VERSIÓN 18.0.0
ABRIL DE 2023
702P09005

# SDK de distribución del software Xerox® FreeFlow® VI eCompose

Guía del usuario

xerox™

Producido en los Estados Unidos de América.

BR38523

# Tabla de contenido

# Introducción

Este capítulo incluye:



Esta guía se ha diseñado para desarrolladores de software que usan el Software Development Kit (SDK) de FreeFlow® VI eCompose Dispatch (VIeCD) para integrar aplicaciones de postprocesamiento con el software VI eCompose (VIeC). Para usar el software VI eCompose, se recomienda que se familiarice con el software o plataformas siguientes:

- Lenguaje Xerox® VIPP®
- Software VIeC Dispatch
- Lenguaje de programación C, C++, o aplicaciones de la plataforma

Se recomienda que los usuarios tengan experiencia con el software VIeC Dispatch. Consulte el ejemplo en la *Guía del usuario de FreeFlow® VI eCompose* y en el *Taller de VI eCompose*.

Este documento incluye información general de VIeC, como una descripción de los estados y los flujos de datos de VIeCD. La información complementa el material de referencia y la aplicación de ejemplo de VIeCD en la *Guía del usuario de FreeFlow® VI eCompose*.

Nota: Todos los nombres de productos de los módulos del software FreeFlow® VI Suite cambiaron a partir de la versión 10.0 de FreeFlow VI Suite.

| NOMBRE DE PRODUCTO DE LEGADO | NOMBRE DE PRODUCTO NUEVO |
|---|---|
| FreeFlow VI Interpreter | FreeFlow VI Compose |
| FreeFlow VI Interpreter Open Edition | FreeFlow VI Compose Open Edition |
| FreeFlow VI Designer | FreeFlow VI Design Pro |
| FreeFlow VI PDF Originator | FreeFlow VI eCompose |
| FreeFlow VIPP® Pro Publisher | FreeFlow VI Design Express |

Todos los demás productos que no se mencionan en la lista conservan el mismo nombre de la versión anterior de FreeFlow VI Suite.

Las referencias al lenguaje, comandos y formato de información variable de VIPP® permanecen sin cambio.

## Foro de clientes de VI Suite

Xerox proporciona un foro de asistencia técnica a la comunidad. El foro de clientes de VI Suite ahora es parte de un foro de asistencia más grande, que le permite publicar y revisar información acerca de los productos y servicios de Xerox desde una sola ubicación. Tómese un minuto para iniciar sesión en la comunidad del foro de clientes: http://vippsupport.xerox.com.

## ¿Qué es el kit de desarrollo de software de VIeCD?

VIeCD SDK consiste en una recopilación de ejemplos y puntos de inicio como código fuente, utilidades y bibliotecas, que pueden usarse para integrar a VIeCD con otros flujos de trabajos. La mayoría del código proporcionado se ha escrito en C (con la excepción del ejemplo olsession, que se ha escrito en C++). Una API (interfaz del programador de la aplicación), la biblioteca de encapsulación de la regla de distribución (vtpdwrap), se presenta en la definición de la clase en el archivo vtpdwrap.h.

También se proporciona la documentación de referencia en formato HTML y Adobe PDF. Los documentos en formato HTML fueron extraídos del código de origen VIeCD SDK y, a continuación, fueron cruzados e indexados, y se pueden encontrar en el directorio /vipodsdk/docs del soporte de distribución de VIeCD SDK. Además del archivo PDF que está leyendo, los documentos readme (léame) PDF se pueden encontrar en los subdirectorios /vipodsdk/apps. Estos proporcionan instrucciones para el uso de las aplicaciones de ejemplo.

## Inicio

Para usar VIeCD SDK, vaya a /vipodsdk/docs/index.html. Este archivo proporciona la información básica necesaria como:

- Una breve introducción a SDK
- Información de licencia
- Dónde obtener información de asistencia y de plataformas compatibles con VIeCD SDK
- Creación del SDK
- Suposiciones acerca de las ubicaciones de los archivos y el entorno
- Antecedentes sobre las decisiones de diseño de VIeCD SDK
- Implementación de VIeCD SDK
- Descripciones de los documentos y los archivos que deben ser revisados para comenzar
- Una breve descripción del contenido de VIeCD SDK, como:
  - Utilidades
  - Bibliotecas
  - Código de ejemplo
  - Aplicaciones de ejemplo de VIPP®
  - Diseño del archivo de Win32

Además, la página principal, `/vipodsdk/docs/index.html`, contiene los enlaces siguientes:

| | |
|---|---|
| Estructuras de datos | Una página que contiene una lista de las estructuras de datos proporcionadas y una breve descripción de las mismas. Para ver las descripciones completas de cada estructura, haga clic en el hipertexto de esta página. |
| Lista de archivos | Una página que contiene una lista de todos los archivos documentados con una breve descripción de los mismos. Para ver las descripciones completas de cada archivo, haga clic en el hipertexto de esta página. |
| Campos de datos | Una página indexada con una lista de todas las estructuras documentadas y los campos de unión y enlaces a las estructuras y a las uniones a las que pertenecen. |
| Elementos globales | Una lista de todas las definiciones de tipos, enumeraciones, definiciones, variables y funciones documentadas con enlaces a la documentación relacionada. |
| Ejemplos | Una lista de los ejemplos proporcionados con el SDK y enlaces al código fuente para estos. |

Después de revisar los archivos, use el documento para obtener información y explicaciones sobre los archivos y las utilidades que componen el VIeCD SDK.

