IBM Unica Marketing Operations
Versión 8 Release 6
25 de mayo de 2012

Módulo de integración

Nota
Antes de utilizar esta información y el producto al que da soporte, lea la información incluida en "Avisos" en la página 33.

Esta edición se aplica a la versión 8, release 6, modificación 0 de IBM Unica Marketing Operations y a todos los releases y las modificaciones posteriores, hasta que se indique lo contrario en nuevas ediciones.

Contenido

Capítulo 1. ¿Qué es Servicios de integración de Marketing Operations?

Servicios de integración de Marketing Operations es un compuesto de lo siguiente.

- **Servicio web de integración de Marketing Operations**

  Los servicios de integración proporcionan una forma para los clientes de IBM® Unica Marketing Operations, los socios y los servicios profesionales de IBM para integrar Marketing Operations con otras aplicaciones que se ejecutan en su entorno.
- **Procedimientos y API de Marketing Operations**

  Los procedimientos personalizados se pueden definir dentro de Marketing Operations para ampliar la lógica empresarial de Marketing Operations de formas arbitrarias. Una vez definidos, estos procedimientos pueden ser los destinos para las llamadas de servicio web de servicios de integración desde otras aplicaciones. Los procedimientos también se pueden definir para enviar mensajes a otras aplicaciones.
- **Desencadenantes de Marketing Operations**

  Los desencadenantes se pueden asociar a eventos y procedimientos en Marketing Operations. Cuando se produce uno de estos eventos, se ejecuta el desencadenante asociado.

¿Cuáles son los requisitos para los servicios de integración de Marketing Operations?

Los servicios de integración de Marketing Operations deben:

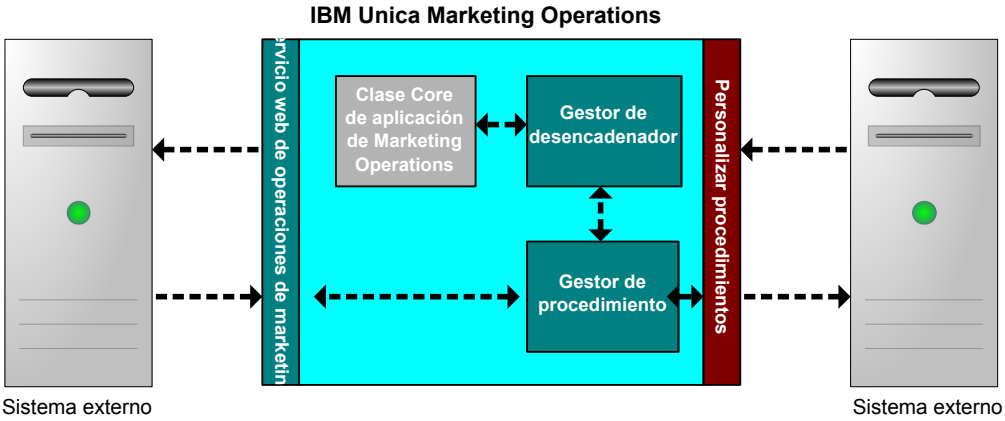
- Acoplar ligeramente la integración del sistema.
- Proporcionar un mecanismo para las aplicaciones de cliente para afectar a Marketing Operations a través de llamadas de servicio web.
- Proporcionar un mecanismo para que se notifique a las aplicaciones de cliente sobre determinados eventos en Marketing Operations.
- Proporcionar un modelo de programación que sea fácil de comprender y utilizar.
- Ser sólidos al recuperarse de un error.
- Garantizar la integridad de los datos.
- Integrarse con y minimizar el efecto en los clientes basados en la GUI de Marketing Operations existentes.
- Proporcionar un acceso preciso a los componentes de Marketing Operations mientras se aísla a los programadores de los detalles de la implementación subyacente.

Cómo empezar con los servicios de integración de IBM Unica Marketing Operations

Puede utilizar las funciones de los servicios de integración de IBM Unica Marketing Operations para crear procedimientos personalizados. Puede utilizar estos procedimientos para desencadenar eventos externos cuando se producen determinados eventos dentro de Marketing Operations. Puede utilizar estos procedimientos para realizar funciones de Marketing Operations desde sistemas o programas externos.

Puede utilizar la API para interactuar con IBM Unica Marketing Operations en el nivel programático, de la misma forma que utiliza la GUI para interactuar con Marketing Operations en un nivel de usuario. Mediante el uso de la API, puede construir procedimientos. Utilizando estos procedimientos puede comunicarse entre Marketing Operations y los sistemas externos. El servicio web de Marketing Operations es el objeto de contenedor para los procedimientos, la API y los desencadenantes.

La arquitectura de los servicios de integración de Marketing Operations se muestra aquí.

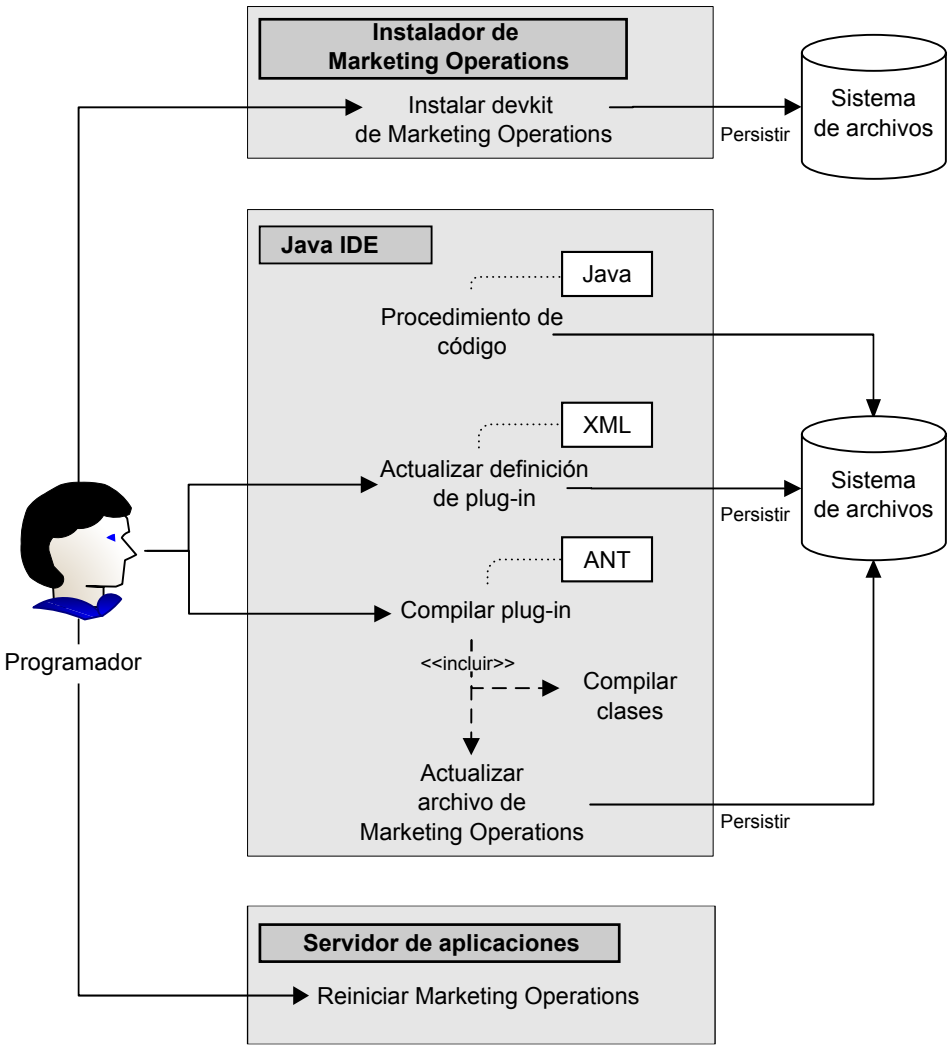


Lo que aparece a continuación son componentes clave de los Servicios de integración.

- Gestor de procedimiento Marketing Operations: amplía la lógica empresarial interactuando con Marketing Operations a través de la API.
- Gestor de desencadenante Marketing Operations: asocia una condición (por ejemplo, el cambio de estado de un objeto de marketing) con una acción (un procedimiento para ejecutar cuando se cumple la condición para el desencadenante).

Metodología

Puede utilizar los componentes de los servicios de integración de IBM Unica Marketing Operations para desarrollar procedimientos personalizados, tal como se muestra en este diagrama:



Tras utilizar el instalador Marketing Operations para instalar el kit del desarrollador, siga estos pasos básicos:

1. Codifique el procedimiento personalizado. Actualmente, debe utilizar Java.
2. Actualice la definición de plugin en el archivo de definición XML.
3. Cree el plugin:
   a. Compile las clases necesarias.
   b. Actualice el archivo Marketing Operations (el archivo WAR).
4. Reinicie Marketing Operations.

Ejemplos básicos para comunicarse entre IBM Unica Marketing Operations y la API

En este apartado se describe un ejemplo básico de establecimiento de comunicación entre la API y Marketing Operations. Si esto no realiza ningún trabajo útil; realiza un viaje de ida y vuelta entre Marketing Operations y los Servicios de integración.

