



Capítulo 9: Canales

Bienvenidos a este capítulo sobre la implementación de canales en BIZUIT. A lo largo de esta serie, nos sumergiremos en las herramientas que BIZUIT nos brinda para integrar sistemas, automatizar tareas y garantizar una conectividad eficiente entre aplicaciones. Estos conocimientos no solo nos permitirán optimizar procesos, sino que también nos ayudarán a reducir la carga de trabajo repetitivo y mejorar la eficiencia general de nuestras operaciones.

En el mundo empresarial, la necesidad de que los sistemas se comuniquen entre sí es cada vez más crucial. BIZUIT nos permite conectar aplicaciones y sistemas distintos mediante lo que denominamos canales. Estos funcionan como puntos de conexión que gestionan la transferencia de datos y la ejecución de comandos sin intervención manual.

Pensemos, por ejemplo, en un sistema de atención al cliente: con los canales de BIZUIT, podemos recibir, procesar y responder solicitudes de manera automática, mejorando la experiencia del cliente y optimizando el tiempo de nuestro equipo.

Estructura del Capítulo

Este capítulo se divide en cuatro unidades principales:

1. **Introducción a los canales en BIZUIT:** En esta primera unidad, exploraremos los conceptos básicos y la importancia de los canales en el contexto de la automatización.
2. **Tipos de canales y casos de uso:** En la segunda unidad, profundizaremos en los diferentes tipos de canales disponibles en BIZUIT y analizaremos ejemplos concretos de su aplicación.
3. **Configuración práctica:** En la tercera unidad, nos enfocaremos en la configuración de cada tipo de canal con ejemplos prácticos que nos ayudarán a implementar lo aprendido.
4. **Mejores prácticas y optimización:** Finalmente, en la cuarta unidad, revisaremos las mejores prácticas para asegurar que nuestros canales sean seguros, confiables y optimizados.

A lo largo de este recorrido, no solo adquiriremos conocimientos teóricos, sino que también desarrollaremos la capacidad de aplicarlos en nuestras propias organizaciones.



Audiencia Ideal

Esta clase está diseñada para desarrolladores de software que buscan implementar integraciones complejas en BIZUIT, automatizar procesos y conectar sistemas externos, así como para arquitectos de tecnología interesados en maximizar la interoperabilidad y conectividad de BIZUIT con otros sistemas empresariales mediante APIs, servicios web y otros protocolos.

Objetivos

1. Entender el concepto y propósito de los canales en BIZUIT: Conocer cómo funcionan los canales como herramientas de integración y automatización dentro de la plataforma.
2. Configurar y utilizar los diferentes tipos de canales: Aprender a configurar y gestionar cada tipo de canal disponible en BIZUIT, como los Programadores de Procesos, Monitoreo de Archivos, Servidores TCP, y Servicios Web, entre otros.
3. Implementar integraciones con sistemas externos: Saber cómo utilizar los canales para conectar BIZUIT con otros sistemas y servicios externos, facilitando la interoperabilidad de la plataforma.
4. Automatizar flujos de trabajo y procesos: Usar los canales para ejecutar procesos de BIZUIT de manera automatizada, optimizando tiempos y mejorando la eficiencia operativa.
5. Aplicar buenas prácticas de seguridad y monitoreo: Configurar los canales de forma segura y aprender a monitorear su actividad para mantener la integridad y el rendimiento del sistema.

Unidad 1: Introducción a los Canales en BIZUIT

Los canales en BIZUIT son conexiones automatizadas entre diferentes sistemas o aplicaciones que nos permiten intercambiar información en tiempo real o a intervalos definidos. Imaginemos que contamos con un sistema de inventario en una plataforma y un sistema de ventas en otra.

Sin canales, la comunicación entre ellos sería compleja y manual; con los canales, logramos una actualización instantánea de inventarios cada vez que se realiza una venta, evitando errores y mejorando la precisión de nuestros datos.

Beneficios de utilizar canales en BIZUIT

¿Qué ganamos al implementar estos canales? Los beneficios pueden resumirse en tres pilares fundamentales:

- **Automatización:** Reducimos las tareas manuales y permitimos que nuestro equipo se enfoque en actividades de mayor valor.
- **Conectividad:** Al enlazar sistemas que, de otro modo, no podrían comunicarse, creamos un flujo continuo de información que mejora la toma de decisiones.
- **Escalabilidad:** Los canales en BIZUIT pueden adaptarse a cambios en la carga de trabajo y a la expansión de nuestro negocio. Esto significa que, sin importar cuánto crezcamos o evolucionemos, los canales se ajustarán a nuestras necesidades.

Ejemplos de escenarios donde los canales agregan valor

Veamos algunos ejemplos concretos en distintos sectores:

- **Sector salud:** En un hospital, los pedidos de estudios médicos deben gestionarse con rapidez. A través de un canal de monitoreo de archivos, BIZUIT puede detectar y cargar automáticamente estos pedidos en el sistema de gestión, asegurando que el equipo médico tenga acceso inmediato a la información.
- **Transporte y logística:** En una empresa de transporte, la gestión de entregas puede optimizarse con un canal que monitoree las cuentas de correo. De esta manera, las solicitudes de servicio se procesan automáticamente y se asignan a las unidades correspondientes sin intervención manual.

Estos ejemplos reflejan cómo los canales no solo optimizan nuestros procesos, sino que también nos permiten escalar de manera eficiente, asegurando que nuestra operación se mantenga ágil y adaptativa.



Unidad 2: Introducción a los Tipos de Canales

En esta unidad, exploraremos en detalle los diferentes tipos de canales que BIZUIT nos ofrece. Cada uno de ellos tiene un propósito específico y nos permite interactuar con distintos sistemas y fuentes de datos.

Analizaremos cómo estos canales facilitan la automatización y mejoran la integración de procesos dentro de nuestras organizaciones.

A medida que avancemos, veremos que no solo son versátiles, sino que también resultan fundamentales para construir una infraestructura empresarial conectada y eficiente.

API Rest

El canal API Rest nos permite exponer funcionalidades de BIZUIT como servicios accesibles a través de peticiones HTTP. Esto resulta clave para conectar BIZUIT con aplicaciones externas que necesiten consultar o enviar datos en tiempo real.

Pensemos en un sistema de inventario externo que requiere conocer el stock de productos en BIZUIT. Al configurar un canal API Rest, podemos exponer esta información y asegurarnos de que cada vez que se haga una solicitud, el sistema de inventario obtenga los datos actualizados.

Este tipo de canal nos permite integrar BIZUIT de manera ágil con aplicaciones modernas, especialmente aquellas que dependen de microservicios o arquitecturas *serverless*, facilitando la interoperabilidad.

Servicios Web

Los servicios web nos permiten ejecutar procesos en BIZUIT cuando se invoca un servicio web, utilizando el protocolo SOAP. Esta opción es ideal para integrar sistemas de manera remota y garantizar una comunicación fluida entre plataformas.

Por ejemplo, si una aplicación externa necesita consultar el estado de un pedido en BIZUIT, podemos exponer esta funcionalidad a través de un servicio web. De esta forma, cada consulta se procesa automáticamente en tiempo real.

Gracias a los servicios web, logramos una integración flexible con aplicaciones de terceros, algo esencial en entornos empresariales donde múltiples sistemas deben interactuar.



Programadores de Procesos

Los programadores de procesos son canales diseñados para ejecutar tareas de forma automática y en intervalos regulares. Son especialmente útiles para actividades repetitivas que deben realizarse en horarios específicos o ciclos predefinidos.

Imaginemos que debemos generar un reporte de ventas diariamente. Con un programador de procesos, podemos configurarlo para que este informe se genere automáticamente a medianoche y se envíe al equipo directivo sin necesidad de intervención manual.

Esto nos asegura que las tareas críticas se realicen a tiempo y sin depender de la disponibilidad del personal, reduciendo el riesgo de retrasos y errores.

Monitoreo de Archivos

El canal de monitoreo de archivos nos permite supervisar carpetas específicas en el sistema operativo o en servidores FTP y ejecutar procesos automáticamente cuando se detecta un nuevo archivo o un cambio en los existentes.

Tomemos como ejemplo una empresa de logística, donde los proveedores suben documentos de inventario a una carpeta FTP. Al implementar un canal de monitoreo de archivos, BIZUIT puede detectar cada nuevo documento, procesarlo y actualizar automáticamente la base de datos de inventarios.

Esto elimina la necesidad de intervención manual y garantiza que la información se mantenga actualizada en tiempo real, algo crucial en operaciones donde la velocidad y precisión son determinantes.

Monitoreo de Cuentas IMAP

Este canal permite a BIZUIT monitorear cuentas de correo IMAP y ejecutar procesos automáticamente cada vez que se recibe un nuevo mensaje. Es ideal para empresas que gestionan solicitudes o reclamos a través del correo electrónico.

Por ejemplo, en un centro de atención al cliente, cada vez que se recibe un correo en la cuenta de soporte, BIZUIT puede analizar su contenido, asignarlo a un agente y registrar el caso en el sistema de gestión. Automatizar el procesamiento de correos no solo ahorra tiempo al equipo de soporte, sino que también mejora la rapidez de respuesta y la satisfacción de los clientes.



Monitoreo de Mensajes Kafka

Kafka es una plataforma de mensajería para la transmisión de datos en tiempo real. Con BIZUIT, podemos configurar un canal Kafka para recibir mensajes y ejecutar procesos en respuesta a ellos.

Por ejemplo, en una empresa que maneja ventas en múltiples tiendas, cada vez que se recibe un mensaje de venta en el canal Kafka, BIZUIT puede actualizar la base de datos y enviar alertas en caso de bajo stock.

Este tipo de integración es ideal para entornos donde el flujo de datos es constante y de gran volumen, permitiéndonos gestionar la información de manera eficiente y con alta escalabilidad.

Administradores de Colas de Mensajes

Los administradores de colas de mensajes nos permiten gestionar el encolado de mensajes antes de enviarlos a BIZUIT, asegurando que cada mensaje sea procesado, incluso si hay fallos temporales en el sistema.

Imaginemos que en una tienda online debemos gestionar los pedidos de los clientes. Con un administrador de colas, cada pedido se almacena antes de ser procesado en BIZUIT, evitando que se pierda en caso de una caída del sistema.

Este canal garantiza una gestión segura y controlada de los mensajes, optimizando el flujo de trabajo y asegurando que ninguna información importante se extravíe.

Servidores TCP

El canal de Servidores TCP en BIZUIT nos permite exponer un servidor TCP para que otras aplicaciones puedan enviar mensajes directamente. Esto es especialmente útil en entornos de producción donde la comunicación en tiempo real es crítica.

Pensemos en un hospital donde BIZUIT actúa como servidor TCP para recibir datos de dispositivos médicos, como monitores de signos vitales en la UCI.

Cada vez que uno de estos dispositivos detecta un valor fuera del rango normal, envía un mensaje a BIZUIT, que inmediatamente activa un proceso de alerta.



Esto puede incluir el envío de notificaciones al personal médico, el registro del evento en el sistema de historia clínica y la activación de alarmas visuales en la sala de control.