## Descripción general de la documentación

Esta guía proporciona información sobre VIeCD SDK y VIeCD In-Circuit Emulator (vtpdice). La guía se organiza como se indica a continuación:

| | |
|---|---|
| Antecedentes | Proporciona información general sobre VIeCD y VIeCD SDK. Este capítulo complementa la información de la *Guía del usuario de FreeFlow VI eCompose* e incorpora información específica sobre VIeCD SDK. Este capítulo proporciona una descripción general de VIeCD y los temas siguientes:<br>Flujo de datos de VIeCD<br>Filtros de IncomingFolders de VIeCD<br>Elegibilidad: El nombre de campo DispatchRule, CommandTemplates y RuleVars<br>Ciclo de vida de los trabajos VIeCD |
| Ejemplos, bibliotecas y utilidades | Proporciona descripciones de los archivos y las utilidades proporcionados con el VIeCD SDK, y ejemplos de cómo utilizarlos. |
| VIeC Dispatch In-Circuit Emulator | Proporciona una descripción ampliada de la utilidad vtpdice, e incluye las siguientes secciones:<br>Uso de vtpdice<br>Uso de vtpdice en modo por lotes |

Para obtener más información acerca del lenguaje VIPP®, VI Compose y los módulos relacionados, consulte la documentación de FreeFlow® Variable Information Suite. La documentación incluye las guías siguientes:

- *Guía del usuario de FreeFlow® VI Compose*: Proporciona la información necesaria para comprender y usar VIPP® y sus aplicaciones. La guía describe los archivos y las utilidades proporcionadas con el software, los recursos necesarios para generar trabajos VIPP® y la información básica para la impresión mediante VIPP®.
- *Manual de referencia de lenguaje VIPP®*: Documenta los comandos de VIPP®, proporciona sugerencias de programación y mensajes de error de VIPP®.
- *Guía del usuario de FreeFlow® VI eCompose*: Contiene información sobre el uso del software VI eCompose para crear y enviar documentos de Adobe PDF, y para administrar de forma remota servidores web de VIeC.
- *Guía del usuario de FreeFlow® VI Design Pro*
- *Guía del usuario de VIPP® Manage*
- *Guía del usuario de FreeFlow® VI Explorer*
- *Glosario y referencia rápida de la documentación de FreeFlow® Variable Information Suite*

Para obtener más información acerca de la formación de VIPP®, póngase en contacto con un representante de Xerox.

# Antecedentes

Este capítulo incluye:



El software VIeC Dispatch proporciona un mecanismo de envío genérico que inicia y supervisa el postprocesamiento de trabajos VIeC por parte de procesos de servidor especificados por el cliente, como procesos de correo electrónico, fax o depósito de documentos. En este rol, el software VIeC Dispatch se puede considerar middleware, ya que media entre los trabajos VIeC completados y el software de postprocesamiento de servidor especificado.

En los parámetros incrustados y en el resto de datos del postprocesamiento incorporados al trabajo VIeC, utilice el comando BOOKMARK de VIPP®.

- Los parámetros y el resto de datos se extraen de los nombres de campo y los valores de archivos de índice generados por VIeC.
- El software VIeC Dispatch transfiere los parámetros y el resto de datos al software de servidor especificado.
- Los archivos de índice de dada trabajo se identifican mediante la extensión **.csv**.

VIeCD es compatible con los flujos de trabajo que requieren la intervención del usuario, o con los flujos de trabajo automáticos, mediante el uso de la función AutoRun y de los filtros de usuario. Por ejemplo:

- Servidores que interactúan con sistemas de adquisición de correo electrónico y que requieren:
  - Comprobación o verificación personal de la salida de VIeC antes del envío
  - Limitaciones de los usuarios que pueden iniciar el envío de dichos trabajos
- Servidores que interactúan con depósitos de documentos que requieren flujos de trabajo automáticos. El proceso VIeC a VIeCD se ejecuta sin intervención del usuario.

Cuando VIeCD invoca a un programa de servidor, los parámetros se extraen de un archivo de índice del trabajo VIeC sobre una base de línea por línea. El proceso crea una nueva instancia del programa de servidor como un nuevo subproceso de cada invocación. VIeCD no permite ninguna interacción directa con el programa de servidor a través del subproceso en stdin/stdout durante la invocación. Es posible que la limitante de la invocación no sea adecuada para interactuar con todos los tipos de programas de servidor. Las posibles incompatibilidades entre el postprocesamiento de VIeC y VIeCD incluyen:

- Programas que requieren algún tipo de interacción del usuario o interacción programática en el modo normal de funcionamiento, por ejemplo, la respuesta Sí o No para la sobreescritura de archivos.
- Programas que requieren algún tipo de estado de sesión en un conjunto de transacciones, como el registro en el servidor de Microsoft Exchange para realizar transmisiones de correo electrónico.
- Soluciones de servidores que implican más de una operación discreta y que requieren la invocación de diversas operaciones discretas de postprocesamiento. Los ejemplos de estas operaciones incluyen la

concatenación u otras combinaciones de archivos especificados en la Plantilla del archivo de datos antes del envío de uno o más programas de servidor.

