción **Simulate Click** también debe estar activada.
- Ahora ya puede cerrar la ventana Update Work Item haciendo clic en el botón de cierre situado en la esquina superior derecha.

- Vuelva al flujo de trabajo Process y haga una llamada al flujo de trabajo recién creado. Asegúrese de utilizar las variables correctas como valores de los argumentos de entrada.
- Para concluir debemos dejar la aplicación en su estado inicial para poder procesar el siguiente elemento. Para ello, haga una llamada al flujo de trabajo creado para ir hasta la página del panel de control.
- Ya hemos concluido la implementación del proceso. A continuación debemos probar todo el proceso. Ya debería haber probado todos los flujos de trabajo por separado después de haberlos creado utilizando valores predeterminados para los argumentos.
    - Ejecute el flujo de trabajo Main varias veces para comprobar que se ejecute correctamente en todos los casos. De lo contrario, corrija los errores y vuelva a ejecutarlo.
    - Use la opción **Reset test data** en el menú **User options** para generar un nuevo conjunto de datos para la prueba.
    - Compruebe la función de reintentar. Defina el parámetro MaxRetryNumber en Config.xlsx como 1 e interfiera con el robot haciendo clic en un menú de vínculos diferente, p. ej. el enlace a la página de inicio justo después de que el robot abra el elemento de trabajo actual. De este modo, el botón Update Work Item no estará disponible y se emitirá una excepción de sistema al no encontrar el elemento UI. El proceso se reinicia y se reanuda la ejecución del elemento de trabajo fallido. Si vuelve a interferir, el proceso se detendrá al alcanzar el número máximo de intentos.

Notas de implementación del proceso

Comenzamos por recuperar la lista de todos los elementos por procesar utilizando el asistente Data Scraping. Pregúntese lo siguiente: «¿Qué pasa si tiene un conjunto grande de transacciones y la actividad de extracción de datos en el estado Init falla debido a que se excede el tiempo de espera del navegador? ¿Cómo podríamos mejorar el diseño para mejorar la gestión de errores?»

- Procesamos los elementos de uno en uno. Todos los elementos son independientes entre sí, por lo que también podemos procesarlos en paralelo utilizando varios robots. En el siguiente ejercicio veremos cómo se utiliza la función Orchestrator Queues para crear un proceso donde los elementos de trabajo se distribuyan entre varios robots y se procesen en un solo paso, en paralelo.