Gracias a este canal, aseguramos una monitorización continua y una intervención médica rápida, mejorando la seguridad del paciente y optimizando la respuesta ante emergencias.



Unidad 3: Configuración de Canales en BIZUIT

En esta unidad, exploraremos la configuración de los diferentes tipos de canales en BIZUIT, herramientas fundamentales para la integración y automatización de procesos dentro del sistema. Los canales permiten establecer conexiones con aplicaciones externas, exponer servicios como API REST, programar tareas recurrentes, monitorear archivos o correos electrónicos, e incluso interactuar con colas de mensajes y servidores TCP.

Detallaremos paso a paso cómo configurar un canal API REST, una de las formas más flexibles y eficientes de exponer funcionalidades de BIZUIT a otros sistemas. Veremos cómo definir el endpoint, establecer parámetros de entrada y salida, configurar opciones de seguridad y realizar pruebas de conexión utilizando herramientas como Postman. Además, aprenderemos sobre otros canales clave, como los canales de Servicio Web, ideales para integraciones con sistemas basados en SOAP, y los programadores de procesos, que permiten la ejecución automática de tareas en intervalos predefinidos.

Por último, abordaremos la configuración de canales más avanzados, como los de monitoreo de archivos y correos IMAP, que facilitan la automatización de flujos de trabajo en función de la llegada de nuevos datos, y los canales TCP y Kafka, que permiten el procesamiento de mensajes en tiempo real desde dispositivos y aplicaciones externas.

Con esta unidad, no solo comprenderás cómo configurar cada tipo de canal, sino que también aprenderás a optimizar su uso para garantizar integraciones robustas y seguras en cualquier entorno de negocio.

Configuración de un Canal API REST

A continuación configuraremos juntos un canal API REST en BIZUIT, paso a paso. Este canal nos permitirá exponer funcionalidades de BIZUIT como servicios REST, permitiendo que otras aplicaciones accedan a la información en tiempo real. En este caso, configuraremos un canal llamado *Empleado Habilitado a Reembolso*, que validará si un empleado está habilitado para un reembolso según su número de empleado.

Definición del canal y su propósito

Comenzamos asignando un nombre al canal. En este caso, hemos decidido llamarlo *Empleado Habilitado Reembolso*. Este nombre será clave para identificar el canal en nuestro sistema y recordar su propósito de manera clara.

Es recomendable incluir una descripción detallada para que otros usuarios o administradores comprendan su uso.

En nuestro caso, podemos describirlo como: *Valida si un empleado está habilitado para reembolso según su número de empleado*. Agregar una descripción detallada nos ayudará a mantener la claridad y organización en proyectos de gran escala.

Configuración del Endpoint

A continuación, definimos el nombre del endpoint, que será el identificador único de nuestra API dentro de BIZUIT. Para este ejemplo, utilizamos el mismo nombre: *Empleado Habilitado a Reembolso*. Es importante que este nombre sea lo suficientemente descriptivo para reflejar claramente la función que cumple.

Luego, seleccionamos el proceso que se ejecutará cuando se invoque este endpoint. En este caso, elegimos *Consulta Habilitación Reembolso*, el cual es el proceso específico en BIZUIT que verificará si el empleado está habilitado para recibir un reembolso.

Modo de REST API y Parámetros de entrada

En este paso, debemos elegir entre dos opciones de modo de REST API:

- **Endpoint Function:** Expone una función específica como un endpoint independiente.
- **BIZUIT SDK API:** Permite una integración más profunda con el SDK de BIZUIT, facilitando el inicio, la continuación o la eliminación de procesos.

Para nuestro caso, utilizaremos la opción Endpoint Function.



Ahora, definimos los parámetros de entrada que requiere nuestro endpoint. En este caso, el proceso seleccionado contiene solo un parámetro de entrada: *Número de Empleado*. Si hubiera más parámetros opcionales, podríamos incluirlos según nuestra necesidad.

Parámetros
string NroEmpleado

Definición de parámetros de salida

Los parámetros de salida son los datos que la API devolverá tras ejecutar el proceso. En este caso, tenemos un solo parámetro de salida: *ReembolsoDisponible*, que indica si el empleado es elegible para reembolso. Si el proceso tuviera múltiples parámetros de salida, podríamos seleccionarlos según nuestras necesidades.

Parámetros de salida
 ReembolsoDisponible

Es fundamental que los parámetros de salida sean claros y relevantes para quienes consuman el servicio, asegurando así una integración efectiva y comprensible.

Opciones de seguridad y activación del canal

Aquí encontramos la opción de *Permitir Ejecución Anónima*. Si activamos esta opción, cualquier usuario podrá acceder al endpoint sin autenticación. Esto puede ser útil en ciertos

casos, pero debemos evaluar cuidadosamente los riesgos de seguridad. En este ejemplo, activaremos la opción para facilitar las pruebas del endpoint.



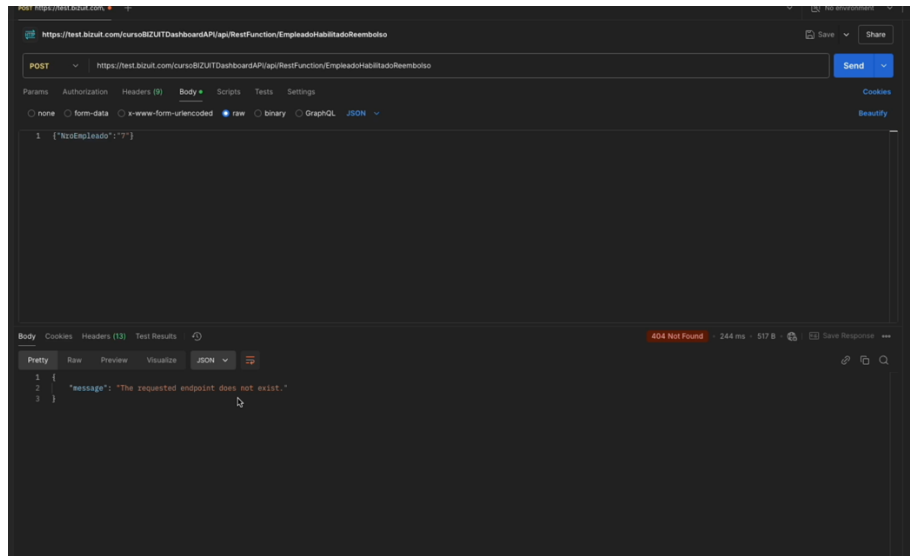
Una vez configurados el nombre del canal y el endpoint, se genera automáticamente una URL de acceso a la API. Esta dirección será la que otros sistemas utilizarán para conectarse y consultar la API.

URL: `'https://test.bizuit.com/cursoBIZUITDashboardAPI/api/RestFunction/EmpleadoHabilitadoReembolso'`

Prueba del endpoint con Postman

Copiamos la URL generada y la probamos en Postman. Por defecto, los *endpoint functions* funcionan con el método POST. En la sección *Body*, seleccionamos *raw.json* y copiamos la estructura del contenido.

Si ingresamos un número de empleado, por ejemplo 7, y el endpoint aún no está guardado, nos devolverá un error indicando que el endpoint no existe. Esto ocurre porque aún no lo hemos activado.



Luego de guardar y habilitar el endpoint, si la solicitud no cumple con el formato esperado, la API nos devolverá un error indicando que el parámetro *Número de Empleado* es requerido.

Opciones avanzadas con BIZUIT SDK API

Si optamos por utilizar el canal API REST como BIZUIT SDK API, podemos definir diferentes métodos para obtener y manipular datos del proceso:

- **GET:** Obtiene los datos de una instancia, enviando el *Instance ID* en el *Query String* o en el *Body*.
- **POST:** Inicia un proceso, enviando los datos en el *Query String* o en el *Body*.
- **PUT:** Permite continuar un proceso, indicando el *Instance ID* y el número de empleado a modificar.
- **DELETE:** Elimina una instancia, permitiendo elegir entre enviarla en el *Query String* o en el *Body*.

Finalización y pruebas del canal

Una vez configurados todos los parámetros y opciones del canal API REST, presionamos Guardar para finalizar la configuración. No debemos olvidar activar el canal para que se pueda acceder al endpoint desde la URL proporcionada.

Es recomendable realizar una prueba del endpoint después de guardarlo, para verificar que funciona correctamente.

Con estos pasos, hemos completado la configuración de un canal API REST en BIZUIT. Este proceso nos permite exponer información de manera controlada y segura, facilitando la integración con otros sistemas.

En este caso, nuestro endpoint valida si un empleado está habilitado para recibir un reembolso. Es importante realizar pruebas y ajustar la configuración según las necesidades de cada caso de uso.

Configuración de un Canal de Servicio Web

Ahora, aprenderemos a configurar un canal de Servicio Web en BIZUIT. Este tipo de canal nos permite exponer un proceso como un servicio web accesible mediante protocolos estándar como SOAP. Esto es especialmente útil cuando necesitamos que otros sistemas, especialmente aquellos más antiguos, puedan invocar métodos en BIZUIT y recibir datos o realizar acciones de forma controlada.

A continuación, veremos paso a paso cómo llevar a cabo esta configuración.

Creando Nuevo Canal Servicio Servicio Web

Canal *
WSEmpleadoHabilitadoReembolso

Descripción del Canal
Valida si un empleado está habilitado para reembolso según su número de empleado via WS

General Configuración general

Proceso *
ConsultaHabilitacionReembolso

Servicio Web *
WSCurso01

Tipos de Publicación
ASMX Web Service

Metodos Configuración de métodos

Nombre de método
ConsultarEmpleadoHabilitado

Prefijo

Incluir parámetros opcionales Usar autenticación por defecto

Parámetros de entrada
string NroEmpleado, string UserName, string Password

Parámetros de salida seleccionados
ReembolsoDisponible

Tipo retorno
string

Paso 1: Configuración General del Canal

Nombre del Canal

Comenzamos asignando un nombre a nuestro canal de Servicio Web. Este nombre nos permitirá identificar el canal dentro de BIZUIT y diferenciarlo de otros servicios. Es importante que el nombre sea descriptivo y refleje con claridad el objetivo del servicio que estamos exponiendo.

Descripción del Canal

A continuación, agregamos una descripción detallada para indicar el propósito de este canal. Una buena descripción facilita que otros usuarios comprendan rápidamente para qué se utiliza.

Selección del Proceso

Después, seleccionamos el proceso en BIZUIT que deseamos exponer a través de este canal. En nuestro caso, elegiremos el proceso ConsultaHabilitacionReembolso.

General Configuración general

Proceso *
ConsultaHabilitacionReembolso

Servicio Web y Tipo de Publicación

En esta sección, debemos seleccionar el Servicio Web existente al cual agregaremos el método que invocará el proceso, o bien, podemos crear un nuevo Webservice. En nuestro caso, crearemos uno nuevo.

Para ello, hacemos clic en "+", asignamos un nombre al servicio y seleccionamos el tipo de publicación, eligiendo entre tecnología WCF o ASMX. Para simplificar la configuración, utilizaremos ASMX.