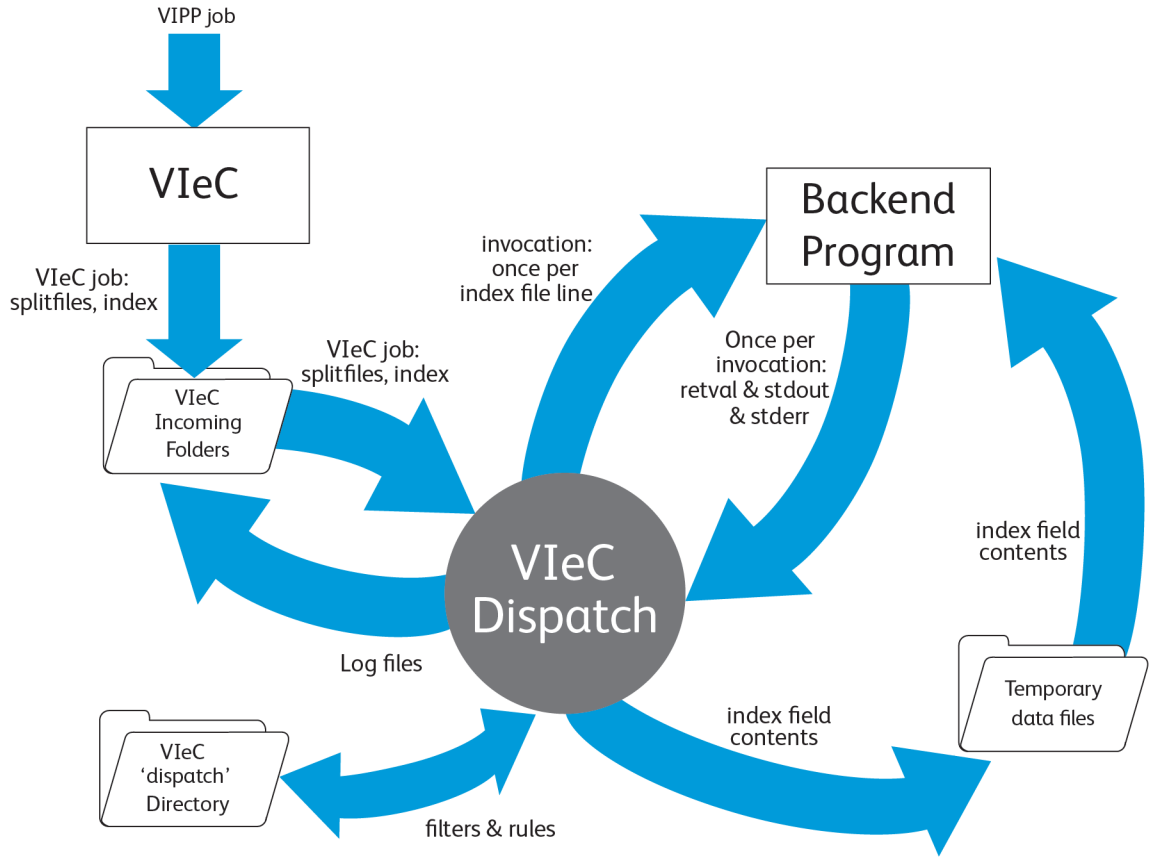
El software VIeCD SDK contiene códigos de ejemplos y bibliotecas que proporcionan la base para generar complementos, proxys, envoltorios o interfaces para resolver los problemas de integración. El enfoque principal de los ejemplos de VIeCD SDK se centra en la aplicación independiente del servidor. La aplicación del servidor actúa como un enlace de sesión entre un simple complemento de cliente que es invocado por VIeCD en un archivo de índice línea por línea y el programa de servidor. El programa de servidor podría requerir la interacción o la intervención del usuario, el estado de la sesión o diversas operaciones de postprocesamiento.

Como alternativa al enfoque cliente/servidor, para programas de servidor que no requieren el estado de la sesión en un conjunto de transacciones, se puede interponer un programa proxy entre VIeCD y los programas de servidor. Los programas de servidor son invocados una vez por cada línea en el archivo de índice. El proxy asume la responsabilidad de interactuar con el usuario en caso necesario, y/o actúa como interfaz de un conjunto de programas de servidor discretos si se necesitan diversas etapas de postprocesamiento. El proxy se comunica con VIeCD a través de sus interfaces documentadas (retval, stdin/stdout), así como con los programas de servidor que realizan el postprocesamiento. No se proporcionan ejemplos explícitos con el VIeCD SDK; sin embargo, puede configurar la utilidad vtpdice para que actúe como proxy entre VIeCD y un programa de servidor compatible con VIeCD. Se recomienda que revise el origen de vtpdice para ver la posible información de implementación del programa proxy.

Para obtener más información, consulte estas secciones de la *Guía del usuario de FreeFlow VI eCompose Dispatch SDK*:

- Ejemplos
- Bibliotecas
- Utilidades
- Uso de vtpdice
- Uso de vtpdice en modo por lotes

## Flujo de datos de VIeCD




A medida que el software VIeCD interactúa con los programas de servidor, VIeCD inspecciona periódicamente los directorios IncomingFolders de VIeC que coinciden con el filtro de usuario especificado. El software VIeCD compara los nombres de campo y los contenidos de la primera línea de cada archivo de índice del trabajo VIeC con un depósito de reglas de envío.

- Si un trabajo VIeC que ha finalizado se puede asociar con una sola regla de distribución, el trabajo se considera elegible para el envío y se convierte en un trabajo VIeCD.
- Si el trabajo VIeCD se aprueba manualmente para su procesamiento, o si satisface los criterios especificados por el usuario para un determinado filtro AutoRun, VIeCD resuelve de forma secuencial y aplica la regla de distribución correspondiente a cada línea del archivo de índice del trabajo VIeC.