En este apartado se utilizan partes de los procedimientos de ejemplo incluidlos con el kit del desarrollador de los servicios de integración de Marketing Operations. Específicamente, puede encontrar el código al que se hace referencia aquí en las líneas siguientes.

- PlanClientFacade.java
- PlanWSNOOPTestCase.java

El método `noop` es una llamada de servicio web a Marketing Operations. Está definido en la clase PlanClientFacade y pasa valores nulos en una matriz.

```
public ProcedureResponse noop(String jobId)
  throws RemoteException, ServiceException {
  NameValueArrays parameters =
    new NameValueArrays(null, null, null, null, null, null, null, null);
  return _serviceBinding.executeProcedure("uapNOOPProcedure", jobId, parameters);
}
```

El procedimiento testExecuteProcedure llama al método `noop` desde PlanClientFacade para establecer un viaje de ida y vuelta con la aplicación Marketing Operations.

```
public void testExecuteProcedure() throws Exception {
  // Tiempo de espera excedido después de un minuto
  int timeout = 60000;
  PlanClientFacade clientFacade = new PlanClientFacade(urlWebService, timeout);
  System.out.println("noop w/no parameters");
  long startTime = new Date().getTime();
  ProcedureResponse response = clientFacade.noop("junit-jobid");
  long duration = new Date().getTime() - startTime;

  // cero o estado positivo => satisfactorio
  System.out.println("Status: " + response.getStatus());
  System.out.println("Duration: " + duration + " ms");
  assertTrue(response.getStatus() >= 0);
  System.out.println("Done.");
}
```

Si desea ver detalles sobre NameValueArrays, ProcedureResponse y otros métodos y tipos de datos listados, consulte los apartados específicos en el recordatorio de esta guía y los JavaDocs.

JavaDocs alojados

Si desea información específica sobre los métodos de API públicos, consulte la clase iPlanAPI en los archivos de documentación de API JavaDocs. Estos archivos están disponibles iniciando sesión en Marketing Operations y seleccionando **Ayuda y documentación del producto** desde cualquier página y descargando el archivo `<version>PublicAPI.zip`.

Capítulo 2. Acerca del servicio web de integración de Marketing Operations

El servicio web proporciona una vista de cliente de los servicios de integración de Marketing Operations, que forman parte del despliegue del servidor IBM Unica Marketing Operations. El servicio se ha diseñado para utilizarse de forma simultánea con los usuarios web de Marketing Operations.

El servicio web soporta una llamada a API, executeProcedure.

Un cliente realiza esta llamada al servicio web directamente.

Autenticación

La autenticación no es necesaria; todos los clientes están asociados a un usuario conocido de IBM Unica Marketing Operations llamado PlanAPIUser. Se da por supuesto que las capacidades de seguridad de este usuario especial las ha configurado un administrador del sistema Marketing Operations para las necesidades de todos los clientes del servicio web.

Es posible que una futura versión del servicio proporcione un mecanismo más general para la autenticación de cliente segura.

Configuración regional

La única configuración regional soportada es la configuración regional configurada actualmente para la instancia del sistema IBM Unica Marketing Operations. Se da por supuesto que todos los datos que dependen de la configuración regional a los que se accede a través del servicio (mensajes, moneda, etc.) están en la configuración regional del sistema.

Es posible que una futura versión del servicio proporcione un mecanismo para que el cliente indique a Marketing Operations que configuración regional utilizar.

Gestión de estado

El servicio web es *sin estado*; la implementación del servicio no guarda información por cliente entre las llamadas a API. Esta característica proporciona una implementación de servicio eficaz y simplifica el soporte a clúster.

Transacciones de base de datos

El servicio web no expone las transacciones de base de datos, ni edita bloqueos en el cliente. Sin embargo, garantiza que el efecto de cualquier ejecución de procedimiento sea *atómico*. Este resultado significa que el procedimiento se realiza correctamente o falla; un fallo deja la base de datos en el mismo estado que si nunca se hubiera llamado a la API.

Acerca de los tipos de datos de Marketing Operations Integration Webservice

En este apartado se definen los tipos de datos utilizados por el servicio web, independientes de una implementación de programación o de enlace de servicio particular.

Se utiliza la siguiente anotación.

- <*tipo*>: <*definición tipo*> define un tipo de datos sencillo. Por ejemplo:
  Manejador: cadena
- <*tipo*>: [<*definición tipo*>] define un tipo de datos o una estructura de datos complejo.
- <*tipo*>: { <*definición tipo*> } define un tipo de datos o una estructura de datos complejo.

Los elementos de tipo complejos y los parámetros de API pueden utilizar estos tipos para declarar matrices. Por ejemplo:

```
Handle [] handles
```

El tipo, manejadores, es una matriz de tipos de manejador.

Tipos primitivos

Los tipos primitivos están limitados a los tipos definidos en la tabla que sigue para simplificar el soporte para enlaces SOAP 1.1. Todos los tipos se pueden declarar como matrices, por ejemplo, **Cadena [ ]**. De forma inherente, los tipos de datos binarios como, por ejemplo, **long** se pueden representar como cadenas mediante un enlace de protocolo (por ejemplo, SOAP). Sin embargo, esta representación no tiene ninguna repercusión en la semántica del tipo, los valores permitidos, etc. tal como lo ve el cliente.

Tabla 1. Tipos primitivos

Tipo de API	Descripción	Tipo SOAP	Tipo Java.
boolean	Valor booleano: **true** o **false**	xsd:boolean	boolean
dateTime	Un valor de fecha y hora	xsd:datetime	Date
decimal	Un valor decimal de precisión arbitraria	xsd:decimal	java.math.BigDecimal
double	Un valor decimal firmado de doble precisión	xsd:double	double
int	Un valor entero de 32 bits firmado	xsd:int	int
integer	Un valor entero firmado de precisión arbitraria	xsd:integer	java.math.BigInteger
long	Un valor entero de 64 bits firmado	xsd:long	long
string	Una cadena de caracteres Unicode	xsd:string	java.lang.String

MessageTypeEnum

```
MessageTypeEnum: { INFORMATION, WARNING, ERROR }
```

MessageTypeEnum es un tipo enumerado que define todos los tipos de mensaje posibles.

- INFORMATION: un mensaje informativo
- WARNING: un mensaje de advertencia
- ERROR: un mensaje de error

Mensaje

```
Mensaje: [tipo MessageTypeEnum, código de cadena, cadena localizedText, cadena logDetail]
```

El mensaje es una estructura de datos que define el resultado de una llamada de API de servicio web. Proporciona campos opcionales para un código no localizado, texto localizado y detalle de registro. Actualmente, todo el texto localizado utiliza la configuración regional establecida para la instancia del servidor IBM Unica Marketing Operations.

Tabla 2. Parámetros de mensaje

Parámetro	Descripción
type	Un MessageTypeEnum, que define el tipo del mensaje.
code	Un código opcional, en formato de cadena, para el mensaje.
localizedText	Una cadena de texto opcional para asociar con el mensaje.
logDetail	Un mensaje de seguimiento de pila opcional.

NameValue

```
NameValue: [nombre cadena, secuencia int]
```

NameValue es un tipo complejo base que define un par nombre-valor. También define una secuencia opcional que utiliza el servicio para construir matrices de valores según sea necesario (las secuencias se basan en cero).

Todos los NameValues con el mismo nombre, pero distintos números de secuencia, se convierten a una matriz de valores y se asocian al nombre común.

El tamaño de matriz se determina mediante el número de secuencia máximo; los elementos de matriz sin especificar tienen valores nulos. Los números de secuencia de matriz deben ser exclusivos. El tipo ampliado proporciona el valor y su tipo.

Tabla 3. Parámetros NameValue

Parámetro	Descripción
nombre	Una cadena que define el nombre de un tipo NameValue.
secuencia	Un entero basado en cero que define el número de secuencia para el valor implícito deNameValue.

Los tipos ampliados NameValue se definen para cada tipo primitivo, del modo siguiente:

Tabla 4. Tipos NameValue ampliados

Tipo ampliado	Descripción
BigDecimalNameValue: NameValue [valor decimal]	Un tipo NameValue cuyo valor es un número decimal de precisión arbitraria.
BigIntegerNameValue: NameValue [valor entero]	Un tipo NameValue cuyo valor es un entero de un tamaño arbitrario.
BooleanNameValue: NameValue [valor booleano]	Un tipo NameValue cuyo valor es un booleano.
CurrencyNameValue: NameValue [valor decimal de configuración regional de cadena]	Un tipo NameValue apto para representar la moneda en alguna configuración regional. La configuración regional es un código de idioma ISO, es decir, los códigos de dos letras minúsculas tal como define ISO-639. Actualmente, la configuración regional debe estar conforme a la configuración regional establecida en la instancia del servidor IBM Unica Marketing Operations.
DateNameValue: NameValue [valor de fecha y hora]	Un tipo NameValue cuyo valor es una fecha.
DecimalNameValue: NameValue [valor doble]	Un tipo NameValue cuyo valor es un número decimal de doble precisión.
IntegerNameValue: NameValue [valor largo]	Un tipo NameValue cuyo valor es un entero de 64 bits.
String NameValue: NameValue [valor de cadena]	Un tipo NameValue cuyo valor es una cadena.