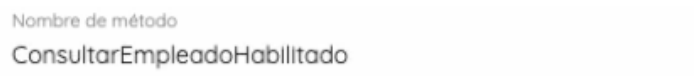


Paso 2: Configuración de Métodos

Nombre del Método

Aquí especificamos el nombre del método que expondremos a través del servicio web. Este nombre debe ser claro y descriptivo, ya que será el que otros sistemas utilizarán para invocar la función en BIZUIT.

Por ejemplo, un nombre adecuado podría ser ConsultarEmpleadoHabilitado, ya que describe con precisión la funcionalidad del método.



Prefijo de Parámetros

Podemos definir si los parámetros del método obtenidos desde los parámetros obligatorios del proceso llevarán o no un prefijo, como por ejemplo "p".

Parámetros Opcionales

Si nuestro proceso contara con parámetros opcionales, podríamos agregarlos al endpoint. Sin embargo, en este caso, nuestro proceso no los posee.

Autenticación por Defecto

Si activamos esta opción, no será necesario ingresar el usuario y contraseña en los parámetros del método, ya que el servicio utilizará las credenciales configuradas en las settings de BIZUIT.

Paso 3: Configuración de Parámetros de Salida

A diferencia del canal API REST, donde podemos seleccionar los parámetros de salida, en el canal de Servicio Web, BIZUIT seleccionará automáticamente el primer parámetro de salida encontrado en el proceso.

El sistema también nos indicará el tipo de retorno del parámetro de salida seleccionado, asegurando que la integración con otros sistemas sea clara y consistente.

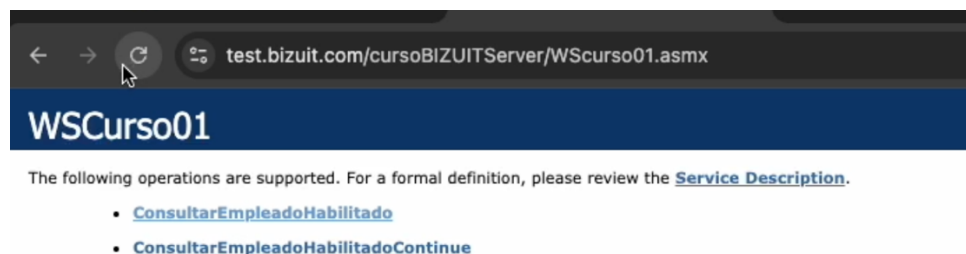
Paso 4: Guardar la Configuración y Realizar Pruebas

Una vez que hemos configurado todos los aspectos del canal y del método, presionamos "Guardar" para activar el servicio web.

Es fundamental realizar pruebas de invocación del servicio desde una aplicación externa para verificar que todos los parámetros de entrada y salida estén correctamente configurados y que el servicio responda de manera esperada.

Para ello, seguiremos estos pasos:

1. Navegamos a la URL de nuestro servidor de BIZUIT y agregamos el nombre del Webservice configurado.
 - o En nuestro caso: `https://[servidor]/WSCurso01.asmx`.
2. Comprobamos que el método que hemos creado está disponible.



Con estos pasos, hemos configurado un canal de Servicio Web en BIZUIT. Gracias a este canal, otros sistemas podrán interactuar con procesos en BIZUIT de manera segura y controlada. Es importante documentar cada detalle del método configurado, incluyendo sus parámetros y el tipo de retorno, para facilitar su uso y mantenimiento en el futuro.

Con esta base, podemos seguir avanzando en la configuración e integración de más servicios en nuestra infraestructura.

Configuración Canal Programadores de Procesos

En esta parte, aprenderemos a configurar un canal Programador de Procesos en BIZUIT. Este tipo de canal nos permite automatizar tareas que deben ejecutarse en horarios o intervalos específicos, sin intervención manual. Es ideal para gestionar tareas repetitivas como generación de reportes, envío de correos automatizados, actualización de datos o backups periódicos.

A lo largo de esta configuración, exploraremos cada paso en detalle.

Creando Nuevo Canal Programador de Procesos

Canal: ReporteCantidadClientes Descripción: Genera un mail diario de cant de clientes y lo envia al equipo directivo

Proceso: ConsultaPeriodicaClientes

Selección Frecuencia de Ejecución:

- Diario
- Semanal
- Mensual
- Anual
- Intervalos
- Otro

Selección día:

- Lunes
- Jueves
- Domingo
- Martes
- Viernes
- Miércoles
- Sábados

Selección hora: --:--

Guardar Cancelar

Paso 1: Configuración Básica del Canal

Nombre del Canal

Lo primero que hacemos es asignar un nombre al canal. Este nombre debe ser claro y descriptivo para que cualquier usuario pueda identificar rápidamente la función del programador.

Por ejemplo, si el canal se utiliza para generar mails diarios con la cantidad de clientes, un nombre adecuado podría ser "ReporteCantidadClientes".

Descripción del Canal

Luego, añadimos una descripción que detalle el propósito del canal.

Este campo es clave para brindar contexto y organización, especialmente cuando trabajamos con múltiples canales en un mismo proyecto.

Selección del Proceso

Después, seleccionamos el proceso en BIZUIT que deseamos automatizar. Es importante recordar que solo estarán disponibles los procesos que no requieran parámetros de entrada obligatorios, ya que este canal no puede enviar valores, solo ejecuta el proceso en los intervalos definidos.

En nuestro caso, utilizaremos el proceso `ConsultaPeriodicaCliente`, que no requiere parámetros y se encarga de:

1. Consultar la cantidad de clientes en el sistema.
2. Enviar un correo electrónico con esta información al equipo directivo.

Paso 2: Selección de la Frecuencia de Ejecución

Una de las decisiones más importantes es definir cada cuánto tiempo se ejecutará el proceso. BIZUIT nos ofrece varias opciones:

- **Diario:** Ejecuta el proceso todos los días a una hora determinada.
- **Semanal:** Permite elegir los días de la semana y la hora de ejecución.
- **Mensual:** Se ejecuta en un día específico de cada mes o en un ciclo determinado (por ejemplo, el día 10 de cada 2 meses).
- **Anual:** Se programa para ejecutarse una vez al año en una fecha y hora específicas.
- **Intervalos:** Ejecuta el proceso en intervalos de tiempo regulares, por ejemplo, cada 30 segundos.
- **Opción Personalizada:** Permite definir una ejecución única en una fecha y hora específicas.

En nuestro caso, configuraremos el canal para ejecutarse cada 30 segundos.

The screenshot shows a configuration window titled "Seleccione Frecuencia de Ejecución:". On the left, there are radio buttons for "Diario", "Semanal", "Mensual", "Anual", "Intervalos", and "Otro". The "Intervalos" option is selected, indicated by a yellow dot. On the right, there are three input fields: "Horas" with a value of 0, "Minutos" with a value of 0, and "Segundos" with a value of 30. A yellow checkmark is visible next to the "Segundos" field.



Paso 3: Revisión y Guardado de la Configuración

Antes de finalizar, revisamos toda la configuración para asegurarnos de que:

- El proceso seleccionado es el correcto.
- La frecuencia y la hora de ejecución coinciden con nuestras necesidades.
- La descripción es clara para otros usuarios o administradores.

Este paso es clave para evitar errores y garantizar que el canal cumpla su función correctamente.

Paso 4: Guardar y Activar el Canal

Finalmente, presionamos "Guardar" para activar el canal. A partir de este momento, el programador ejecutará automáticamente el proceso según la frecuencia y el horario configurados.

Es recomendable monitorear el canal después de su primera ejecución para verificar que funcione correctamente y genere los resultados esperados.

Con estos pasos, hemos configurado un canal Programador de Procesos en BIZUIT. Este tipo de canal es esencial para la automatización de tareas periódicas, permitiéndonos mantener procesos críticos en funcionamiento sin intervención manual.

Ya sea para generar reportes, realizar actualizaciones o ejecutar backups, los Programadores de Procesos en BIZUIT nos ayudan a garantizar que estas tareas se ejecuten de manera puntual y eficiente.

Configuración Canal Monitoreo de Archivos

A continuación, aprenderemos a configurar un canal de Monitoreo de Archivos en BIZUIT. Este canal nos permite automatizar procesos basados en la detección de nuevos archivos en carpetas específicas, ya sean locales, compartidas o ubicadas en servidores FTP.

Gracias a esta funcionalidad, BIZUIT detecta automáticamente nuevos archivos y ejecuta un proceso en respuesta, optimizando la integración de datos y reduciendo la intervención manual. A continuación, veremos cada paso en detalle.

Paso 1: Configuración Básica del Canal

Nombre y Descripción del Canal

Lo primero que hacemos es asignar un nombre al canal, que debe ser lo suficientemente descriptivo para que cualquier usuario pueda identificar su función de manera rápida.

Por ejemplo, si el canal monitorea pedidos en un sistema de archivos, podríamos nombrarlo "MonitoreoPedidosFileSystem".

Luego, agregamos una descripción detallada, indicando el propósito del canal. Un buen ejemplo sería:

"Monitorea la carpeta compartida de pedidos y procesa cada archivo nuevo."

Esto facilitará la gestión del canal y ayudará a otros usuarios a comprender rápidamente su función.

Paso 2: Configuración de Carpetas



Carpeta de Escucha

En esta sección, configuramos la carpeta donde BIZUIT monitoreará la llegada de nuevos archivos.

Si trabajamos con FTP, debemos habilitar la opción correspondiente y, si el servidor no admite acceso anónimo, ingresamos el usuario y la contraseña.

Si la carpeta está en una red compartida que requiere credenciales, también debemos especificarlas. En caso de que utilicemos FTPS, activamos la opción SSL.

Usaremos la carpeta local C:\MonitoreoFS\Entrada, pero podríamos configurar una carpeta compartida o una ubicación en un servidor FTP.

Carpeta de Procesados

Definimos la carpeta donde BIZUIT moverá los archivos una vez procesados exitosamente.

Esto mantiene la Carpeta de Escucha ordenada y permite hacer seguimiento de los archivos ya gestionados.

Usaremos la carpeta C:\MonitoreoFS\OK.

Carpeta de Error

También establecemos una Carpeta de Error, donde se almacenarán los archivos que no pudieron procesarse correctamente debido a errores en el sistema o en el formato del archivo.

Esto nos permite realizar una revisión manual de los archivos problemáticos y tomar medidas correctivas.

Usaremos la carpeta C:\MonitoreoFS>Error.

Paso 3: Filtro de Archivos

Extensión del Archivo

Definimos un filtro para los tipos de archivo que queremos monitorear.

Por ejemplo, si solo queremos procesar archivos .txt o .csv, lo especificamos en esta sección.

Esto evita que se procesen archivos no deseados.

Extension *	Seleccione una Condición	Nombre *
.txt	Que comience	TEST

Condiciones Adicionales

Podemos agregar **criterios avanzados de selección**, como nombres de archivos específicos o patrones en los nombres, asegurando que **solo se procesen archivos relevantes**.

Paso 4: Selección del Proceso a Ejecutar

Elección del Proceso

Aquí seleccionamos el proceso en BIZUIT que se ejecutará automáticamente cuando se detecte un archivo en la Carpeta de Escucha.