En el contexto de una regla de distribución, cada línea del archivo de índice del trabajo VIeC resulta en una llamada independiente del programa de servidor especificado por el cliente, lo que hace que efectivamente se convierta en una subtarea del trabajo VIeCD. La ejecución secuencial de cada subtarea conlleva opcionalmente la escritura en disco del contenido de uno o más campos de las líneas del archivo de índice y, a continuación, la realización de una llamada al programa de servidor. A medida que se finaliza cada subtarea, el flujo de retval, stdout y stderr se inspecciona en el contexto de la regla de distribución a fin de determinar si ha ocurrido un aviso o error. Si ocurrió un aviso o un error, el software determina si el procesamiento de subtareas continúa o se detiene. Los resultados del procesamiento de subtareas se acumulan en un archivo de registro en el mismo directorio en el que los resultados del procesamiento de trabajos de VIeC se almacenaron, habitualmente, un

subdirectorio del directorio de entrada del usuario.

## Filtros IncomingFolders de VIeCD

VIeCD supervisa el directorio IncomingFolders de VIeC en el contexto de un filtro IncomingFolders especificado. El valor predeterminado permite a VIeCD supervisar todas las IncomingFolders de todos los usuarios de VIeC. Sin embargo, VIeCD se puede configurar para procesar trabajos de un determinado conjunto de usuarios, y/o para un determinado conjunto de IncomingFolders, ya que solo las IncomingFolders especificadas de los usuarios de VIeC que coinciden con el filtro de usuario indicado pueden ser procesados a través de VIeCD.

## Elegibilidad: CommandTemplates, RuleVars y DispatchRule FieldName

VIeC Dispatch determina la elegibilidad para el procesamiento de trabajos VIeC terminados que coinciden con el filtro IncomingFolders. Si VIeC job coincide con una sola regla de distribución, VIeC Dispatch considera el trabajo elegible para ser procesado. CommandTemplate es la sección de una regla de distribución que determina el programa de back-end invocado y sus parámetros.

### ARCHIVO DE ÍNDICE

Para determinar la elegibilidad, el software VIeC Dispatch inspecciona los nombres de campo del archivo de índice de cada trabajo VIeC y compara los campos con las secciones de CommandTemplate de las reglas de distribución disponibles.

En el ejemplo 1, si ninguna CommandTemplate de las reglas de distribución contiene el campo mailto, el trabajo VIeC es inelegible porque no se puede aplicar ninguna regla de distribución. Si los nombres de campo del archivo de índice de un trabajo VIeC se ajustan a la CommandTemplate de una sola regla de distribución, el archivo se considera elegible.

Ejemplo 1

En este ejemplo, los siguientes nombres de campo del archivo de índice del trabajo VIeC y la CommandTemplate de la regla de distribución única son:

```
...,"Pages","FileSequence","mailto"
```

```
blat c:\bodytemp.txt -t $mailto
```

El trabajo VIeC es elegible para ser procesado con VIeCD si la regla de distribución en el ejemplo es la única con una CommandTemplate que contiene un solo campo mailto.

Alternativamente, el trabajo VIeC es inelegible para ser procesado si existe otra regla que contenga la siguiente información variable de CommandTemplate, ya que se podría aplicar más de una regla de distribución:

```
splat -x $mailto
```

Ejemplo 2

Este ejemplo supone:

- Dos trabajos VIeC, cada uno con uno de estos nombres de campo de archivo de índice:

  ```
  ...,"Pages","FileSequence","mailto"
  ```

  ```
  ...,"Pages","FileSequence","mailto","cc"
  ```

- Dos reglas de distribución, cada una con una de las siguientes como la CommandTemplate:

  ```
  blat c:\bodytemp.txt -t $mailto
  ```

  ```
  blat c:\bodytemp.txt -t $mailto -c $cc
  ```

En este ejemplo, ambos trabajos VIeC son elegibles para procesamiento. VIeCD reconoce que el primer trabajo VIeC no puede usarse con la segunda regla de distribución debido a que no existe un campo con el nombre cc en el primer archivo de índice del trabajo. La primera regla de distribución se aplica al primer trabajo VIeC. VIeCD reconoce que el segundo trabajo VIeC no puede usarse con la primera regla de distribución debido a

que no existe un campo con el nombre cc en la CommandTemplate de la primera regla de distribución. La segunda regla de distribución se aplica al segundo trabajo VIeC.

### RULEVARS

Además de los nombres de campo del archivo de índice definidos por un trabajo VIeC, una regla de distribución puede también definir los campos adicionales y los valores se aplican a la CommandTemplate. Un requisito común de esto es una regla de distribución cuya CommandTemplate requiere una clave para ejecutarse. No es deseable incluir la contraseña en el trabajo VIeC. Estos campos adicionales y sus valores se pueden especificar en la sección RuleVar de la regla de distribución.

Ejemplo 3

En este ejemplo, la CommandTemplate de la regla de distribución y la regla de distribución única con el campo del usuario son:

```
foo -user $user -password $password
```

```
...,"Pages","FileSequence","user"
```

En este caso, el trabajo VIeC es inelegible para ser procesado, ya que no hay campos de contraseña en los nombres de campo del archivo de índice.

Ejemplo 4

El trabajo VIeC en el ejemplo 3 será elegible para procesarse cuando la regla de distribución tenga una entrada RuleVar para el campo de contraseña:

```
password=mypassword
```

En este caso, el campo de contraseña de la CommandTemplate es proporcionado por la definición de RuleVar de la regla de distribución.