Finalmente, se define una matriz de tipos NameValue ampliados para su uso cuando es necesario definir un conjunto de NameValues de distintos tipos.

```
     NameValueArrays: [
BooleanNameValue[]    booleanValues,
StringNameValue[]     stringValues,
IntegerNameValue[]    integerValues,
BigIntegerNameValue[] bigIntegooleanNameValue,
DecimalNameValue[]    decimalValues,
BigDecimalNameValue[] bigDecimalValues
DateNameValue[]       dateNameValues
CurrencyNameValue[]   currencyValues
     ]
```

executeProcedure

Sintaxis

```
executeProcedure(string key, string jobid, NameValueArrays paramArray)
```

Devuelve

```
int: estado
Message[]: mensajes
```

Descripción

Este método invoca al procedimiento especificado con una matriz de parámetros opcionales. La llamada se ejecuta de forma síncrona; es decir, bloquea el cliente y devuelve el resultado después de la finalización.

Parámetros

Tabla 5. Parámetros de executeProcedure

Nombre	Descripción
key	La clave exclusiva del procedimiento para ejecutar. Se devuelve un error *RemoteException* si no hay ningún procedimiento enlazado a la **clave**.
jobid	Cadena opcional que identifica el trabajo asociado a esta ejecución de procedimiento. Esta cadena es un elemento de paso, pero se puede utilizar para unir los trabajos del cliente con la ejecución de un procedimiento concreto.
paramArray	Una matriz de parámetros para pasar al procedimiento. Se devuelve un estado de error y un mensaje si uno o más de los parámetros no es válido (el tipo incorrecto, un valor ilegal, etc.). Es decisión del cliente determinar los parámetros, sus tipos y el número de valores necesarios para el procedimiento.

Parámetros de devolución

Tabla 6. Parámetros de devolución de executeProcedure

Nombre	Descripción
status	Un código entero: • 0 indica que el procedimiento se ha ejecutado correctamente • un entero indica un error Los procedimientos pueden utilizar el estado para indicar distintos niveles de errores.
messages	Una matriz de cero o más estructuras de datos de mensaje. Si el **estado** es 0, esta matriz no contiene mensajes de ERROR, pero podría contener mensajes INFORMATIVOS o de ADVERTENCIA. Si el **estado** no es cero, los mensajes pueden contener cualquier combinación de mensajes de ERROR, INFORMATIVOS y de ADVERTENCIA.

WDSL de servicios de integración de Marketing Operations

En este tema se define el WSDL (Web Services Definition Language) para los servicios de integración de Marketing Operations. El WSDL se ha definido a mano y es la palabra final en la definición del servicio web.

Axis

Esta versión de servicio web utiliza Axis2 1.5.2 para generar las clases del lado del servidor que conforman la implementación del servicio web desde el archivo WSDL. Los usuarios pueden utilizar cualquier versión de Axis o una técnica no Axis para crear una implementación del lado del cliente para integrarse con la API desde el WSDL proporcionado.

Versión de protocolo

La versión del protocolo se enlaza explícitamente al WSDL del modo siguiente:

- como parte del nombre de WSDL, por ejemplo, `PlanIntegrationService1.0.wsdl`

- Como parte del targetNamespace de WSDL, por ejemplo, `xmlns:tns="http://webservices.unica.com /MktOps/services/PlanIntegrationServices1.0?wsdl"`

WSDL

Se proporciona un archivo WSDL con los servicios de integración de IBM Unica Marketing Operations: `PlanIntegrationServices1.0.wsdl`. El WSDL se entrega en el directorio `integration/examples/soap/plan`. El script de creación de ejemplo utiliza este archivo para generar los stubs del lado del cliente apropiados para conectarse al servicio web.

Capítulo 3. Procedimientos de IBM Unica Marketing Operations

Un *procedimiento* es una clase Java estándar o personalizada alojada por IBM Unica Marketing Operations que realiza alguna unidad de trabajo. Los procedimientos proporcionan una forma para los clientes, socios y los servicios profesionales de IBM Unica para ampliar la lógica empresarial de Marketing Operations de formas arbitrarias.

Los procedimientos siguen un modelo de programación sencillo, utilizando una API bien definida para afectar a los componentes gestionados por Marketing Operations. Los procedimientos se "descubren" a través de un mecanismo de búsqueda sencilla y un archivo de definiciones basado en XML. Marketing Operations ejecuta los procedimientos de acuerdo con las necesidades de sus "clientes". Por ejemplo, en respuesta a una solicitud de integración (de entrada) o la activación de un desencadenante (interna o de salida).

Los procedimientos se ejecutan de forma sincrónica con respecto a su cliente; los resultados pasan a estar disponibles directamente para el cliente y mediante un mecanismo de auditoría persistido. La ejecución de un procedimiento también podría provocar que se activen otros eventos o desencadenantes en Marketing Operations.

Los procedimientos se deben grabar en Java.

Supuestos

Tenga en cuenta los siguientes supuestos en relación a los procedimientos.

- Las clases de implementación del procedimiento se empaquetan en un árbol de clases independiente o un archivo jar y se ponen a disposición general en IBM Unica Marketing Operations a través de una ruta de URL. El gestor de ejecución del procedimiento utiliza un cargador de clases independiente para cargar estas clases, según sea necesario. De forma predeterminada, Marketing Operations busca en el siguiente directorio:

  `<Marketing Operations_home>/devkits/integration/examples/classes`

  Para cambiar este valor predeterminado, establezca el parámetro **integrationProcedureClasspathURL** bajo **Valores > Configuración > Marketing Operations > umoConfiguration > integrationServices**.
- El nombre de clase de implementación del procedimiento sigue los convenios de denominación de Java aceptados para evitar conflictos con “unica” y clases de otros proveedores. En concreto, los clientes no deben colocar procedimientos en el árbol de paquetes **com.unica** o **com.unicacorp** .
- La implementación del procedimiento se codifica en la versión de Java Runtime utilizada por IBM Unica Marketing Operations en el servidor de aplicaciones (como mínimo, JRE 1.5.10).
- IBM Unica Marketing Operations proporciona algún número de bibliotecas de código abierto y de otros proveedores; los servidores de aplicaciones también utilizan distintas versiones de estas bibliotecas. Por regla general, esta lista cambia de release a release.

Nota: Para evitar posibles problemas de compatibilidad, los procedimientos no deben utilizar ninguna biblioteca de código abierto, de otros proveedores o específica del servidor de aplicaciones.

Sin embargo, si dichos paquetes son utilizados por un procedimiento, o por las clases secundarias importadas por el procedimiento, su uso debe coincidir exactamente con los paquetes proporcionados por Marketing Operations y/o el servidor de aplicaciones. Tenga en cuenta que en este caso, puede tener que volver a trabajar en el código del procedimiento, si una versión posterior de Marketing Operations actualiza o abandona una biblioteca.

- La clase de implementación del procedimiento se carga mediante la política de carga de clases utilizada normalmente por IBM Unica Marketing Operations (por regla general, **padre-último**). El servidor de aplicaciones puede proporcionar opciones y herramientas de desarrollo para volver a cargar las clases que se aplicarán a los procedimientos de Marketing Operations, pero esto no es necesario.
- El procedimiento debe tener las hebras protegidas con respecto a su propio estado; es decir, su método de ejecución no pueden depender de los cambios de estado interno de una llamada a otra.
- Un procedimiento no puede crear hebras por su cuenta.

Diseño

El diseño se debe centrar en la producción de una sola unidad de trabajo que se debe realizar de forma atómica. De forma ideal, el procedimiento realiza alguna cadena de tareas que se pueden planificar para ejecutarse de forma asíncrona más adelante; este modelo de integración de "activar y olvidar" se produce en la última carga en ambos sistemas.

La clase de implementación del procedimiento utiliza la API de IBM Unica Marketing Operations para leer y actualizar los componentes de Marketing Operations, invocar servicios, etc. Otros paquetes de Java se pueden utilizar para realizar otras tareas.

Nota: Sólo las clases y los métodos documentados estarán soportados en futuros releases de Marketing Operations. Todas las demás clases y métodos en Marketing Operations se deberán considerar fuera de los límites.

Una vez codificadas y compiladas, las clases de implementación de procedimiento deben estar disponibles en Marketing Operations. Los scripts de creación proporcionados con los servicios de integración de Marketing Operations colocan los procedimientos compilados en la ubicación predeterminada. El paso final de desarrollo es actualizar el archivo de definiciones del plugin del procedimiento personalizad utilizado por Marketing Operations para descubrir los procedimientos personalizados.