Este proceso debe cumplir con ciertas condiciones:

- Debe tener un solo parámetro de entrada obligatorio y ser de tipo escalar.
- Si tiene parámetros opcionales, al menos uno debe ser escalar.

En nuestro caso, el proceso que usaremos tiene un parámetro obligatorio escalar y dos opcionales. Como el obligatorio es prioritario, los dos opcionales aparecen deshabilitados.

Este proceso envía un correo con el contenido del archivo recibido. Por lo tanto, seleccionamos ese parámetro obligatorio en nuestra configuración.

Parámetro de Envío: Contenido o Ruta del Archivo

Aquí elegimos si BIZUIT debe enviar el contenido completo del archivo o solo la ruta del archivo al proceso seleccionado.

- **Usar contenido:** BIZUIT lee el archivo completo y envía su contenido al proceso.
- **Usar ruta del archivo:** Solo se envía la ubicación del archivo, permitiendo que el proceso lo acceda directamente.

Si el archivo es grande, lo más eficiente es enviar solo la ruta para evitar sobrecarga de datos.

Codificación

Seleccionamos la codificación del archivo (por ejemplo, UTF-8), asegurando que se procesen correctamente caracteres especiales.

The screenshot shows a configuration window titled "Proceso a ejecutar" with a subtitle "Indique proceso y parámetro al cual se enviará el contenido del archivo o su ruta." The interface includes a dropdown menu for "Proceso" set to "MonitoreoPedidosFileSystem", a dropdown for "Parámetro de Proceso" set to "DatosPedido", and a dropdown for "Codificación" set to "UTF8". There is also a radio button labeled "Usar contenido" which is selected.



Paso 5: Configuración Avanzada

Intervalo de Monitoreo

En esta sección, definimos cada cuánto tiempo BIZUIT revisará la Carpeta de Escucha en busca de nuevos archivos.

Configuramos el monitoreo cada 5 segundos para una detección rápida, pero este intervalo puede ajustarse según las necesidades del sistema.

Paso 6: Guardar la Configuración y Realizar Pruebas

Una vez configurados todos los parámetros, revisamos la configuración y presionamos "Guardar" para activar el canal.

A partir de este momento, BIZUIT comenzará a monitorear la Carpeta de Escucha y ejecutará el proceso configurado cada vez que detecte un archivo que cumpla con los filtros establecidos.

Pruebas y Validación

- Colocamos un archivo en la Carpeta de Escucha con la extensión .txt y el prefijo TEST en el nombre.
- Verificamos que el archivo se mueve a la Carpeta de Procesados. Confirmamos que el proceso se ejecuta correctamente y envía un correo con el contenido del archivo.
- Cambiamos el contenido del archivo y lo guardamos nuevamente en la Carpeta de Escucha.
- Observamos que el nuevo archivo se procesa sin sobrescribir el anterior, ya que BIZUIT le asigna un identificador único.

Configuración Canal Monitoreo de Cuentas IMAP

En esta sección, aprenderemos a configurar un canal de Monitoreo de Cuentas IMAP en BIZUIT.

Este tipo de canal permite que BIZUIT monitoree una cuenta de correo electrónico, detecte los mensajes entrantes y ejecute un proceso en respuesta a cada nuevo correo recibido.

Es una herramienta ideal para automatizar la gestión de correos electrónicos, facilitando tareas como la recepción de solicitudes de soporte, órdenes de compra o cualquier comunicación entrante que requiera procesamiento automatizado

Paso 1: Configuración Básica del Canal

Nombre y Descripción del Canal

Lo primero que hacemos es asignar un nombre al canal, el cual debe ser descriptivo y fácil de identificar dentro de nuestro sistema.

Llamaremos a nuestro canal "CanalTicketSoporte", ya que su función será procesar correos de soporte. A continuación, agregamos una descripción detallada para indicar el propósito del canal.

Esta descripción facilitará la comprensión de la función del canal a otros usuarios o administradores.

Paso 2: Configuración de la Cuenta IMAP

Servidor IMAP y Puerto

Aquí debemos ingresar los datos del servidor IMAP y el puerto de conexión de la cuenta de correo que queremos monitorear.

Si usamos Gmail, el servidor será imap.gmail.com y el puerto 993 para conexiones seguras.

Opciones de Seguridad

Si el servidor IMAP requiere una conexión cifrada, activamos la opción "Usar SSL".

Esto garantiza la protección de los datos durante la transmisión, evitando accesos no autorizados.

Credenciales de la Cuenta

Aquí ingresamos el usuario y la contraseña de la cuenta de correo que BIZUIT monitoreará.

Formulario de credenciales de la cuenta. El campo "Usuario *" contiene el texto "labuser@tycon.com.ar". El campo "Contraseña *" contiene caracteres ocultos por asteriscos. Hay un ícono de lupa a la derecha del campo de contraseña.

Es importante que las credenciales sean correctas, ya que sin ellas BIZUIT no podrá acceder a los correos.

Google API Service (Opcional)

Si estamos configurando una cuenta de Google, podemos activar la opción "Usar Google API Service".



Esto tiene una ventaja que nos permite integrarnos directamente con los servicios de Google, ofreciendo una conexión más fluida y segura.

Paso 3: Configuración del Proceso a Ejecutar

Selección del Proceso

A continuación, elegimos el proceso en BIZUIT que se ejecutará cuando se reciba un nuevo correo. Este proceso puede realizar diversas acciones, como:

- Crear un ticket de soporte.
- Almacenar el correo en una base de datos.
- Enviar una respuesta automática.

Para que un proceso pueda ser seleccionado, debe cumplir con las siguientes condiciones:

- Debe tener entre 1 y 6 parámetros de entrada obligatorios de tipo escalar.
- Si no tiene parámetros obligatorios, debe contar con al menos un parámetro opcional de tipo escalar.

Formulario de selección del proceso a ejecutar. El campo "Proceso a ejecutar" muestra "SoporteTicket" y un ícono de flecha hacia abajo. El texto "Indique proceso y parámetros a los" está a la derecha.

Asignación de Parámetros del Correo

Aquí configuramos cómo se enviarán los diferentes elementos del correo al proceso en BIZUIT:

- **Remitente:** Se asigna a un parámetro específico del proceso
- **Asunto:** Se envía a un parámetro seleccionado.
- **Contenido del mensaje:** Se envía al proceso para su procesamiento.
- **Archivos adjuntos:** Podemos asignarlos a un parámetro del proceso.
- **Destinatarios en copia y copia oculta:** Se pueden capturar en parámetros separados.

The screenshot shows a configuration table with three columns. Each column has a header 'Enviar el [elemento] al parámetro' and a dropdown menu. The first column is for 'EmailCliente', the second for 'AsuntoTicket', and the third for 'DescripcionTicket'. Below each header, there are two rows of configuration options: 'Enviar los [elemento] al parámetro' and 'Enviar los destinatarios en copia [oculto] al parámetro'. The dropdown menus are currently set to 'Selecciona Parámetro'.

Esta configuración nos permite personalizar el flujo de datos según las necesidades del negocio.

Paso 4: Configuración Avanzada

The screenshot shows the 'Avanzadas' section of the configuration interface. It features a header 'Configuraciones avanzadas del canal.' and a setting 'Procesar correos recibidos desde hace: 0 días'. Below this, there is a toggle switch labeled 'Escribir en la Base de Datos' which is currently turned on.

Procesar Correos Recibidos Desde...

Aquí indicamos desde qué fecha queremos que BIZUIT procese los correos electrónicos. Si queremos incluir correos de los últimos 5 días, configuramos "Procesar correos recibidos desde hace 5 días".

Escribir en la Base de Datos

Si activamos esta opción, los datos de cada correo procesado se guardarán automáticamente en una base de datos.

Beneficio: Es útil para tener un registro histórico de los correos procesados, lo cual es ideal para gestión documental o análisis posterior.

Comprobación: En el visor de registros de BIZUIT podemos verificar que el canal está funcionando y que los datos se están almacenando correctamente.



Paso 5: Guardar la Configuración y Activar el Canal

Una vez configurados todos los parámetros, realizamos una revisión final y presionamos "Guardar" para activar el canal.

Desde este momento, BIZUIT comenzará a monitorear la cuenta de correo y ejecutará el proceso configurado cada vez que llegue un nuevo correo.

Paso 6: Pruebas del Canal

Para asegurarnos de que la configuración es correcta, realizamos una prueba:

1. Enviamos un correo electrónico a la cuenta configurada.
2. BIZUIT detecta el correo y ejecuta el proceso SoporteTicket.
3. El proceso recibe 1 parámetro obligatorio y 3 opcionales.
4. En la interfaz de BIZUIT, verificamos que se ha generado una nueva tarea en Interacción de Usuario.

En esta tarea veremos las siguientes columnas:

- **Mail:** Dirección del remitente.
- **Asunto:** Asunto del correo recibido.
- **Descripción:** Contenido del correo.

Si todo funciona como se espera, el canal está correctamente configurado.

Con estos pasos, hemos configurado exitosamente un canal de Monitoreo de Cuentas IMAP en BIZUIT. Esto nos permite automatizar la gestión de correos electrónicos, ejecutar procesos en respuesta a nuevos correos y facilita la integración con otros sistemas y bases de datos.

Como recomendación final podemos probar la configuración con correos de prueba, verificar que todos los parámetros están correctamente asignados y además monitorear el visor de logs para detectar posibles errores.

Este tipo de canal es esencial para la automatización de la gestión de correos, permitiendo a las empresas procesar mensajes de manera rápida y eficiente.

Configuración Canal Servidores TCP

Procederemos ahora a configurar un canal de Servidor TCP en BIZUIT. Este tipo de canal nos permite recibir mensajes directamente a través del protocolo TCP, ideal para integraciones en tiempo real en las que dispositivos o sistemas externos envían datos de manera continua.

Su utilidad es especialmente relevante en entornos de IoT, dispositivos médicos o comunicaciones industriales.

A lo largo de esta sección, exploraremos paso a paso el proceso de configuración del canal y su servidor.

The screenshot shows the 'Creando Nuevo Canal Servidor TCP' (Creating New TCP Server Channel) configuration form. The form includes the following fields and controls:

- Canal ***: ServidorTCP_IOT
- Descripción**: Recibe datos en tiempo real desde dispositivos IOT para procesamiento.
- Configuración**: Indique puerto, servidor, tipo de mensaje y proceso a ejecutar.
- Proceso ***: A dropdown menu.
- Dirección IP y Puerto del Host TCP ***: 127.0.0.5000, with 'Editar' and 'Eliminar' buttons.
- Parámetro de Entrada ***: Mensaje, with a dropdown arrow.
- Parámetro de Salida**: Respuesta, with a dropdown arrow.
- Es un mensaje HL7**: A toggle switch.
- Guardar Registro**: A yellow button.
- Mensaje HL7**: A text input field.
- Guardar** and **Cancelar** buttons at the bottom right.

Paso 1: Configuración del Canal

Nombre y Descripción del Canal

Primero, asignamos un nombre descriptivo al canal, como "ServidorTCP_DispositivosIoT". Este nombre nos ayudará a identificarlo dentro de BIZUIT. Luego, en la descripción, especificamos el propósito del canal, por ejemplo: "Recibe datos en tiempo real desde dispositivos IoT para procesamiento".