### NOMBRE DEL CAMPO DEL ARCHIVO DE ÍNDICE RESERVADO

El sistema final que se utiliza en VIeC Dispatch para seleccionar una regla de distribución para un determinado trabajo VIeC implica el uso de un nombre de campo de archivo de índice reservado. Si el trabajo VIeC incluye entre los nombres de campo de su archivo de índice el nombre de campo reservado DispatchRule, el contenido de ese campo de la primera línea del archivo de índice se compara con los nombres del conjunto de reglas de distribución ambiguas.

Ejemplo 5

La ambigüedad se resuelve y la regla de distribución se aplica al trabajo VIeC cuando se satisfacen las condiciones en un trabajo con los nombres de campo de archivo de índice de mailto y DispatchRule:

- El trabajo se compara con las dos reglas con blat y splat descritas en el Ejemplo 1 debido al nombre de campo mailto.
- El contenido del nombre de campo reservado, DispatchRule, para la primera línea del registro de índice es SMTP email.
- Una de las reglas de distribución tiene Correo electrónico SMTP como el nombre de la regla de distribución.

## Filtros AutoRun

Cuando un trabajo VIeC se considera elegible para ser procesado por VIeCD, se compara al filtro AutoRun especificado actualmente. Los filtros AutoRun permiten especificar un determinado conjunto de usuarios de VIeC y un grupo de IncomingFolders para dichos usuarios, a fin de determinar los trabajos VIeC procesados automáticamente por VIeCD. La configuración predefinida es la desactivación de AutoRun para todos los usuarios, de manera que deban seleccionar los trabajos de forma manual para realizar su procesamiento.

## Procesamiento

Cuando VIeC Dispatch procesa un trabajo VIeC, lee el archivo de índice del trabajo VIeC línea a línea, y aplica sus valores de campo en el contexto de la regla de distribución aplicable. Cada aplicación realiza una llamada independiente del programa de servidor especificado en la CommandTemplate de la regla de distribución. Cada una de estas llamadas (una por cada línea del archivo de índice del trabajo VIeC) se considera una subtarea del trabajo.

Se realizarán las siguientes acciones para cada línea del archivo de índice hasta que se detecte una condición de detención por aviso o error, o cuando no queden líneas en el archivo de índice que se procesa:

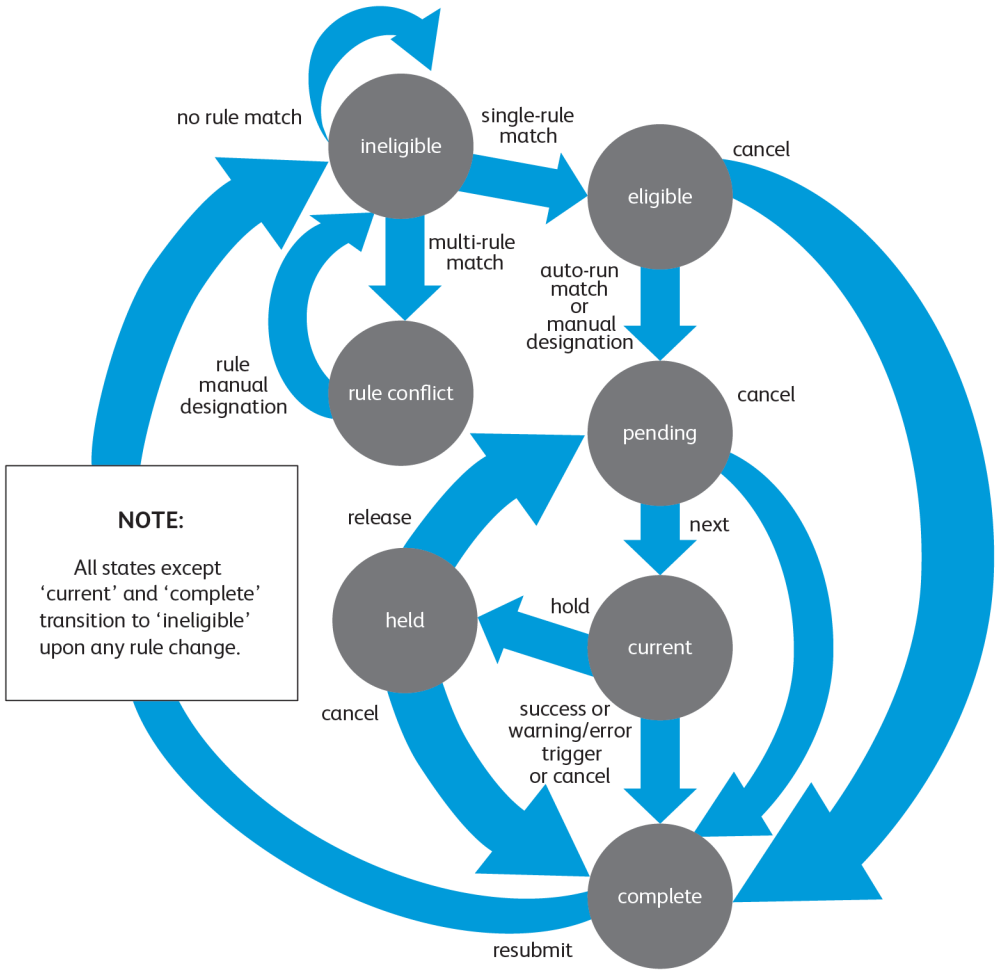
- Si alguno de los nombres de campo figuran en la sección DatafileTemplate de la regla de distribución, sus contenidos se escriben en el disco según lo especificado. Los archivos escritos se consideran temporales, y sus contenidos solo serán válidos durante la duración de la llamada del programa de servidor especificado en la CommandTemplate.
- A continuación, los nombres de campo en la CommandTemplate se resuelven a través de la sustitución de los valores de RuleVars (si los hay) y de los valores extraídos de la línea correspondiente del archivo de índice. El resultado es el comando y los parámetros para la llamada del programa de servidor. A continuación, se envían los comandos y los parámetros al sistema operativo para la ejecución como un subproceso de VIeC Dispatch.
- Cuando el subproceso del programa de servidor finaliza, VIeCD interpreta sus flujos retval, stdout y stderr en el contexto de las secciones pertinentes de la regla de distribución para determinar si se ha generado una situación de aviso o error y, en caso afirmativo, si debe continuar el procesamiento de las líneas restantes del archivo de índice.