El procedimiento debe implementar la interfaz **com.unica.publicapi.plan.plugin.procedure.IProcedure** y tener un constructor parameter-less (modelo JavaBean usual). La codificación y compilación de cada procedimiento se realiza en un IDE Java de elección del cliente como, por ejemplo, Eclipse, Borland JBuilder, Idea, etc. El código de ejemplo se proporciona con IBM Unica Marketing Operations como kits de herramientas de desarrollador en la siguiente ubicación.

<Marketing Operations-home>/devkits/integration/examples/src/procedure

Configuración

Utilice los parámetros en **Valores > Configuración > Marketing Operations > umoConfiguration > integrationServices** para configurar el módulo de integración de Marketing Operations.

Si desea ver detalles, consulte la publicación *Marketing Operations Guía de instalación*.

Ciclo de vida de procedimiento

El ciclo de vida de tiempo de ejecución de un procedimiento es:

1. Descubrimiento e inicialización
2. Selección (opcional)
3. Ejecución
4. Destrucción

Descubrimiento e inicialización

IBM Unica Marketing Operations deben conocer todos los procedimientos personalizados y estándar disponibles para una instancia de instalación particular. Este proceso se llama descubrimiento.

Nota: Los procedimientos estándar (procedimientos definidos por el equipo de ingeniería de Marketing Operations) son conocidos de forma implícita y, por lo tanto, no necesitan ninguna acción para ser descubiertos.

Los procedimientos personalizados están definidos en el archivo de definición del plugin de procedimiento. El gestor de plugin de Marketing Operations lee este archivo durante la inicialización. Para cada procedimiento encontrado, el gestor de plugin hace lo siguiente:

1. Crea una instancia del procedimiento; se cambia a su estado a INSTANTIATED.
2. Crea un nuevo registro de auditoría de procedimiento.
3. Si no se ha podido crear una instancia del procedimiento, se llama a su método **`initialize()`** con los parámetros de inicialización encontrados en su archivo de descripción de plugin. Si este método lanza una excepción, se registra el estado y se abandona el procedimiento. De lo contrario, el procedimiento pasa al estado INITIALIZED. Ahora está preparado para ejecutarse.
4. Crea un nuevo registro de auditoría de procedimiento.
5. Si no se ha podido inicializar el procedimiento, se llama al método **`getKey()`** para determinar la clave utilizada por los clientes para hacer referencia al procedimiento. Esta clave se asocia a la instancia y se guarda para una búsqueda posterior.

Selección

De vez en cuando, es posible que IBM Unica Marketing Operations necesite presentar una lista de los procedimientos disponibles a los usuarios, por ejemplo, para permitir a los administradores configurar un desencadenante. Esto sólo se realiza después de que se haya inicializado el procedimiento. Los métodos **`getDisplayName()`** y **`getDescription()`** del procedimiento se utilizan con este propósito.

Ejecución

En algún momento después de que se haya inicializado el procedimiento, IBM Unica Marketing Operations recibe una solicitud para ejecutar el procedimiento. Esto puede suceder de forma simultánea con otros procedimientos (o el mismo procedimiento) que se ejecutan en otras hebras.

Durante la ejecución, el gestor de ejecución de procedimiento realiza lo siguiente.

1. Inicia una transacción de base de datos.
2. Establece el estado del procedimiento en EXECUTING.
3. Crea un nuevo registro de auditoría de procedimiento.
4. Llama al método **execute()** del procedimiento con un contexto de ejecución y cualquier parámetro de ejecución proporcionado por el cliente. La implementación del método utiliza la API de Marketing Operations según sea necesario, adquiriendo bloqueos de edición y pasándolos junto al contexto de ejecución. Si el método de ejecución lanza una excepción, el gestor de ejecución marca la transacción para la retrotracción.
5. Confirma o retrotrae la transacción de acuerdo con los resultados de ejecución; establece el estado del procedimiento en EXECUTED.
6. Libera los bloqueos de edición pendientes.
7. Crea un nuevo registro de auditoría de procedimiento.

Nota: El método **execute()** no debe alterar los datos de la instancia del procedimiento.

Destrucción

Cuando IBM Unica Marketing Operations concluye, el gestor de plugin de procedimiento pasa por todos los procedimientos cargados. Para cada procedimiento encontrado, realiza lo siguiente:

1. Llama al método destroy() del procedimiento para permitir al procedimiento limpiar antes de que se destruya la instancia.
2. Cambia el estado de procedimiento a FINALIZED (no se puede ejecutar).
3. Crea un nuevo registro de auditoría de procedimiento.
4. Destruye la instancia del procedimiento.

Bloqueo de datos

IBM Unica Marketing Operations utiliza un esquema de bloqueo de edición pesimista; es decir, sólo un usuario está autorizado para un acceso de actualización a instancias de componente a la vez. Para el usuario de la GUI, este bloqueo se realiza en el nivel de ficha visual. En algunos casos, esto representa un subconjunto de una instancia (por ejemplo, una ficha de resumen de proyecto), mientras que en otros casos, representa muchas instancias (la ficha de flujo de trabajo). Una vez que un usuario adquiere un bloqueo, todos los demás usuarios están restringidos a un acceso de sólo lectura a los datos relacionados.

Nota: Los bloqueos de edición no son lo mismo que las transacciones de base de datos.

Para poder garantizar que otro usuario no sobrescriba sin querer los cambios realizados por un procedimiento en una instancia de componente o un grupo de instancias, un procedimiento debe adquirir los bloqueos apropiados antes de

actualizar los datos de componente. El objeto de contexto de ejecución pasado al método **execute()** de procedimiento se utiliza para conseguir esto.

Antes de que el procedimiento actualice ningún dato, debe llamar al método **acquireLock()** del contexto para cada bloqueo que necesite. Por ejemplo, si un procedimiento va a actualizar un proyecto y el flujo de trabajo asociado, el procedimiento debe adquirir bloqueos para ambos.

Si otro usuario ya tiene un bloqueo, el método **acquireLock()** lanza una excepción **LockInUseException** inmediatamente. Para poder minimizar colisiones, el procedimiento debe liberar el bloqueo tan pronto como actualice el objeto.

El gestor de ejecución libera automáticamente los bloqueos pendientes cuando vuelve el método de ejecución. En cualquier caso, los bloqueos sólo se conservan durante la vida de la transacción de base de datos. Es decir, los bloqueos caducan si se excede el tiempo de espera de la transacción específica de la base de datos.

Transacciones de procedimiento

El gestor de ejecución de procedimiento envuelve automáticamente la ejecución de un procedimiento con una transacción de base de datos, confirmándola o retrotrayéndola según sea apropiado basándose en el resultado de la ejecución del procedimiento. Así se garantiza que las actualizaciones en la base de datos IBM Unica Marketing Operations no son visibles para otros usuarios, hasta la confirmación y que las actualizaciones son atómicas.

El grabador del procedimiento debe seguir adquiriendo los bloqueos de edición necesarios para asegurarse de que los otros usuarios no pueden grabar cambios en la base de datos antes de que se complete la ejecución del procedimiento.

Comunicación de resultados

El método **execute()** de un procedimiento devuelve un código de estado entero y cero o más mensajes que se registran y se conservan en la tabla de auditoría de procedimiento de IBM Unica Marketing Operations. El cliente también puede comunicar la información de estado de algún otro modo.

Registro del procedimiento

IBM Unica Marketing Operations tiene un archivo de registro independiente para los procedimientos.

<Marketing Operations-home>\logs\procedure.log

El gestor de ejecución de procedimiento registra el ciclo de vida de cada procedimiento y crea registros de auditoría.

- **logInfo()**: graba un mensaje informativo en el registro de procedimiento
- **logWarning()**: graba un mensaje de aviso en el registro de procedimiento
- **logError()**: graba un mensaje de error en el registro de procedimiento
- **logException()**: vuelca el seguimiento de pila para la excepción en el registro de procedimiento

Clases Java clave

El kit de desarrollo de integración proporcionado contiene un conjunto de Javadoc para las clases de soporte y la API pública de IBM Unica Marketing Operations. Las más importantes se listan aquí.

- IProcedure (com.unica.publicapi.plan.plugin.procedure.IProcedure): interfaz que todos los procedimientos deben implementar. Los procedimientos pasan a través de un ciclo de vida bien definido y acceden a la API de Marketing Operations para realizar el trabajo.
- ITriggerProcedure (com.unica.publicapi.plan.plugin.procedure.ITriggerProcedure): interfaz que todos los procedimientos de desencadenante deben implementar (interfaz de marcador).
- IExecutionContext (com.unica.publicapi.plan.plugin.procedure.IExecutionContext): interfaz de objeto de contexto opaco transferido al procedimiento mediante el gestor de ejecución. Este objeto tiene métodos públicos para registrar y editar la gestión de bloqueos. El procedimiento también pasa este objeto a todas las llamadas a PlanAPI.
- IPlanAPI (com.unica.publicapi.plan.api.IPlanAPI): interfaz en la API de Marketing Operations. El contexto de ejecución proporciona un método **`getPlanAPI()`** para recuperar la implementación correcta.

Ejemplo de procedimiento

Este ejemplo muestra un procedimiento estándar para cambiar el estado de un proyecto desde un servicio web de integración o un desencadenante.