This partial screenshot shows the top section of the configuration form:

- Canal ***: ServidorTCP_IOT
- Descripción**: Recibe datos en tiempo real desde dispositivos IOT para procesamiento.

Paso 2: Selección del Proceso y Parámetros

Proceso

Seleccionamos el proceso que queremos que BIZUIT ejecute cada vez que el servidor reciba un mensaje. Este proceso debe estar previamente configurado y podría incluir cualquier

acción, como almacenar los datos en una base de datos o generar una alerta. Debemos tener en cuenta que solo estarán disponibles aquellos procesos que:

- Cuenten con uno y solo un parámetro de entrada escalar obligatorio, o
- Tengan al menos un parámetro de entrada escalar opcional y cualquier cantidad de parámetros escalares de salida.

Parámetro de Entrada y Salida

Configuramos el "Parámetro de Entrada" donde se almacenará el contenido del mensaje recibido para que el proceso pueda utilizarlo. Si es necesario, también establecemos un "Parámetro de Salida" para que el proceso pueda responder con un mensaje al remitente.

The screenshot shows a configuration window titled "Configuración" with the subtitle "Indique puerto, servidor, tipo de mensaje y proceso a ejecutar". It features two main sections: "Proceso" with a dropdown menu set to "TCPTest" and an "Agregar" button, and "Parámetro de Entrada" with a dropdown menu set to "Mensaje". To the right, there is a section for "Parámetro de Salida" with a dropdown menu set to "Respuesta".

Paso 3: Opciones Avanzadas del Mensaje

Si el mensaje que esperamos recibir está en formato HL7, activamos la opción "Es un mensaje HL7". Este formato es comúnmente utilizado en sistemas de salud y permite que BIZUIT interprete el mensaje adecuadamente.

Mensaje HL7

Si hemos activado la opción HL7, especificamos aquí el tipo de mensaje esperado. Esto ayuda a BIZUIT a interpretar correctamente la estructura y el contenido del mensaje.

Guardar Registro

Activamos "Guardar Registro" si queremos almacenar cada mensaje recibido en una base de datos para auditoría o análisis posterior. Esta opción resulta especialmente útil cuando necesitamos mantener un historial de los mensajes procesados.

Configuración del Servidor TCP

Una vez configurado el canal, pasamos a la configuración del servidor TCP, que será el encargado de recibir los mensajes. Aquí establecemos detalles técnicos como la IP, el puerto, tiempos de espera y ajustes de entrada/salida. Podemos seleccionar un servidor existente o crear uno desde cero.

Paso 4: Configuración del Host TCP

Creando Nuevo Host TCP

Dirección IP del Host *
127.0.0.0

Puerto Host *
5000

Tiempos Indique los tiempos.

Tiempo de Espera en la Recepción *
30000

Tiempo de Espera para la Respuesta *
30000

Ajustes de I/O Establezca los valores de envío y recepción.

Caracteres de Inicio de mensaje
0B

Caracteres de Inicio de respuesta
0B

Caracteres de fin de mensaje
1C,0D

Caracteres de fin de respuesta
1C,0D

Tamaño de Paquete en la recepción *
1024

Tamaño de Paquete en la respuesta *
1024

Otros Ajustes adicionales.

Linger Seconds *
60

Codificación *
ASCII

Dirección IP y Puerto del Host

Ingresamos la "Dirección IP del Host" y el "Puerto Host" en los que el servidor TCP estará escuchando. Por ejemplo, la IP podría ser la dirección interna de la red o una dirección pública, y el puerto debe estar abierto para recibir conexiones entrantes (por ejemplo, el puerto 5000).

Tiempo de Espera en la Recepción y para la Respuesta

Definimos el "Tiempo de Espera en la Recepción", que indica cuánto tiempo esperará el servidor para recibir un mensaje, y el "Tiempo de Espera para la Respuesta", que determina el tiempo que se esperará antes de enviar una respuesta. Estos valores se configuran en milisegundos; por ejemplo, 30000 milisegundos equivalen a 30 segundos.

Caracteres de Inicio y Fin de Mensaje

Definimos los "Caracteres de Inicio de Mensaje" y "Caracteres de Fin de Mensaje" que BIZUIT utilizará para delimitar cada mensaje recibido. Estos caracteres son esenciales para estructurar correctamente los datos, especialmente en comunicaciones continuas. Un ejemplo de configuración podría ser el carácter hexadecimal "0B" para el inicio y "1C,0D" para el final.

Tamaño de Paquete en la Recepción y en la Respuesta

Establecemos el "Tamaño de Paquete en la Recepción" y el "Tamaño de Paquete en la Respuesta". Estos valores determinan la cantidad de datos que el servidor TCP procesará en cada paquete. Un tamaño de 1024 bytes es comúnmente utilizado para manejar datos en bloques eficientes. La elección del tamaño dependerá de la longitud del mensaje.



Linger Seconds

El parámetro "Linger Seconds" define el tiempo adicional durante el cual el servidor TCP mantendrá abierta la conexión después de enviar una respuesta. Por ejemplo, si lo configuramos en 60 segundos, la conexión permanecerá abierta ese tiempo para facilitar comunicaciones recurrentes.

Cantidad Máxima de Mensajes Concurrentes

Especificamos el número máximo de mensajes que el servidor TCP puede manejar simultáneamente, así como el intervalo de tiempo para su procesamiento. Esto nos permite gestionar la carga del servidor y optimizar la eficiencia en la recepción de mensajes.

Codificación

Seleccionamos la codificación de los datos que el servidor TCP utilizará al recibir y enviar mensajes. Aunque ASCII es una opción común, podemos elegir otra codificación según las necesidades de compatibilidad con sistemas específicos.

Paso 5: Guardar la Configuración

Una vez que hemos completado todos los ajustes, verificamos que cada configuración sea correcta y presionamos "Guardar". Luego, seleccionamos el servidor de la lista y habilitamos el servicio de Canal Servidor TCP. A partir de este momento, BIZUIT comenzará a escuchar en la IP y puerto especificados, listo para recibir y procesar mensajes en tiempo real.

Con estos pasos, hemos configurado un canal Servidor TCP en BIZUIT, incluyendo los parámetros del canal y la configuración del host TCP. Este tipo de canal es fundamental para integraciones en tiempo real y permite que BIZUIT reciba y procese datos de sistemas externos de manera eficiente. Es recomendable realizar pruebas con datos simulados para asegurarnos de que todo funciona correctamente.

Configuración Canal Monitoreo de Mensajes Kafka

Ahora, configuraremos un canal de Monitoreo de Mensajes Kafka en BIZUIT. Este canal nos permitirá conectar BIZUIT a un sistema Kafka para consumir mensajes de un tema (topic) específico.

Es ideal para integrar flujos de datos en tiempo real y responder a eventos generados en múltiples aplicaciones o sistemas, como IoT, análisis de datos o procesamiento de eventos.

A continuación, configuraremos los aspectos básicos y avanzados de este canal.

Creando Nuevo Canal de Monitoreo de mensajes Kafka

Canal * CanalKafkaTopicoClientes Descripción Recibe Datos en tiempo real de sistemas de clientes en línea

Cuenta Kafka Indique los credenciales y configuraciones de la cuenta monitorear.

Broker Server * 192.168.100.222 Id del Grupo * CLIENTES AutoOffsetReset Earliest

Topic * Customers Usuario * adminkafka Contraseña * ****

Proceso a ejecutar Indique proceso que se ejecutara al consumir los mensajes de la cuenta.

Proceso * TCPTest El mensaje se cargara el parámetro de opcionel DataOpcional

Escribir en la Base de Datos

Avanzadas Configuraciones avanzadas del canal.

Guardar Cancelar

Nombre y Descripción del Canal

Primero, asignamos un nombre descriptivo al canal, como "CanalKafkaTopicoClientes". Este nombre nos ayudará a identificarlo dentro de BIZUIT.

Luego, en la descripción, especificamos el propósito del canal, por ejemplo: "Recibe datos en tiempo real desde sistemas de clientes en línea."

Paso 1: Configuración Básica de la Cuenta Kafka

Broker Server

Configuramos el "Broker Server", que es la dirección del servidor Kafka que queremos monitorear. Este broker será el punto de conexión de BIZUIT al sistema Kafka.

ID del Grupo

Especificamos el "ID del Grupo", lo que nos permitirá coordinar múltiples consumidores en la lectura de mensajes, asegurando que cada mensaje se procese solo una vez. Es importante definir este ID correctamente para que BIZUIT pueda identificar los mensajes de manera adecuada.

AutoOffsetReset

Definimos la opción "AutoOffsetReset", que determina desde qué punto comenzaremos a consumir mensajes si no existe un desplazamiento previo (offset). Podemos elegir entre:

- **"Earliest"**: Inicia desde el mensaje más antiguo.



- **"Latest"**: Inicia desde el mensaje más reciente.

Esta configuración nos permitirá controlar el punto de partida del consumo de mensajes en el sistema.

Topic

Ingresamos el nombre del "Topic", que es el canal de comunicación en Kafka donde se publican los mensajes que BIZUIT procesará.

Usuario y Contraseña

Si el servidor Kafka requiere autenticación, ingresamos el usuario y la contraseña correspondientes. Esto garantizará que solo los usuarios autorizados puedan acceder a los datos del tópico.

Paso 2: Configuración del Proceso a Ejecutar

Seleccionamos el proceso en BIZUIT que se ejecutará cada vez que recibamos un mensaje de Kafka. Este proceso debe estar configurado previamente y puede incluir acciones como almacenar datos en la base de datos, activar alertas o iniciar otros flujos de trabajo.

También debemos seleccionar el parámetro del proceso al cual se enviará el mensaje.



Escribir en la Base de Datos

Si queremos registrar cada mensaje recibido en una base de datos, activamos la opción "Escribir en la Base de Datos". Esto nos permitirá mantener un historial de los mensajes y realizar análisis posteriores.

Paso 3: Configuración Avanzada

Importancia Alta

ApiVersionRequest, EnableAutoCommit, EnableAutoOffsetStore

- Activamos "ApiVersionRequest" para solicitar la versión de la API de Kafka, lo que nos ayudará a garantizar la compatibilidad con diferentes versiones del servidor.

- Habilitamos "EnableAutoCommit" para que BIZUIT confirme automáticamente la recepción de cada mensaje.
- Activamos "EnableAutoOffsetStore" para almacenar los desplazamientos (offsets) de manera automática.

ApiVersionRequest

EnableAutoCommit

EnableAutoOffsetStore

SecurityProtocol

Seleccionamos el "SecurityProtocol" adecuado para la conexión, eligiendo entre opciones como SSL o SASL_SSL, dependiendo de la configuración de seguridad de Kafka. Esto garantizará una comunicación cifrada y segura.

SessionTimeoutMs y MaxPollIntervalMs

Configuramos "SessionTimeoutMs", que define el tiempo de espera de la sesión en milisegundos.

También especificamos "MaxPollIntervalMs", que determina el intervalo máximo entre cada lectura de mensajes para mantener el consumidor activo.