## Ciclo de vida de los trabajos VIeCD

Durante su ciclo de vida, los trabajos VIeCD pueden estar en cualquiera de los estados siguientes:

- inelegible
- conflicto de reglas
- elegible
- pendiente
- actual
- retenido
- completado

Estos estados se muestran en Trabajo actual y en las pestañas de la GUI principal de VIeC Dispatch. Estas opciones se describen con más detalle:



### INELEGIBLE

Los trabajos VIeC que superan el filtro IncomingFolders dan lugar a la creación de una instancia de trabajo VIeCD con el estado inicial Inelegible, lo que, a su vez, crea el estado inicial del ciclo de vida del trabajo VIeCD. La elegibilidad de los trabajos VIeCD que entran en este estado se comprueba de forma inmediata al evaluar los nombres de campo del primer registro del archivo de índice según la CommandTemplate y la RuleVar de cada regla de distribución disponible.

**Una única regla de distribución**

El trabajo VIeCD pasa al estado Elegible.

**Más de una regla de distribución**

El trabajo pasa al estado Conflicto de reglas.

**No hay reglas de distribución**

El trabajo permanece en el estado Inelegible.

### CONFLICTO DE REGLAS

Los trabajos en este estado son aquellos que tienen más de una regla de distribución aplicable. Se abandonará este estado si:

- La regla que se debe aplicar se selecciona de forma manual.
- Se cambia o agrega una regla de distribución.

En cualquier caso, el trabajo volverá al estado Inelegible para, a continuación, volver ser evaluado. Al volver al estado Inelegible:

- El trabajo pasará al estado Elegible si se selecciona manualmente una regla de distribución.
- Es posible que el trabajo se mantenga como no elegible si no se pueden aplicar reglas de distribución.
- El trabajo puede volver al estado Conflicto de reglas si todavía hay diversas reglas de distribución aplicables.

### ELEGIBLE

Los trabajos en este estado pueden ser procesados pero todavía no están pendientes. Un determinado trabajo seguirá en el estado elegible hasta que sea seleccionado de forma manual por el usuario para su procesamiento, o hasta que satisfaga los criterios del filtro AutoRun actual. Cuando el trabajo satisface los criterios del filtro AutoRun, pasa automáticamente al estado pendiente.

La cancelación de un trabajo en el estado elegible hará que pase al estado completado (cancelado).

### PENDIENTE

Los trabajos en este estado se procesarán en el orden mostrado en la pantalla Pendiente de Distribución de VIeC, el elemento superior se procesará en primer lugar. Se puede cambiar el orden de los trabajos pendientes mediante la interfaz gráfica de usuario.

La cancelación de un trabajo en el estado pendiente hará que pase al estado completado (cancelado).

### ACTUAL

Se está procesando un trabajo en este estado. Solo puede haber un trabajo en este estado.

La retención de un trabajo actual hace que se suspenda temporalmente su procesamiento (de una manera ordenada) y que se pase al estado retenido.

La cancelación de un trabajo en el estado actual hace que este pase al estado completado con un estado cancelado.

### RETENIDO

Se ha suspendido el procesamiento de un trabajo retenido. La liberación del trabajo hace que pase al estado pendiente a la espera de seguir el procesamiento.

La cancelación de un trabajo en el estado retenido hará que pase al estado completado (cancelado).

### COMPLETADO

Este es el estado final del ciclo de vida del trabajo VIeCD. Los trabajos completados tienen uno de estos estados:

- correcto
- aviso
- error
- cancelar

La función de reenvío de la transición se suministra para su comodidad. El reenvío de un trabajo elimina todos los estados acumulados, los seguimientos, etc. del trabajo, y lo pasa al estado Inelegible para que pueda volver a ser procesado como si hubiera llegado al sistema por la primera vez.

# Ejemplos, bibliotecas y utilidades

Este capítulo incluye:



VIeCD SDK proporciona ejemplos, bibliotecas y utilidades que se pueden usar para integrar aplicaciones de servidor con VI eCompose. Se proporcionan descripciones de estos componentes.

## Ejemplos

Estos ejemplos de código se proporcionan con VIeCD SDK:

**forward**

Reenvío de archivos simple para VIeCD.

**client**

Un cliente de socket sencillo que ilustra la conexión a una aplicación de sesión VIeCD.

**server**

Un servidor de socket sencillo que puede utilizar como base para una aplicación de sesión del servidor VIeCD.

**server2**

Un servidor de socket más sofisticado que puede utilizar como base para una aplicación de sesión del servidor VIeCD. El ejemplo muestra cómo usar la biblioteca `vtpdsession`.