Nota: No modifique los procedimientos de ejemplo y sus definiciones XML, ya que los ejemplos se sobrescriben cuando se actualiza IBM Unica Marketing Operations y perderá los cambios. Debe crear y modificar todos los procedimientos personalizados en un directorio diferente.

```
// ProjectStateChangeProcedure
// (c) Copyright 2006 by Unica Corporation. Reservados todos los derechos.

package com.unica.uap.plugin.procedure.standard;

import java.util.Iterator;
import java.util.List;
import java.util.Locale;
import java.util.Map;

import com.unica.publicapi.plan.api.Handle;
import com.unica.publicapi.plan.api.IExecutionContext;
import com.unica.publicapi.plan.api.IPlanAPI;
import com.unica.publicapi.plan.api.LockInUseException;
import com.unica.publicapi.plan.api.ProjectHandle;
import com.unica.publicapi.plan.api.ProjectStateEnum;
import com.unica.publicapi.plan.plugin.PluginVersion;
import com.unica.publicapi.plan.plugin.procedure.IProcedure;
import com.unica.publicapi.plan.plugin.procedure.ProcedureExecutionException;
import com.unica.publicapi.plan.plugin.procedure.ProcedureInitializationException;
import com.unica.publicapi.plan.plugin.procedure.ProcedureMessage;
import com.unica.publicapi.plan.plugin.procedure.ProcedureMessageTypeEnum;
import com.unica.publicapi.plan.plugin.procedure.ProcedureResult;

/**
 * <b>ProjectStateChangeProcedure</b> es un procedimiento Marketing Operations estándar
 * que intenta la
 * transición del estado de un proyecto.
```

```
 * <p>
 * Espera los siguientes parámetros de inicialización:
 * <ul>
 *   <li>debug: Objeto booleano, <tt>true</tt> o <tt>false</tt>, que indica
 *   si el rastreo de depuración está habilitado o no</li>
 * </ul>
 *
 * <p>
 * Espera los siguientes parámetros de ejecución:
 * <ul>
 *   <li>hProject: formato de matriz de cadenas del manejador del proyecto, por ejemplo,
 *     "http://mymachine:7001/MktOps/affiniumplan.jsp?
 *         cat=projecttabs&projectid=12"</li>
 *   <li>uapState: formulario de matriz de cadena de un nuevo estado de proyecto, por ejemplo,
 *      "COMPLETED". Tenga en cuenta que es importante distinguir mayúsculas y minúsculas.</li>
 * </ul>
 *
 */
public final class ProjectStateChangeProcedure implements IProcedure {
// parámetros de inicialización
private final static String DEBUG_INITPARAMETER_NAME = "debug";
  // parámetros de ejecución
  private final static String HPROJECT_PARAMETER_NAME = "hProject";
  private final static String STATE_PARAMETER_NAME
    = IPlanAPI.PROJECT_ATTRIBUTE_STATEENUM; // al igual que el nombre de atributo

  // nuestros códigos de estado
  private final static int STATUS_SUCCESS = 0;

    // propiedad de depuración. Establezca el parámetro de inicialización "debug" del procedimient
  //   en true para habilitar el rastreo de depuración
    private boolean _debug = false;
    private boolean isDebug() { return _debug; }

  // el nombre simple es el nombre de clase sin calificar
  public String getName() {
        return "uapProjectStateChangeProcedure";
    }

  // el nombre de visualización siempre es clave
  public String getDisplayName(Locale locale) {
        // por ahora sólo es para EN
        return getName();
    }

  // la descripción siempre en inglés
    public String getDescription(Locale locale) {
        // por ahora sólo es para EN
          return "Un procedimiento para la transición del estado de un proyecto.";
    }

  // la versión a la que hemos codificado; debe ser 1.0.0 por ahora
  public PluginVersion getVersion() {
    return new PluginVersion(1,0,0);
  }

  // inicializar instancia a partir de parámetros de inicialización
    public void initialize(Map initParameters)
      throws ProcedureInitializationException {
    // el único parámetro que tenemos es: debug, Boolean
        if (initParameters.containsKey(DEBUG_INITPARAMETER_NAME)) {
            try {
                _debug = ((Boolean)initParameters.get(DEBUG_INITPARAMETER_NAME)).
                       booleanValue();
            } catch (Exception exception) {
                throw new ProcedureInitializationException("Problema al utilizar "
                                    + DEBUG_INITPARAMETER_NAME
```

```
                                        + " init parameter: "
                                        + exception.getMessage());
            }
            }
        }

      // ejecutar: expect hProject and state enum
        public ProcedureResult execute(IExecutionContext context, Map parameters)
          throws ProcedureExecutionException {
            // obtener parámetros de ejecución: esperamos dos:
            // - hProject: string[] formulario del manejador del proyecto
            // - uapState: cadena[] formulario de ProjectStateEnum
            ProjectHandle hProject = null;
        if (parameters.containsKey(HPROJECT_PARAMETER_NAME)) {
          try {
            hProject = (ProjectHandle)
    Handle.makeHandle(((String[])parameters.get(HPROJECT_PARAMETER_NAME))[0]);
          } catch (Exception exception) {
            throw new ProcedureExecutionException("Problema al utilizar "
                                + HPROJECT_PARAMETER_NAME
                                + " parameter: "
                                + exception.getMessage());
          }
        } else throw new ProcedureExecutionException(HPROJECT_PARAMETER_NAME
                                + " se debe proporcionar parámetro.");

            ProjectStateEnum stateEnum = null;
        if (parameters.containsKey(STATE_PARAMETER_NAME)) {
          try {
            stateEnum =
              ProjectStateEnum.valueOf(((String[])parameters.
                get(STATE_PARAMETER_NAME))[0]);
          } catch (Exception exception) {
            throw new ProcedureExecutionException("Problema al utilizar "
                                + STATE_PARAMETER_NAME
                                + " parameter: "
                                + exception.getMessage());
          }
        } else throw new ProcedureExecutionException(STATE_PARAMETER_NAME
                                + " se debe proporcionar parámetro.");

        int status = -1;
        ProcedureMessage[] messages = null;
            try {
               // intente adquirir un bloqueo de edición para el proyecto
                context.acquireLock(hProject, IExecutionContext.LOCK_ALL_FIELDS);

                // utilizar PlanAPIImpl para actualizar el estado
                IPlanAPI planAPI = context.getPlanAPI();
                planAPI.updateAttribute(context, hProject, STATE_PARAMETER_NAME,
                            new ProjectStateEnum[]{stateEnum});

               // éxito
              status = STATUS_SUCCESS;

            } catch (Exception exception) {
              // escribir la pila de rastreo si hay depuración
                if (isDebug()) {
                    context.logError(getName(), exception);
                }
                throw new ProcedureExecutionException(exception);
            } finally {
              // liberar nuestro bloqueo
              try {
                context.releaseAllLocks();
              } catch (Exception exception) { /* ignored */ }
            }
```

```
        return new ProcedureResult(status, messages);
    }

    public void destroy( ){
         // no necesitamos hacer nada más
    }
}
```

Archivo de definiciones del plugin de procedimiento

En este tema se describe el archivo de definiciones del plug-in de procedimiento. Este archivo define la clase de implementación, los metadatos y demás información sobre los procedimientos personalizados que se van a alojar en IBM Unica Marketing Operations. De forma predeterminada, se da por sentado que la definición del plugin del procedimiento está en la siguiente ruta:

```
<Marketing Operations-home>/devkits/integration/examples/src/
  procedures/procedure-plugins.xml
```

Este archivo es un documento XML que contiene la información descrita a continuación.

Procedimientos: una lista de cero o más elementos **Procedure**.

Procedimiento: un elemento que define un procedimiento. Cada procedimiento contiene los siguientes elementos.

- **key** (opcional): una cadena que define la clave de búsqueda para el procedimiento. Esta clave debe ser exclusiva con respecto a todos los procedimientos personalizados y estándar (proporcionados por IBM) alojados por una instancia de Marketing Operations determinada. Si no se ha definido, adopta de forma predeterminada el valor de la versión completa del elemento **className**. Los nombres que empiezan con la cadena "uap" se reservan para ser utilizados por IBM Unica Marketing Operations.
- **className** (obligatorio): nombre de paquete completo de la clase del procedimiento. Esta clase debe implementar la clase IProcedure (com.unica.public.plan.plugin.procedure.IProcedure).
- **initParameters** (opcional): una lista de cero o más elementos initParameter.

  initParameter(opcional): parámetro que debe transferirse al método de inicializar() procedimiento. Este elemento incluye el nombre de parámetro anidado, el tipo y los elementos de valor.
  - name: cadena que define el nombre del parámetro
  - type: nombre de clase opcional de la clase de envoltorio Java que define el tipo del valor de parámetro. Debe adoptar uno de los siguientes tipos:
    - java.lang.String (el valor predeterminado)
    - java.lang.Integer
    - java.lang.Double
    - java.lang.Calendar
    - java.lang.Boolean
  - valor: forma de cadena del valor del atributo de acuerdo con su tipo

Capítulo 4. Acerca de la API de IBM Unica Marketing Operations

La API de IBM Unica Marketing Operations es una fachada que proporciona una vista de cliente de una instancia de Marketing Operations en ejecución. Sólo se expone un subconjunto de las capacidades de Marketing Operations. La API se ha diseñado para ser utilizada de forma simultánea por los usuarios web de Marketing Operations y las solicitudes SOAP y los desencadenantes del servicio web de servicio de integración de Marketing Operations. La API soporta los siguientes tipos de operaciones.