En importancia media y baja, definimos cada uno de los campos necesarios. Podemos consultar la documentación en línea para conocer el propósito de cada uno.

Importancia media			
ApiVersionFallbacks	AutoCommitIntervalMs	BrokerVersionFallback	<input checked="" type="checkbox"/> CheckCrcs
0	5000	0.10.0	
Debug	FetchErrorBackoffMs	FetchMaxBytes	GroupInstanceId
	500	52428800	
MaxPartitionFetchBytes	MessageMaxBytes	PartitionAssignmentStrategy	QueueIdMaxMessagesKbytes
1048576	1000000		65536
QueueIdMaxMessages	ReceiveMaxBytes	ReconnectBackoffMaxMs	ReconnectBackoffMs
100000	100000	10000	100

Maximum Kafka protocol request message size. Due to differing framing overhead between protocol versions the producer is unable to reliably enforce a strict max message limit of produce [unreadable]

Paso 4: Guardar la Configuración

Una vez que hemos completado todos los ajustes, revisamos la configuración para asegurarnos de que sea correcta y presionamos "Guardar". Luego, habilitamos el servicio de Monitoreo de Mensajes de Kafka. A partir de este momento, BIZUIT comenzará a consumir mensajes del tópico especificado y a ejecutar el proceso configurado en tiempo real.

Con estos pasos, hemos configurado un canal de Monitoreo de Mensajes Kafka en BIZUIT.

Este canal es esencial para integraciones en tiempo real con sistemas de eventos y nos permitirá consumir y procesar grandes volúmenes de datos de manera eficiente. Es recomendable probar la configuración para verificar que los mensajes se reciben correctamente y que el proceso se ejecuta como se espera.

Configuración Canal Administradores de Colas de Mensajes

Configuraremos ahora un canal Administrador de Cola de Mensajes en BIZUIT. Este canal nos permitirá gestionar la cola de mensajes y definir cómo se procesarán de manera secuencial o en caso de errores.

Es especialmente útil en aplicaciones donde necesitamos garantizar la entrega de mensajes o ejecutar procesos de manera confiable, con opciones de reintento en caso de fallas.

Exploraremos cada uno de los aspectos de la configuración de este canal.

Creando Nuevo Canal Administrador de Cola de Mensajes

Nombre de Cola de Mensaje *
ColaDemo

General Configuración General del Servicio

Proceso * ProcesoAsincrono Parámetro de Entrada * DataClienteCola Parámetro de Salida * PaisCliente

Reintentos 3

1º Reintento	5	Intervalo en segundos
2º Reintento	5	Intervalo en segundos
3º Reintento	5	Intervalo en segundos

Habilitado

Guardar Cancelar

Paso 1: Configuración General de la Cola de Mensajes

Nombre de la Cola de Mensajes

Asignamos un nombre descriptivo a la cola de mensajes, como "ColaDemo". Este nombre nos permitirá identificar la cola dentro del sistema y facilitará su administración cuando tengamos múltiples colas configuradas en BIZUIT.

Proceso

Seleccionamos el proceso que queremos ejecutar cuando se reciba un mensaje en la cola. En este caso, utilizamos "ProcesoAsincrono", un proceso que consulta la base de clientes a partir

de un ID recibido. Este proceso debe estar previamente configurado en BIZUIT y preparado para recibir y procesar los datos correspondientes.

Debemos tener en cuenta que solo estarán disponibles aquellos procesos que:

- Cuenten con uno y solo un parámetro de entrada escalar obligatorio.
- O al menos un parámetro de entrada escalar opcional.
- Pueden tener tantos parámetros escalares de salida como necesitemos.

Parámetro de Entrada

El "Parámetro de Entrada" define el campo donde se almacenará la información del mensaje. En este caso, seleccionamos "DatoClienteCola", que contendrá la información que la consulta SQL necesita para ejecutarse correctamente.

Parámetro de Salida

El "Parámetro de Salida" es donde el proceso enviará la respuesta o el resultado del procesamiento. En este ejemplo, configuramos "PaisCliente" como el campo de salida de nuestro proceso asíncrono. Esta configuración asegura que el resultado del proceso esté disponible para futuras referencias o acciones.

Paso 2: Configuración de Reintentos

Cantidad de Reintentos

En el campo "Reintentos", especificamos el número de intentos que BIZUIT realizará en caso de que el proceso falle. En este ejemplo, configuramos el sistema para intentar tres veces si ocurre un error. Esto nos permite contar con múltiples oportunidades de procesar el mensaje en caso de problemas temporales.

Reintentos

1º Reintento Intervalo en segundos

2º Reintento Intervalo en segundos

3º Reintento Intervalo en segundos

Intervalo de Reintento

Definimos el intervalo de tiempo entre cada intento de Re ejecución. En este caso, configuramos los siguientes intervalos:



- **Primer intento:** 5 segundos.
- **Segundo intento:** 15 segundos.
- **Tercer intento:** 30 segundos.
-

Esto significa que, si el proceso falla, BIZUIT esperará 5 segundos antes de volver a intentar procesar el mensaje. Si el segundo intento también falla, esperará 15 segundos antes de intentarlo nuevamente. Si el tercer intento también falla, esperará 30 segundos. Si después de estos intentos el mensaje no logra procesarse correctamente, será enviado a la cola de mensajes fallidos.

Este enfoque ayuda a reducir la carga del sistema y permite resolver problemas transitorios que podrían estar causando los fallos en el proceso.

Paso 3: Habilitar el Canal

Habilitado

Finalmente, activamos el interruptor de "**Habilitado**" para asegurarnos de que la cola esté operativa y lista para recibir y procesar mensajes. Si esta opción no está activada, el canal no estará activo y no procesará ningún mensaje.

Paso 4: Guardar la Configuración

Una vez que hemos configurado todos los parámetros, revisamos la configuración para asegurarnos de que todo esté correcto. Luego, presionamos "Guardar" y habilitamos el canal Administrador de Cola de Mensajes. A partir de este momento, BIZUIT gestionará los mensajes de la cola según los parámetros configurados, incluyendo los reintentos en caso de error. Con estos pasos, hemos configurado un canal Administrador de Cola de Mensajes en BIZUIT.

Este canal es fundamental para garantizar la entrega confiable de mensajes y ejecutar procesos de manera controlada, especialmente en aplicaciones críticas que requieren un manejo de errores eficiente y reintentos automáticos.

Ahora, realizaremos pruebas de configuración para verificar que los mensajes se procesen correctamente y que el sistema responda adecuadamente en caso de fallos.

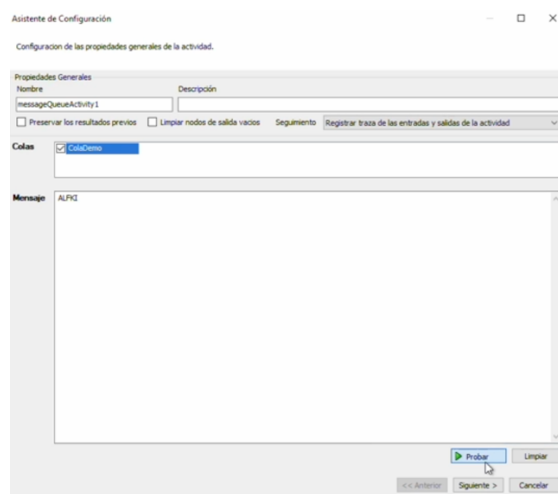
Prueba del Canal Administradores de Colas de Mensajes

Ahora, exploraremos cómo probar el funcionamiento del administrador de cola de mensajes de BIZUIT. Comenzamos creando un proceso que reciba, como parámetros, los datos que deseamos enviar al proceso que gestionará la cola.

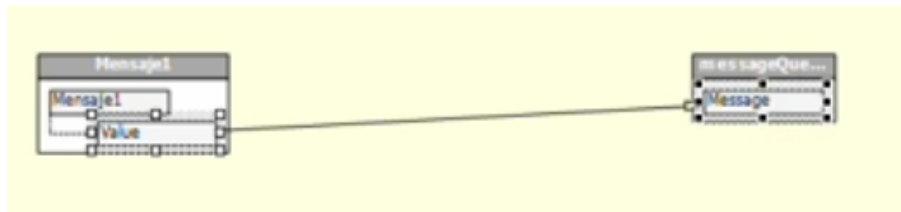
Paso 5: Creación del Proceso de Prueba

Una vez que hemos creado el proceso, configuramos una actividad de Cola de Mensajes. En ella, seleccionamos la cola a la que queremos enviar el mensaje. En nuestro caso, el proceso encolado (llamado *ProcesoAsincrono*) está diseñado para recibir un Id de Empleado, y utilizaremos el id de cliente ALFKI como ejemplo.

Al ingresar el id de cliente, pulsamos el botón Probar. Si todo funciona correctamente, el sistema nos devolverá un ID de mensaje, confirmando que el mensaje se ha encolado exitosamente.



Procedemos a mapear el parámetro de entrada del proceso, denominado *Mensaje1*, con el parámetro *Message* esperado por la actividad.



A continuación, mapeamos el nodo *ColaDemo* devuelto por la actividad al parámetro de salida del proceso, llamado *Respuesta*, para poder retornar el Id del mensaje encolado.



Luego, configuramos el valor por defecto del parámetro de entrada *Mensaje1* con el valor "ANTON" y ejecutamos el proceso.

Configuración de Parámetros

Configuración de Parámetros

Agregar nuevo parámetro

Nombre: Mensaje1

Descripción:

Tipo de Parámetro: Escalar XML Dirección: In

Propiedades del Parámetro Escalar

Tipo de Dato: string

Valor Predeterminado: ANTON

Propiedades del Parámetro XML

Establecer Esquema XML Predeterminado

Aceptar Cancelar

Paso 6: Configuración del Visor de Cola en BIZUIT Dashboard

Ingresamos al módulo Visor de cola de mensajes en el BIZUIT Dashboard, el cual nos permite visualizar y monitorizar el estado de los mensajes encolados.

Monitor de Cola de Mensajes

Fecha Desde: 7/10/2024 12:00 a.m. Fecha Hasta: 7/11/2024 11:59 p.m. Cola: Todas Filtros

Estado	Total	Fecha Inicio
Total	2	2024-11-07T15:25:20
En Proceso	0	
Procesado con Éxito	2	2024-11-07T15:25:20
Procesado con Errores	0	

Estado	Nombre de Cola	Transacciones	Procesado con Éxito	Procesando	Procesado con Errores	
<input checked="" type="checkbox"/>	ColaDemo	2	2	0	0	<input type="checkbox"/>

Pantalla de Monitoreo General

En la pantalla de monitoreo general observamos un resumen de los mensajes en las colas configuradas, que incluye las siguientes estadísticas:

- **Total:** cantidad total de mensajes procesados.
- **En Proceso:** mensajes que se encuentran actualmente en procesamiento.
- **Procesado con Éxito:** cantidad de mensajes procesados sin inconvenientes.
- **Procesado con Errores:** cantidad de mensajes que presentaron algún error durante el procesamiento.