**olSend**

Un cliente de socket que se conecta a un enlace de sesión de Microsoft Outlook.

**olsession**

Un servidor de socket que actúa como un enlace de sesión del servidor back-end de VIeCD a MS Outlook.

**wrap**

Muestra cómo usar la biblioteca del contenedor de la regla de distribución `vtpdwrap`.

Puede encontrar el origen de estos ejemplos en la sección `src/examples` de la distribución de *VIeCD SDK*. Los archivos binarios previamente compilados de los ejemplos se encuentran en la sección de contenedor. Las aplicaciones de VIPP® de muestra que puede utilizar para ejecutar los ejemplos se encuentran en la sección de aplicaciones.

Los ejemplos se describen más detalladamente en las secciones siguientes.

### FORWARD

La manera más directa de resolver una incompatibilidad entre VIeCD y un programa back-end consiste en reemplazar el programa back-end con un programa con mayor compatibilidad con VIeCD. Por ejemplo, un simple uso de VIeCD permite automatizar la copia de archivos PDF de trabajos VIeC completados a otra ubicación, conocido como reenvío de trabajos. Una manera para automatizar el envío de trabajos es a través de una llamada a la utilidad xcopy de la línea de comandos de Windows en la CommandTemplate de una regla de distribución. Sin embargo, al copiar un archivo a una nueva ubicación, xcopy solicitará la confirmación sobre si la copia del mismo sobrescribirá a otro archivo. Las solicitudes de confirmación no son compatibles con VIeCD, puesto que no se puede realizar ninguna interacción con el subproceso en stdin o stdout y harán que VIeCD se pare o que se agote el tiempo de espera. Además, el valor de retorno de xcopy no es informativo si se producen otros problemas durante el proceso.

El ejemplo de reenvío muestra una manera de encapsular funcionalidades para aumentar la compatibilidad

con el entorno de llamada de VIeCD. El programa no es interactivo. En su lugar dispone de argumentos de línea de comandos para especificar si se desea sobrescribir un archivo si existe. Los valores devueltos indican lo siguiente:

| | |
|---|---|
| 0 | success (éxito) |
| valores negativos | error de uso o entorno |
| valores positivos de retorno | error del sistema, por ejemplo, un error de disco lleno o de permisos |

Es práctico sustituir solo los procesos back-end más sencillo de esta manera.

Para obtener instrucciones sobre cómo usar el ejemplo de reenvío, consulte `/vipodsdk/apps/forward/readme.pdf`.

### CLIENT

El ejemplo client proporciona un proxy simple entre VIeCD y un determinado servidor, en el que el cliente se comunica con el servidor a través de una conexión de sockets. Se asume que el servidor se está ejecutando antes de iniciar la comunicación con el trabajo VIeCD a través del cliente. El ejemplo client se comunicará con el servidor o los servidores de ejemplo del servidor 2.

Si se realiza una llamada al cliente en la `CommandTemplate` de la regla de distribución, cada vez que VIeCD realiza dicha llamada o a través de un proceso independiente:

- El cliente establece una conexión con el servidor y le pasa los parámetros con los que se realizó la llamada al servidor.
- A continuación, el cliente espera a que el servidor proporcione el valor de retorno que representa el estado de procesamiento del servidor en esa transacción en particular.
- El cliente pasa el valor devuelto desde el servidor a VIeCD como valor de retorno del cliente.

Consulte `/vipodsdk/apps/client/readme.pdf` para obtener instrucciones sobre cómo usar el ejemplo cliente.

### SERVER

Este ejemplo proporciona un servidor básico que se comunica con el ejemplo client (cliente) descrito más arriba. No hace nada al recibir las solicitudes de transacción de los procesos cliente, y siempre devuelve el mismo valor de retorno encapsulado para representar el estado de la transacción.

La implementación es minimalista de forma intencional, ya que el servidor solo envía valores de retorno a las conexiones cliente entrantes. No se puede pedir el cierre del proceso de servidor y debe cerrarse manualmente. Además, al cerrarse, el servidor no permite la finalización de las conexiones del cliente pendientes antes de salir. El servidor se apaga y hace que las conexiones del cliente se valgan por sí mismas.

El ejemplo server puede ampliarse para realizar tareas útiles más potentes, pero su propósito principal es compararse al ejemplo `server 2` más abajo.

Consulte `/vipodsdk/apps/server/readme.pdf` para obtener instrucciones sobre cómo usar el

ejemplo server.

### SERVER2

Este ejemplo es equivalente al ejemplo server descrito más arriba. Sin embargo, muestra cómo aprovechar la biblioteca `vtpdsession` para producir la misma funcionalidad con menos código. El código es también más potente, ya que la implementación de la biblioteca `vtpdsession` facilita el cierre de los clientes conectados como parte de su proceso de finalización.

Consulte `/vipodsdk/apps/server2/readme.pdf` para obtener instrucciones sobre cómo usar el ejemplo server2.

### OLSEND

Como los ejemplos anteriores, `olsend` funciona de manera similar pero, en este caso, es una corrección de compatibilidad entre VIeCD y el ejemplo `olsession` más abajo.

Consulte `/vipodsdk/apps/olsend/readme.pdf` para obtener instrucciones sobre cómo usar el ejemplo `olsend`.