- Creación y supresión de componente
- Descubrimiento (por tipo de componente, valor de atributo, etc.)
- Inspección de componente (a través de sus atributos, enlaces especializados, etc.)
- Modificación de componente

Versiones y compatibilidad con versiones anteriores

Las futuras versiones de esta API serán compatibles con versiones anteriores con todos los releases menores y de mantenimiento que compartan un número de versión principal. Sin embargo, IBM se reserva el derecho de romper la compatibilidad con una versión anterior para releases principales de punto cero (x.0) si el caso empresarial o técnico lo garantiza.

El número de versión principal de esta API se incrementará si se realiza cualquiera de los siguientes cambios.

- La interpretación de datos ha cambiado
- La lógica empresarial ha cambiado (por ejemplo, la funcionalidad del método de servicio ha cambiado)
- Los parámetros de método y/o tipos de retornos han cambiado

El número de versión menor de la API se incrementará si se realiza cualquiera de los cambios siguientes (tenga en cuenta, que estos cambios son, por definición, compatibles con las versiones anteriores).

- Se ha añadido un método nuevo
- Se ha añadido un nuevo tipo de datos y su uso está restringido a un método nuevo
- Se ha añadido un elemento nuevo a un tipo enumerado
- Se ha definido una nueva versión con un sufijo de versión

Seguridad de usuario

Se da por supuesto que la autenticación la va a realizar el gestor de ejecución del procedimiento y la información del usuario autenticado se enlaza al contexto de ejecución utilizado por todas las API. La API no expone el usuario autenticado, pero lo pasará a IBM Unica Marketing Operations para utilizarlo según proceda.

Sin embargo, es posible que el usuario autenticado no esté autorizado para realizar todas las operaciones expuestas por la API; en este caso, el método de la API lanzará una excepción **`AuthorizationException`**.

Configuración regional

La única configuración regional soportada por esta versión es la configuración regional configurada actualmente para la instancia del servidor IBM Unica Marketing Operations. Todos los datos que dependen de la configuración regional a los que se acceder a través de la API (mensajes, moneda, etc) se da por supuesto que están en la configuración regional de este sistema.

Gestión de estados

Esta API no tiene estados, lo que significa que la API no guarda ninguna información por cliente entre las llamadas.

Sin embargo, tenga en cuenta que las llamadas de API específicas pueden cambiar el estado de las instancias de componente subyacentes gestionadas por IBM Unica Marketing Operations, y estos cambios de estado se pueden conservar en una base de datos.

Transacciones de base de datos

Esta API no expone las transacciones de base de datos en el cliente, pero utilizará dicha información si está incluida en el contexto de ejecución. Si se ha iniciado una transacción, el efecto de todas las llamadas de API dentro de un procedimiento concreto será atómico. Los demás usuarios de IBM Unica Marketing Operations no verán los cambios hasta que el procedimiento confirme satisfactoriamente la transacción.

Las llamadas de API que actualizan la base de datos deben, en primer lugar, adquirir un bloqueo de edición para evitar que otros usuarios de Marketing Operations modifiquen los datos subyacentes durante el transcurso de las llamadas de API. Los demás usuarios no podrán actualizar los componentes bloqueados hasta que se complete la API; de igual forma, otro usuario o cliente de la API de Marketing Operations puede haber adquirido el bloqueo sobre los datos deseados que impedirá que se complete la llamada de API.

Proceso de eventos

Las operaciones realizadas en los componentes de IBM Unica Marketing Operations a través de esta API generan los mismos eventos que si la operación hubiera sido realizada por un usuario web de Marketing Operations. En particular, los desencadenantes que esperan determinados eventos se activarán al final en ambos casos. A los usuarios suscritos a determinadas notificaciones (por ejemplo, cuando cambia el estado de un proyecto) se les notificarán los cambios de estado que se derivan de llamadas de API, así como de acciones de usuarios web.

Acerca de los tipos de datos de API

La API de IBM Unica Marketing Operations contiene los siguientes tipos de datos públicos.

- **ApprovalMethodEnum**: Tipo enumerado para métodos de aprobación.
- **ApprovalStateEnum**: Tipo enumerado para estado de aprobación.
- **AssetLibraryStateEnum**: Tipo enumerado para estados de biblioteca de activos.
- **AssetStateEnum**: Tipo enumerado para estados de aserción.
- **AttachmentTypeEnum**: Tipo enumerado para archivo adjunto.

- **AttributeMap**: Una correlación Java que contiene atributos y sus valores.
- **BudgetPeriodEnum**: Tipo enumerado para tipos de periodo de presupuesto.
- **BudgetTypeEnum**: Tipo enumerado para tipos de presupuesto
- **Handle**: Identifica una instancia de componente en una instancia de Marketing Operations particular.
- **InvoiceStateEnum**: Tipo enumerado para estados de factura.
- **MonthEnum**: Tipo enumerado para meses de un año natural.
- **ProjectCopyTypeEnum**: Tipo enumerado para opciones de proyecto de copia.
- **ProjectStateEnum**: Tipo enumerado para estados de proyecto.
- **TaskStateEnum**: Tipo enumerado para estados de tarea.
- **QuarterEnum**: Tipo enumerado para trimestres de un año natural.
- **WeekEnum**: Tipo enumerado de semanas de un año natural.

Los valores posibles para cada tipo de datos se mencionan abajo.

Tipos enumerados

ApprovalMethodEnum

ApprovalMethodEnum es un tipo enumerado que define todos los métodos posibles de una aprobación. Los valores posibles son:

- SIMULTANEOUS
- SEQUENTIAL

ApprovalStateEnum

ApprovalStateEnum es un tipo enumerado que define todos los estados posibles de una aprobación. Los valores posibles son:

- NOT_STATED
- ON_HOLD
- IN_PROGRESS
- COMPLETED
- CANCELLED

AssetLibraryStateEnum

AssetLibraryStateEnum es un tipo enumerado que define todos los estados posibles de una biblioteca de activos. Los valores posibles son:

- ENABLED
- DISABLED

AssetStateEnum

AssetStateEnum es un tipo enumerado que define todos los estados posibles de un activo. Los valores posibles son:

- DRAFT
- LOCK
- FINALIZE
- ARCHIVE

AttachmentTypeEnum

AttachmentTypeEnum es un tipo enumerado que define todos los tipos posibles de un archivo adjunto. Los valores posibles son:

- URL
- FILE
- ASSET

BudgetPeriodEnum

BudgetPeriodEnum es un tipo enumerado que define todos los periodos de presupuesto posibles. Los valores posibles son:

- QUARTERLY
- MONTHLY
- WEEKLY
- YEARLY
- ALL

BudgetTypeEnum

BudgetTypeEnum es un tipo enumerado que define todos los tipos de presupuesto posibles. Los valores posibles son:

- TOTAL
- FORECAST
- COMMITTED
- ACTUAL
- ALLOCATED

InvoiceStateEnum

InvoiceStateEnum es un tipo enumerado que define todos los estados posibles de una factura. Los valores posibles son:

- DRAFT
- PAYABLE
- PAID
- CANCELLED

MonthEnum

MonthEnum es un tipo enumerado que define todos los valores posibles para mes.

ProjectCopyTypeEnum

ProjectCopyTypeEnum es un tipo enumerado que define todos los tipos posibles de proyecto de copia. Los valores posibles son:

- COPY_USING_PROJECT_METRICS
- COPY_USING_TEMMPLATE_METRICS

Si desea ver detalles sobre estos estados de proyecto y tarea, consulte la publicación *Marketing Operations Guía de usuario*.

ProjectStateEnum

ProjectStateEnum es un tipo enumerado que define todos los estados posibles de un proyecto o solicitud. Los valores posibles son:

- NO_INICIADO
- IN_PROGRESS
- ON_HOLD
- IN_RECONCILIATION
- LATE: indica que el proyecto no se ha iniciado su fecha de inicio planificado.
- OVERDUE: indica que el proyecto no se ha completado antes de fecha de finalización planificada.
- DRAFT
- SUBMITTED
- RETURNED
- ACCEPTED
- COMPLETED
- CANCELLED
- DELETED

TaskStateEnum

TaskStateEnum es un tipo enumerado que define todos los estados posibles de una tarea de flujo de trabajo. Los valores posibles son:

- PENDING
- ACTIVE
- FINISHED
- SKIPPED
- DISABLED

QuarterEnum

QuarterEnum es un tipo enumerado que define todos los valores posibles para trimestre.

WeekEnum

WeekEnum es un tipo enumerado que define todos los valores posibles para semana.