Pantalla de Detalle de la Cola de Mensajes

Al refrescar la vista, se despliega el detalle de los mensajes *Procesados con Éxito*, permitiéndonos acceder a información más específica. Aquí podemos visualizar, para cada mensaje, los siguientes datos:

Ver	Historial	MessageId	InstanceId	Mensaje	Respuesta	Fecha Inicio	Fecha Fin	Fecha de Cola	Reintentos
Q		3347dfcc-b35a-4c26-be7f-43663def673b	704bd900-173e-4b14-8292-72ef19d4785f	ALFKI	Germany	07/11/2024 15:25:20	07/11/2024 15:25:20	07/11/2024 15:25:29	0
Q		116d4528-c3df-4cb2-9830-c3cca9de092c	6629f44d-0a54-4905-91f4-0ba33986f13d	ANTON	Mexico	07/11/2024 15:25:20	07/11/2024 15:25:20	07/11/2024 15:24:15	0

- **MessageId:** identificador único del mensaje.
- **Instanciald:** identificador de la instancia que procesó el mensaje en BIZUIT.
- **Mensaje:** contenido del mensaje, en nuestro ejemplo, el id de empleado (por ejemplo, *BOLID* o *ALFKI*).
- **Respuesta:** resultado o respuesta generada por el proceso encolado al procesar el mensaje.
- **Fecha de Inicio y Fecha de Fin:** indican los tiempos de inicio y finalización del procesamiento.
- **Fecha de Cola:** momento en el que se encoló el mensaje.
- **Reintentos:** número de veces que se ha intentado procesar el mensaje, lo que nos ayuda a identificar posibles incidencias.

Además, podemos visualizar la traza de ejecución de aquella instancia que se procesó con éxito.

Detalle de cola de mensajes

Historial de Mensajes

MessageId	InstanceId	Mensaje	Respuesta	Estado	Fecha Inicio	Fecha Fin	Fecha de Cola	Reintentos
T16d4528-c3df-4cb2-9830-c3cc99de092c	6629f44d-0b54-4905-91f4-0ba33986f13d	ANTON	Mexico	OK	07/11/2024 15:25:20	07/11/2024 15:25:20	07/11/2024 15:24:15	0

Cerrar

Verificación de los Reintentos

Para comprobar el funcionamiento de los reintentos, provocamos intencionalmente un fallo en el proceso encolado. Para ello, insertamos una actividad de excepción, republicamos el proceso y ejecutamos nuevamente el proceso de encolado.



Observaremos que, tras iniciar la ejecución, aparece un mensaje en proceso. Al refrescar la vista, notamos que se han registrado tres reintentos, hasta que finalmente el mensaje deja de reintentarse.

Detalle de cola de mensajes

Id Message	Id Instancia	Mensaje	Cantidad Reintentos				
Respuesta							
Filtrar Eliminar Filtros							
MessageId	InstanceId	Mensaje	Respuesta	Fecha Inicio	Fecha Fin	Fecha de Cola	Reintentos
60c4f164-6a24-4881-9d8b-8cb47cc2441	5faef69f-832e-4947-ae31-d5ebf738180d	ANTON				07/11/2024 15:28:49	3

Cerrar



Accedemos al detalle de los mensajes procesados con errores y examinamos la información específica de cada uno; en nuestro caso, solo se ha registrado uno.

Aquí podemos identificar la excepción o el error que se produjo durante el procesamiento, lo que resulta fundamental para la resolución de problemas, ya que nos permite conocer en detalle la causa del fallo, la actividad involucrada y los intentos de reprocesamiento.

Si seleccionamos la opción de detalle (haciendo clic en la lupa de un mensaje específico), accedemos a un registro completo de todos los intentos de procesamiento.

MessageId	InstanceId	Mensaje	Respuesta	Estado	Fecha Inicio	Fecha Fin	Fecha de Cola	Reintentos
60c41f64-6a24-4881-9d8b-8cb47fcc2441	04f77048-3f47-4065-8bed-0f06451afc14	ANTON	System.Exception: Error Forzado at Tjcon.BIZUIT.Activities.ThrowException.ExceptionActivity.Execute(ActivityExecutionContext executionContext) at System.Workflow.ComponentModel.ActivityExecutor.Operation.Run(WorkflowCoreRuntime workflowCoreRuntime) at System.Workflow.Runtime.Scheduler.Run()	Procesando	07/11/2024 15:28:50	07/11/2024 15:28:50	07/11/2024 15:28:49	0
60c41f64-6a24-4881-9d8b-8cb47fcc2441	b186047d-7f49-48f2-960a-dd8833f19562	ANTON	System.Exception: Error Forzado at Tjcon.BIZUIT.Activities.ThrowException.ExceptionActivity.Execute(ActivityExecutionContext executionContext) at System.Workflow.ComponentModel.ActivityExecutor.Operation.Run(WorkflowCoreRuntime workflowCoreRuntime) at System.Workflow.Runtime.Scheduler.Run()	Procesando	07/11/2024 15:29:05	07/11/2024 15:29:05	07/11/2024 15:28:49	1
60c41f64-6a24-4881-9d8b-8cb47fcc2441	5fae69f-832e-4947-9e3f-d9ebf738180d	ANTON	System.Exception: Error Forzado at Tjcon.BIZUIT.Activities.ThrowException.ExceptionActivity.Execute(ActivityExecutionContext executionContext) at System.Workflow.ComponentModel.ActivityExecutor.Operation.Run(WorkflowCoreRuntime workflowCoreRuntime) at System.Workflow.Runtime.Scheduler.Run()	Procesando	07/11/2024 15:29:20	07/11/2024 15:29:20	07/11/2024 15:28:49	2
60c41f64-6a24-4881-9d8b-8cb47fcc2441	a35d43f1-560b-4d03-8dce-90bf7e4c2759	ANTON	System.Exception: Error Forzado at Tjcon.BIZUIT.Activities.ThrowException.ExceptionActivity.Execute(ActivityExecutionContext executionContext) at System.Workflow.ComponentModel.ActivityExecutor.Operation.Run(WorkflowCoreRuntime workflowCoreRuntime) at System.Workflow.Runtime.Scheduler.Run()	Procesando	07/11/2024 15:29:50	07/11/2024 15:29:50	07/11/2024 15:28:49	3

En esta sección se muestra cada intento con su estado, la fecha y hora exacta de cada procesamiento y los errores que pudieron haber ocurrido.

Finalmente, si deseamos reenviar un mensaje que falló, contamos con la opción de reenvío para que vuelva a intentar procesarse. Esto es especialmente útil una vez que se ha solucionado el problema que causó el error, permitiéndonos procesar los mensajes sin necesidad de ingresarlos nuevamente.

Procedemos a quitar la actividad de error y republicamos el proceso encolado. Luego, reenviamos el mensaje desde la consola de mensajes fallidos y comprobamos que, en esta ocasión, se procesa correctamente.

Con esta demostración hemos aprendido a configurar y probar el administrador de cola de mensajes en BIZUIT, interpretar la información detallada del módulo de visor de colas y utilizar eficazmente las herramientas para gestionar los mensajes.

Este proceso es fundamental para garantizar que el flujo de mensajes en nuestros procesos se lleve a cabo de forma controlada, permitiéndonos identificar y resolver cualquier inconveniente que pueda surgir.

Configuración Canal Clientes TCP

En esta sección aprenderemos a configurar un canal Cliente TCP en BIZUIT. Este tipo de canal nos permite establecer una conexión con un servidor TCP externo para enviar y recibir datos en tiempo real, lo cual es esencial para integrar BIZUIT con dispositivos o aplicaciones remotas.

The screenshot shows the 'Creando Nuevo Canal Cliente TCP' (Creating New TCP Client Channel) configuration page. It includes a form with the following fields and values:

Configuración General del Servicio		
Dirección IP del Servidor TCP *	Puerto Servidor *	Tipo de Protocolo *
129.333.444.211	8900	Tcp
Tempo de Espera para el envío *	Caracteres de inicio de envío *	Caracteres de fin de envío *
5000	0d	!c,0d
Tempo de Espera en la Recepción *	Caracteres de inicio de respuesta *	Caracteres de fin de respuesta *
5000	0d	!c,0d
Codificación *	Tipo de Socket *	Familia de Direcciones *
UTF8	Stream	InterNetwork
Unger Seconds *	Tamaño de Paquete Predeterminado *	
60	8192	

Below the configuration table, there is a section for 'Proceso a ejecutar' (Process to execute) with the following values:

Proceso *	Parámetro *
TCPTest	Mensaje

Configuración Inicial del Canal

Asignación del Nombre y Descripción

Comenzamos asignando un nombre descriptivo al canal, por ejemplo, ClienteTCP. En la descripción, especificamos su propósito, como: "Recibe y envía datos en tiempo real a un servidor TCP".

Esto nos ayudará a identificar fácilmente el canal en la interfaz de BIZUIT.

Configuración General del Servicio

En esta sección, configuramos los parámetros básicos para la conexión:

Dirección IP del Servidor TCP

Ingresamos la Dirección IP del servidor al que BIZUIT se conectará. Verifiquemos que la IP sea accesible desde la red de BIZUIT.



Puerto del Servidor

Definimos el Puerto Servidor que se utilizará para la comunicación.

Muchos servicios emplean el puerto 5000 o 8080, pero este valor dependerá de la configuración del servidor.

Tipo de Protocolo

Seleccionamos el Tipo de Protocolo:

- **TCP:** Para conexiones confiables.
- **UDP:** Para transmisiones más rápidas, aunque con menor seguridad.

Tiempos de Espera

Configuramos los tiempos máximos de espera para:

- **Envío:** El tiempo que BIZUIT esperará para enviar un mensaje.
- **Recepción:** El tiempo que esperará por la respuesta del servidor.
- **Ejemplo:** 5000 ms (equivalente a 5 segundos) para cada uno.

Caracteres de Inicio y Fin de Mensaje

Definimos los caracteres que indican:

- **Inicio y fin del mensaje enviado:** Esto delimita claramente el contenido del mensaje.
- **Inicio y fin de la respuesta:** Para que BIZUIT sepa cuándo comienza y termina la respuesta del servidor.

Codificación

Seleccionamos la codificación adecuada para el intercambio de datos, por ejemplo, UTF-8 o ASCII, lo que garantiza la correcta interpretación de caracteres especiales.

Tipo de Socket y Familia de Direcciones

Elegimos el Tipo de Socket y la Familia de Direcciones (por ejemplo, IPv4 o IPv6) según la red y el servidor con el que nos comunicamos.

Tamaño de Paquete Predeterminado

Establecemos el tamaño máximo de datos que se enviarán o recibirán en cada paquete. Esto es fundamental para manejar de manera eficiente grandes volúmenes de información.



Configuración del Proceso a Ejecutar

Elegimos el proceso de BIZUIT que se ejecutará cuando se reciba información del servidor TCP. En este caso, usamos el proceso TCPTest.

Definición del Parámetro

Asignamos el parámetro que recibirá la información del servidor. Utilizamos el parámetro Mensaje, que contendrá el dato de entrada.