Manejador

Un Manejador es un objeto de URL especial que hace referencia a una instancia de componente en una instancia concreta de Marketing Operations. Los Manejadores incluyen el tipo de componente, el identificador de datos internos, un URL base de instancia, etc. Los Manejadores utilizados o generados por la API se pueden externalizar en un URL completo que se puede utilizar para ir a una vista del componente en la GUI de Marketing Operations, se pueden cortar y pegar, enviar en correos electrónicos, se pueden utilizar en otro procedimiento como parámetro, etc.

Los Manejadores sólo son válidos para una instancia de servicio Marketing Operations concreta, o una instancia en clúster, pero se garantiza que son válidos durante todo el ciclo de vida del servicio desplegado. Como resultado, los Manejadores se pueden guardar en un archivo para una referencia posterior, pero

no se pueden utilizar para acceder a componentes en otra instancia de Marketing Operations (aunque estén en la misma máquina). Sin embargo, Marketing Operations proporciona un mecanismo para correlacionar distintos URL base con la instancia actual para dar cabida a la reubicación de una instancia a otra máquina (por ejemplo, en el caso de que la máquina original no funcione correctamente).

Los Manejadores dependen del cliente. Por ejemplo, un desencadenante puede pasar un Manejador a un procedimiento, que lo utiliza como parámetro en una llamada SOAP a un sistema de otro proveedor, que da una vuelta y vuelve a emitir una solicitud SOAP a Marketing Operations para invocar a un procedimiento que actualiza un atributo.

La clase de Manejador tiene métodos de fábrica para crear manejadores a partir de distintos tipos de URL.

Objeto de aprobación:

```
http://mymachine:7001/MktOps/affiniumplan.jsp?cat=approvaldetail&approvalid=101
```

Objeto AssetLibrary:

```
http://mymachine:7001/MktOps/affiniumplan.jsp?cat=library&id=101
```

Objeto de carpeta de activos:

```
http://mymachine:7001/MktOps/affiniumplan.jsp?cat=folder&id=101
```

Objeto de activo:

```
http://mymachine:7001/MktOps/affiniumplan.jsp?cat=asset&assetMode=VIEW_ASSET
  &assetid=101
```

Objeto de archivo adjunto:

```
http://mymachine:7001/MktOps/affiniumplan.jsp?cat=attachmentview&
  attachid=101&parentObjectId=101&parentObjectType=project
```

Objeto de cuenta financiera:

```
http://mymachine:7001/MktOps/affiniumplan.jsp?cat=accountdetails&accountid=101
```

Objeto de elemento de línea de factura:

```
http://mymachine:7001/MktOps/affiniumplan.jsp?cat=invoicedetails&
  invoiceid=134&line_item_id=101
```

Objeto de factura:

```
http://mymachine:7001/MktOps/affiniumplan.jsp?cat=invoicedetails&invoiceid=134
```

Objeto de marketing:

```
"http://mymachine:7001/MktOps/affiniumplan.jsp?cat=componenttabs&
  componentid=creatives &componentinstid=1234"
```

Cuadrícula de objeto de marketing:

```
"http://mymachine:7001/MktOps/affiniumplan.jsp?cat=componenttabs&
  componentid=creatives &componentinstid=1234&gridid=grid"
```

Fila de cuadrícula de objeto de marketing:

```
"http://mymachine:7001/MktOps/affiniumplan.jsp?cat=componenttabs&
  componentid=creatives &componentinstid=1234&gridid=grid&gridrowid=101"
```

Proyecto:

```
"http://mymachine:7001/MktOps/affiniumplan.jsp?cat=projecttabs&projectid=1234"
```

Cuadrícula de proyecto:

```
"http://mymachine:7001/MktOps/affiniumplan.jsp?cat=projecttabs&
  projectid=1234&gridid=grid"
```

Fila de cuadrícula de proyecto:

```
"http://mymachine:7001/MktOps/affiniumplan.jsp?cat=projecttabs&
  projectid=1234&gridid=grid &gridrowid=101"
```

Objeto de elemento de línea de proyecto:

```
"http://mymachine:7001/MktOps/affiniumplan.jsp?cat=projecttabs&
  projectid=1234&projectlineitemid=123&projectlineitemisversionfinal=false"
```

Objeto de equipo:

```
http://mymachine:7001/MktOps/affiniumplan.jsp?cat=teamdetails&
  func=edit&teamid=100001
```

Objeto de usuario:

```
http://mymachine:7001/MktOps/affiniumplan.jsp?cat=adminuserpermissions&
  func=edit&userId=101
```

Tarea de flujo de trabajo:

```
"http://mymachine:7001/MktOps/affiniumplan.jsp?cat=projectworkflow&
  projectid=1234&taskid=5678"
```

Correlación de atributos

Un AttributeMap es una correlación que sólo contiene atributos. El atributo *name* es la clave de entrada de correlación y el atributo de matriz de *values* (tener en cuenta que es plural) es el valor de entrada de la correlación.

AttributeMap incluye los siguientes campos:

- *Nombre*: el nombre programático del atributo. Este nombre sirve como una clave única para acceder al atributo dentro de la instancia de componente en la que se produce.

  Nota: El *nombre* no es necesariamente el nombre de visualización presentado al usuario en la GUI. Para los componentes creados con plantillas (como, por ejemplo, proyectos o tareas de flujo de trabajo), el nombre de atributo se especifica mediante la definición del elemento de plantilla y debe ser exclusivo. Para otros componentes, el nombre del atributo normalmente se suele derivar mediante programa de la instancia del componente del lado del servidor (por ejemplo, a través de una introspección Java).

  Nota: Por convenio, los atributos personalizados incluyen el nombre del formulario en el cual está definida la versión editable: <nombre_formulario>.<nombre_atributo>.
- *valores*: una matriz de objetos Java que contiene cero o más valores de atributo. El tipo de cada valor debe ser el mismo y estar conforme al tipo de atributo tal como está definido en Marketing Operations. Sólo están soportados el envoltorio Java y los tipos Marketing Operations:
  - AssetLibraryStateEnum: un valor de tipo enumerado AssetLibraryStateEnum.
  - AssetStateEnum: un valor de tipo enumerado AssetStateEnum.
  - AttachmentTypeEnum: un valor de tipo enumerado AttachmentTypeEnum.

- AttributeMap: una correlación que contiene atributos.
- BudgetPeriodEnum: un valor de tipo enumerado BudgetPeriodEnum.
- BudgetTypeEnum: un valor de tipo enumerado BudgetTypeEnum.
- Handle: una referencia a una instancia de componente, fila de cuadrícula, atributo, etc.
- InvoiceStateEnum: un valor de tipo enumerado InvoiceStateEnum.
- java.io.File: representación de un archivo.
- java.lang.Boolean: un valor booleano, ya sea True o False
- java.lang.Double: un valor de número decimal de doble precisión.
- java.lang.Float: un valor de número decimal de precisión única
- java.lang.Integer: un valor entero de 32 bits
- java.lang.Long: un valor entero de 64 bits
- java.lang.Object: objeto Java genérico
- java.lang.String: una cadena de cero o más caracteres Unicode
- java.math.BigDecimal: valor de número decimal firmado de precisión arbitraria. Apto para divisas; la interpretación del valor depende de la configuración regional de la moneda para el cliente.
- java.math.BigInteger: valor entero de precisión arbitraria.
- java.net.URL: un objeto URL (Universal Resource Locator).
- java.util.ArrayList: lista de objetos.
- java.util.Calendar: una valor de fecha y hora para una configuración regional particular.
- java.util.Date: un valor de fecha y hora. Este tipo está en desuso. Utilice en su lugar java.util.Calendar o java.util.GregorianCalendar.
- java.util.GregorianCalendar: GregorianCalendar es una subclase concreta de java.util.Calendar y proporciona el sistema de calendario estándar más utilizado en el mundo.
- MonthEnum: un valor de tipo enumerado MonthEnum.
- ProjectStateEnum: un valor de tipo enumerado ProjectStateEnum.
- QuarterEnum: un valor de tipo enumerado QuarterEnum.
- TaskStateEnum: un valor de tipo enumerado TaskStateEnum.
- WeekEnum: un valor de tipo enumerado WeekEnum.

Los metadatos de un atributo (por ejemplo, la descripción y el nombre de visualización localizados) se definen mediante la plantilla asociada al atributo y su instancia de objeto padre. Los atributos proporcionan un mecanismo sencillo pero ampliable para exponer ambos atributos de instancia de objeto, los obligatorios y los opcionales como, por ejemplo, nombre de proyecto, código y fecha de inicio.

Nota: Para implementar la fecha, los usuarios puede seleccionar java.util.Calendar o java.util.GregorianCalendar.

Excepciones comunes

En este tema se describen algunas excepciones comunes lanzadas por la API.

- AuthorizationException: El usuario asociado al contexto de ejecución del cliente no está autorizado para realizar la operación solicitada. Esta excepción puede ser lanzada por cualquier método de API, de modo que no se declara.

- DataException: Se ha producido una excepción en la capa de base de datos subyacente en IBM Unica Marketing Operations. Consulte el registro SQL para ver detalles.
- InvalidExecutionContextException: Hay un problema con un contexto de ejecución pasado a un método de API (por ejemplo, el método no se inicializó correctamente). Esta excepción puede ser lanzada por cualquier API, de modo que no se declara.
- NotLockedException: Se ha intentado actualizar los datos de componente sin adquirir primero el bloqueo necesario. Consulte el método acquireLock() de IExecutionContext.

Método de API

Si desea información específica sobre los métodos de API públicos, consulte la clase iPlanAPI en los archivos de documentación de API JavaDocs. Estos archivos están disponibles iniciando sesión en Marketing Operations y seleccionando **Ayuda y documentación del producto** desde cualquier página y descargando el archivo *<version>*PublicAPI.zip.

Cómo contactar con el soporte técnico de IBM Unica

Si encuentra un problema que no puede resolver consultando la documentación, el contacto responsable del soporte técnico puede registrar una llamada con el soporte técnico de IBM Unica . Utilice la información de este apartado para asegurarse de que el problema se resuelve de forma eficaz y satisfactoria.

Si usted no es el contacto responsable del soporte técnico de su empresa, póngase en contacto con el administrador de IBM Unica para obtener información.

Información a recopilar

Antes de ponerse en contacto con el soporte técnico de IBM Unica , recopile la siguiente información.

- Una breve descripción de la naturaleza del problema.
- Los mensajes de error detallados que aparecen cuando se produce el problema.
- Los pasos detallados para reproducir el problema.
- Los archivos de registro, archivos de sesión, archivos de configuración y archivos de datos relacionados.
- Información sobre el producto y el entorno del sistema, que puede obtener tal como se describe en "Información del sistema".

Información del sistema

Cuando llame al soporte técnico de IBM Unica , es posible que se le pida que proporcione información sobre el entorno.

Si el problema no le impide iniciar una sesión, mucha de esta información está disponible en la página Acerca de, que proporciona información sobre las aplicaciones IBM Unica instaladas.

Puede acceder a la página Acerca de seleccionando **Ayuda > Acerca de**. Si no se puede acceder a la página Acerca de, puede obtener el número de versión de cualquier aplicación IBM Unica visualizando el archivo `version.txt` situado en el directorio de instalación para cada aplicación.

Información de contacto para el soporte técnico de IBM Unica

Para ver las formas de contacto con el soporte técnico de IBM Unica , consulte el sitio web de soporte técnico del producto IBM Unica : (http://www.unica.com/about/product-technical-support.htm).

Avisos

Esta información se ha desarrollado para productos y servicios ofrecidos en los EE.UU.

Es posible que IBM no ofrezca los productos, servicios o características que se tratan en este documento en otros países. Consulte con su representante local de IBM para obtener más información sobre los productos y servicios disponibles actualmente en su zona. Cualquier referencia a un producto, programa o servicio de IBM no pretende afirmar ni implicar que sólo se pueda utilizar dicho producto, programa o servicio de IBM. En su lugar se puede utilizar cualquier producto, programa o servicio funcionalmente equivalente que no infrinja ningún derecho de propiedad intelectual de IBM. Sin embargo, es responsabilidad del cliente evaluar y verificar el funcionamiento de cualquier producto, programa o servicio que no sea de IBM.

IBM puede tener patentes o solicitudes de patente pendientes que cubran el tema tratado descrito en este documento. La posesión de este documento no le otorga ninguna licencia sobre estas patentes. Puede enviar consultas sobre licencias, por escrito, a:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
EE.UU.

Para consultas sobre licencias relacionadas con información de doble byte (DBCS), póngase en contacto con el Departamento de propiedad intelectual de IBM de su país o envíe consultas, por escrito, a:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japón

El siguiente párrafo no se aplica al Reino Unido ni a ningún otro país donde dichas disposiciones entren en contradicción con la legislación local: INTERNATIONAL BUSINESS MACHINES CORPORATION PROPORCIONA ESTA PUBLICACIÓN "TAL CUAL" SIN NINGÚN TIPO DE GARANTÍA, NI IMPLÍCITA NI EXPLÍCITA, INCLUYENDO, PERO SIN LIMITARSE A, LAS GARANTÍAS IMPLÍCITAS DE NO VULNERABILIDAD, COMERCIALIZACIÓN O ADECUACIÓN PARA UN PROPÓSITO DETERMINADO. Algunos estados no permiten la renuncia de garantías explícitas o implícitas en determinadas transacciones, por lo tanto, es posible que esta declaración no se aplique en su caso.

Esta información podría incluir inexactitudes técnicas o errores tipográficos. Periódicamente se realizan cambios en la información aquí contenida; estos cambios se incorporarán en nuevas ediciones de la publicación. IBM puede realizar mejoras y/o cambios en los productos y/o los programas descritos en esta publicación en cualquier momento sin previo aviso.

Las referencias en esta información a sitios web que no son de IBM se proporcionan únicamente para su comodidad y no se deben considerar en modo alguno como una recomendación de dichos sitios web. Los materiales de dichos sitios web no forman parte de los materiales para este producto IBM y el uso de estos sitios web corre a cuenta y riesgo del cliente.

IBM puede utilizar o distribuir cualquier información que se le facilite de la manera que considere adecuada, sin contraer por ello ninguna obligación con el remitente.

Los titulares de licencias de este programa que deseen obtener información sobre el mismo con el fin de permitir: (i) el intercambio de información entre programas creados independientemente y otros programas (incluido este) y (ii) el uso mutuo de la información que se ha intercambiado, deberán ponerse en contacto con:

IBM Corporation
170 Tracer Lane
Waltham, MA 02451
EE.UU.

Dicha información puede estar disponible, sujeta a los términos y las condiciones apropiados, que incluye, en algunos casos, el pago de una tarifa.

El programa bajo licencia descrito en este documento y todo el material bajo licencia disponible para él mismo los proporciona IBM de acuerdo con los términos del Acuerdo de Cliente de IBM, el Acuerdo Internacional de Programa bajo Licencia de IBM o cualquier acuerdo equivalente entre las partes.

Los datos de rendimiento incluidos aquí se han determinado en un entorno controlado. Por lo tanto, los resultados obtenidos en otros entornos operativos podrían variar de forma significativa. Algunas mediciones se pueden haber realizado en sistemas de nivel de desarrollo y no existen garantías de que estas mediciones sean las mismas en sistemas de disponibilidad general. Además, es posible que algunas mediciones se hayan calculado mediante extrapolación. Los resultados reales pueden variar. Los usuarios de este documento deben verificar los datos aplicables para su entorno específico.

La información relativa a productos que no son de IBM se ha obtenido de los proveedores de estos productos, sus anuncios publicados u otras fuentes disponibles de forma pública. IBM no ha probado estos productos y no puede confirmar la precisión del rendimiento, la compatibilidad o cualquier otra reclamación relacionada con productos que no son IBM. Las preguntas sobre las capacidades de productos no IBM se deberán dirigir a los proveedores de estos productos.

Todas las sentencias relacionadas con la futura dirección o intención de IBM están sujetas al cambio o la retirada sin previo aviso y sólo representan los objetivos y las metas.

Todos los precios de IBM mostrados son precios de venta sugeridos de IBM, son actuales y están sujetos a cambio sin previo aviso. Los precios de distribuidor pueden variar.

Esta información contiene ejemplos de datos e informes utilizados en el funcionamiento diario de la empresa. Para ilustrarlos de la forma más completa posible, los ejemplos incluyen los nombres de personas, compañías, marcas y

productos. Todos estos nombres son ficticios y cualquier similitud con los nombres y las direcciones utilizados por una empresa real es mera coincidencia.

LICENCIA DE COPYRIGHT:

Esta información contiene programas de aplicación de muestra en lenguaje fuente, que ilustran las técnicas de programación en distintas plataformas operativas. Puede copiar, modificar y distribuir estos programas de ejemplo de cualquier modo sin pagar a IBM con el fin de desarrollar, utilizar, comercializar o distribuir programas de aplicaciones que se ajusten a la interfaz de programación de aplicaciones para la plataforma operativa para la que se ha escrito el código de ejemplo. Estos ejemplos no se han verificado de forma exhaustiva bajo todas las condiciones. Por lo tanto, IBM no puede garantizar ni dar por sentado la fiabilidad, capacidad de servicio o funcionamiento de estos programas. Los programas de muestra se proporciona "TAL CUAL", sin ningún tipo de garantía. IBM no será responsable de ningún daño resultante del uso de los programas de muestra.

Si está visualizando esta información en copia software, es posible que las fotografías y las ilustraciones en color no aparezcan.

Marcas registradas

IBM, el logotipo de IBM e ibm.com son marcas registradas o marcas comerciales registradas de International Business Machines Corp., registradas en muchas jurisdicciones en todo el mundo. Otros nombres de producto y servicio pueden ser marcas registradas de IBM u otras compañías. Hay disponible una lista actual de marcas registradas de IBM en la web en “Copyright and trademark information” (Información de copyright y marca registrada) en www.ibm.com/legal/copytrade.shtml.

Impreso en España