Guardar la Configuración

Antes de finalizar, revisamos que todos los detalles estén correctos. Luego, presionamos el botón Guardar para activar el canal Cliente TCP. Desde ese momento, BIZUIT estará preparado para conectarse al servidor TCP y gestionar el envío y la recepción de datos conforme a los parámetros que hemos configurado.

Mediante estos pasos, hemos configurado de forma efectiva un canal Cliente TCP en BIZUIT.

Este canal es una herramienta fundamental para asegurar la comunicación en tiempo real con servidores externos, permitiéndonos integrar y automatizar el intercambio de datos de manera confiable. Es recomendable probar la configuración con el servidor correspondiente para confirmar que todo funcione según lo previsto y ajustar cualquier parámetro en caso de ser necesario.

Con esta guía, pretendemos que comprendas de manera clara y práctica cada uno de los componentes necesarios para implementar y aprovechar al máximo un canal Cliente TCP en BIZUIT.



Resumen de la Unidad

En esta unidad, abordamos cómo configurar diferentes tipos de canales en BIZUIT, comenzando con el canal API REST, que permite exponer servicios y acceder a información en tiempo real. A través de pasos progresivos, aprendimos a definir un canal, configurar su endpoint, seleccionar los parámetros de entrada y salida, y activar las opciones de seguridad.

También exploramos cómo probar estos servicios mediante herramientas como Postman, además de detallar cómo configurar canales de Servicio Web, Programadores de Procesos, y Monitoreo de Archivos y Cuentas IMAP. Todos estos canales automatizan procesos, mejorando la integración y la eficiencia operativa.

Exploramos la configuración de diversos canales en BIZUIT, desde la creación de un canal API REST para exponer servicios en tiempo real hasta la configuración de servicios web, programadores de procesos, y canales para monitorear archivos, correos IMAP, servidores TCP y mensajes Kafka. Cada configuración se inicia asignando un nombre descriptivo y una explicación detallada que clarifica el propósito del canal, definiendo endpoints, parámetros de entrada y salida, y estableciendo medidas de seguridad y autenticación para garantizar comunicaciones seguras y eficientes.

Además, se abordaron aspectos prácticos como la integración con procesos específicos de BIZUIT, la automatización de tareas periódicas y la implementación de reintentos para gestionar errores en la transmisión de mensajes. En resumen, esta unidad ofrece una guía integral y didáctica que permite a los usuarios configurar, probar y optimizar distintos canales de comunicación, facilitando la integración de BIZUIT con sistemas externos y mejorando la gestión y supervisión de procesos en entornos complejos.



Unidad 4: Buenas Prácticas en la Configuración de Canales

En esta unidad profundizaremos en las mejores prácticas para la configuración de canales en BIZUIT. En nuestro enfoque didáctico, no solo nos centraremos en lograr que los canales funcionen correctamente, sino en garantizar su seguridad, eficiencia y confiabilidad a lo largo del tiempo.

A continuación, revisaremos cómo proteger nuestros canales, monitorear su actividad y optimizar su rendimiento para asegurar un funcionamiento continuo y sin inconvenientes.

Configuración Segura de Canales

En BIZUIT sabemos que la seguridad es la base de cualquier sistema de integración. Al configurar un canal, ya sea de API REST o de servicios web, es fundamental que implementemos métodos de autenticación sólidos. Recomendamos habilitar la autenticación basada en tokens o API keys, de manera que solo usuarios o aplicaciones autorizadas puedan acceder a la información.

Revisemos periódicamente los permisos y apliquemos el principio de "menor privilegio", garantizando que cada usuario tenga acceso únicamente a lo estrictamente necesario. En un canal API REST que expone datos de clientes, asignamos una API key específica y la actualizamos de forma regular para reforzar la seguridad.

Frecuencias de Actualización y Límites de Conexiones

Cada canal debe operar con una frecuencia de actualización acorde a su función. Por ejemplo, en canales de monitoreo de archivos o de cuentas IMAP, no es necesario verificar cambios cada segundo; una frecuencia de 10 a 15 minutos suele ser suficiente.

Establezcamos límites de conexión para evitar que múltiples solicitudes simultáneas sobrecarguen el sistema. En el caso de un canal de Monitoreo de Mensajes Kafka, por ejemplo, es prudente limitar el número de consumidores concurrentes para prevenir conflictos en el procesamiento de datos. Si gestionamos un canal de Monitoreo de Archivos que supervisa



una carpeta FTP compartida por varios proveedores, una actualización cada 5 minutos puede equilibrar la eficiencia y la carga del sistema.

Cifrado y Conexiones Seguras

Para canales como los Servidores y Clientes TCP, es esencial habilitar conexiones seguras (SSL/TLS) que protejan la integridad de los datos durante la transmisión.

Usemos siempre certificados SSL actualizados y evitemos protocolos obsoletos.

En cualquier canal que soporte conexiones seguras, aseguremos que se utilice la versión más reciente de TLS. En un canal TCP que recibe datos de dispositivos médicos, el cifrado garantiza que la información no pueda ser interceptada ni manipulada.

Monitoreo y Auditoría de la Actividad

Configuración de Alertas

Configurar alertas nos permite detectar de manera temprana problemas en la operación de los canales. BIZUIT nos permite activar notificaciones que se disparen en caso de errores de conexión, sobrecargas o reintentos fallidos.

Usemos alertas de "reintentos fallidos" para identificar rápidamente problemas críticos, como en canales de Monitoreo de Archivos o cuentas IMAP, y actuar antes de que se acumulen incidencias.

Si un canal de Monitoreo de Cuentas IMAP detecta tres fallos consecutivos en el acceso, una alerta nos avisará para que revisemos la autenticación y solucionemos el inconveniente.

Registros y Auditoría

Mantener un historial detallado de la actividad de nuestros canales es clave tanto para el análisis de problemas como para el cumplimiento de normativas. BIZUIT nos permite habilitar registros de auditoría que documentan todos los eventos importantes.

Activemos el registro de auditoría en los canales críticos y almacenemos esta información en una base de datos, facilitando así la revisión y el rastreo de posibles incidencias.

En un canal API REST que gestiona solicitudes de clientes, la auditoría nos permitirá conocer quién accedió a cada recurso, lo cual es esencial en caso de consultas o disputas.



Uso de Dashboards de Monitoreo en Tiempo Real

Integrar dashboards en tiempo real nos ayuda a visualizar de forma inmediata el estado y el rendimiento de nuestros canales. Con estas herramientas, podemos monitorear la actividad, el consumo de recursos y detectar posibles cuellos de botella.

Configuremos dashboards específicos para canales de alta carga, como Kafka, y ajustemos la configuración si observamos picos en el volumen de mensajes. Si notamos un incremento repentino en el flujo de mensajes en un canal Kafka, podemos asignar temporalmente más recursos o limitar el consumo hasta que el flujo se estabilice.

Consejos de Optimización y Mantenimiento

Ajuste de Parámetros de Rendimiento

A medida que aumenta la carga de trabajo, es vital ajustar los parámetros de rendimiento de nuestros canales. Esto incluye el tamaño de los paquetes de datos, los intervalos de reintentos y el número de conexiones simultáneas.

Para canales de Servidores TCP, experimentemos con distintos tamaños de paquete para optimizar la transmisión de datos y evitar saturaciones en el ancho de banda. Un canal TCP que transmite datos de sensores puede beneficiarse de paquetes más grandes, permitiendo una transmisión más fluida y sin interrupciones.

Pruebas de Carga y Monitoreo de Cuellos de Botella

Realizar pruebas de carga de forma regular nos permite verificar que los canales soporten el volumen de datos sin problemas y detectar posibles cuellos de botella.

Establezcamos un ambiente de pruebas donde simulemos la carga máxima esperada y analicemos el comportamiento del canal, ajustando parámetros como los tiempos de espera o los límites de mensajes concurrentes.

En un canal Kafka que recibe datos de ventas en tiempo real, las pruebas de carga pueden revelar la necesidad de ajustar el sistema para manejar múltiples mensajes por segundo sin afectar el rendimiento.

Mantenimiento Preventivo y Documentación

El mantenimiento preventivo es esencial para garantizar la operatividad continua de nuestros canales. Esto implica revisar periódicamente las configuraciones, actualizar credenciales y limpiar los registros antiguos para evitar saturaciones.



Programemos revisiones trimestrales de los canales y documentemos cada cambio en un documento centralizado, asegurando una transición fluida en caso de modificaciones o cambios en el equipo de soporte. Si modificamos el intervalo de reintentos en un canal de IMAP, documentemos el cambio y la justificación para que cualquier miembro del equipo comprenda la modificación.

Con estas prácticas, nos aseguramos de que los canales en BIZUIT no solo funcionen de manera óptima, sino que también sean seguros, monitoreados y mantenidos de forma proactiva.

Al aplicar estos consejos y estrategias, garantizamos un entorno de integración robusto y resiliente, listo para enfrentar cualquier desafío y adaptarse a las necesidades crecientes de nuestros sistemas.



Resumen del Capítulo

En este capítulo, se exploró cómo los canales en BIZUIT permiten la integración y automatización de procesos empresariales, facilitando la comunicación entre sistemas internos y externos. Se destacó su capacidad para actualizar datos en tiempo real, reducir la intervención manual y mejorar la eficiencia operativa en sectores como salud y logística.

Se introdujo el concepto de canales, explicando su papel como conectores automatizados que optimizan la sincronización de información y agilizan la toma de decisiones. Se analizaron los distintos tipos de canales disponibles en BIZUIT, incluyendo API REST, servicios web, programadores de procesos, monitoreo de archivos y cuentas IMAP, servidores y clientes TCP, mensajes Kafka y colas de mensajes, cada uno con aplicaciones específicas para la automatización y optimización de flujos de trabajo.

Se ilustraron casos de uso en los que los canales permitieron desde la gestión eficiente de registros médicos hasta la automatización de la actualización de pedidos y la transmisión de datos en tiempo real en arquitecturas distribuidas.

Posteriormente, se abordó la configuración práctica de los canales, detallando el proceso para definir endpoints, establecer parámetros de entrada y salida, configurar opciones de seguridad y realizar pruebas de integración con herramientas como Postman.

Se explicó cómo exponer procesos internos como servicios REST o SOAP, programar tareas automáticas con los programadores de procesos, activar flujos de trabajo con monitoreo de archivos y correos, y establecer servidores TCP y Kafka para el procesamiento de eventos en tiempo real. También se cubrió la gestión de colas de mensajes, asegurando que los procesos se ejecuten de manera controlada y confiable.

Finalmente, se presentaron buenas prácticas para garantizar la seguridad, eficiencia y sostenibilidad de los canales configurados.

Se abordaron estrategias de autenticación y control de accesos, optimización del rendimiento mediante la correcta configuración de frecuencias y límites, monitoreo con alertas y dashboards, y gestión de errores con mecanismos de reintento y recuperación automática.

A lo largo del capítulo, se consolidó el conocimiento sobre los canales en BIZUIT, demostrando su importancia en la integración de sistemas y la automatización de procesos. Se proporcionaron herramientas y metodologías para implementar estos canales de manera



eficiente, asegurando que su uso sea seguro, escalable y alineado con las necesidades del negocio.