### OLSESSION

A partir del ejemplo server2 anterior, olsession es un servidor que establece un inicio de sesión con Microsoft Outlook. El servidor olsession actúa como un enlace de sesión, que tiene las características siguientes:

- Permite a VIeCD interactuar con un programa, en este caso Outlook, que puede requerir interactuar con el usuario. Si Outlook todavía no se está ejecutando cuando se inicia el servidor, se requiere inicio de sesión y clave.
- Se aprovecha del estado de la sesión. El estado de la sesión es un inicio de sesión, seguido de varias transacciones y, a continuación, el cierre de la sesión al salir.
- No es compatible con la interacción directa con VIeCD porque no tiene un retval y no utiliza stdin o stdout.

El cliente olsend, en combinación con el servidor olsession, proporciona una solución de enlace de VIeCD a Outlook que puede usarse en situaciones de uso habitual, de la misma manera que la utilidad pública blat se puede usar como interfaz de VIeCD y servidores de correo electrónico SMTP/POP3. Para más información sobre la utilidad blat, consulte http://www.interlog.com/~tcharron/blat. html.

Además, la implementación de olsession muestra:

- Cómo usar los ejemplos server2 y client de VIeCD SDK como interfaz de VIeCD con programas de servidor bastante sofisticados (en este caso, Microsoft Outlook).
- Que los componentes de VIeCD SDK son compatibles con programas C++. El código olsession es C++, enlazado con las bibliotecas vtpdsession y OOC (ambas escritas en ANSI C).
- Que VIeCD se puede integrar con aplicaciones de Microsoft a través de la interfaz COM. La mayoría de aplicaciones de Microsoft tienen interfaces programáticas que pueden ser accedidas a través de COM.

Para obtener instrucciones sobre cómo usar el ejemplo de olsession, consulte `/vipodsdk/apps/olsession/readme.pdf`.

### WRAP

El ejemplo wrap muestra cómo utilizar la biblioteca `vtpdwrap` para leer, modificar y crear una regla de distribución.

Este ejemplo también muestra cómo notificar a VIeCD que una regla de distribución ha sido modificada de forma programática. Cuando se agrega, modifica o elimina una regla de distribución del entorno de VIeCD, VIeCD debe volver a evaluar todos los trabajos VIeC completados y no procesados en términos de su elegibilidad para el procesamiento de VIeCD, en el contexto de un nuevo conjunto de reglas. Cuando se realizan cambios de reglas en la interfaz de usuario gráfica de VIeCD, VIeCD puede realizar esta tarea de forma automática dado que era el origen del cambio. Sin embargo, si un cambio de regla se realiza debido a las acciones de un programa externo, VIeCD debe ser notificado de la manera mostrada en el ejemplo wrap.

Consulte `/vipodsdk/apps/wrap/readme.pdf` para obtener instrucciones sobre cómo usar el ejemplo wrap.

## Bibliotecas

Las bibliotecas proporcionadas con VIeCD SDK se pueden usar para generar reglas de distribución en aplicaciones o en otros procesos de cliente, y para implementar código de servidor de sesión.

### VTPDWRAP

`vtpdwrap` es una biblioteca de contenedor que puede utilizarse para generar, leer, modificar o escribir reglas de distribución de VIeC. Se recomienda el uso de la biblioteca de `vtpdwrap` en lugar de manipular directamente las reglas de distribución, ya que aísla al proveedor de la solución de los efectos de cambios en el formato del archivo de reglas de distribución subyacentes.

### VTPDSESSION

Algunos ejemplos proporcionados con VIeCD SDK ilustran cómo establecer un puente con un proceso de back-end, o un proceso que requiere alguna forma de estado sobre un conjunto de transacciones. Estos ejemplos se implementan mediante un modelo cliente/servidor basado en sockets. La implementación del código de servidor de la sesión subyacente es bastante genérico, y ha sido extraído en su propia biblioteca, `vtpdsession`. La biblioteca se usa en los ejemplos `server2` y `olsend`, y también puede ser usada por los proveedores de la solución si se puede usar un enfoque basado en sockets para las implementaciones de puente de sesión.

## Utilidades

Para las actividades de integración y desarrollo, es útil poder supervisar la comunicación entre VIeCD y el programa de back-end. Para esta comunicación, VIeCD SDK proporciona la utilidad VIeCD In-Circuit Emulator (vtpdice). vtpdice se puede interponer entre VIeCD y un programa de back-end para proporcionar información de diagnóstico además de la inserción automática de errores a fin de facilitar su resistencia durante las pruebas. Además de ser una utilidad muy útil, vtpdice sirve como ejemplo ilustrativo de cómo interponer un proxy entre VIeCD y un programa de back-end. El código fuente completo de vtpdice se incluye con el SDK de VIeCD.

Para obtener más información, consulte VIeC Dispatch In-Circuit Emulator.

# VIeC Dispatch In-Circuit Emulator

Este capítulo incluye:



El VIeC Dispatch In-Circuit Emulator (vtpdice) proporciona una herramienta de diagnóstico y prueba para su uso durante la integración y el desarrollo del complemento back-end de VIeCD.

Nota: La utilidad VIeCD In-Circuit Emulator (VIeCDICE) se encuentra en el SDK (vtpdice.exe).

vtpdice será llamado por VIeCD desde un proceso de servidor, de la misma manera que se realizan las llamadas a programas de servidor OEM o de otros fabricantes. Específicamente, se pretende que VIeCD realice una llamada a vtpdice con los mismos argumentos que se pasarían a un programa back-end de objetivo arbitrario.

Por ejemplo, supongamos que el objetivo es Blat (una herramienta de envío de correo electrónico POP3 de línea de comandos freeware) a través de VIeCD. Durante el desarrollo, puede ser útil realizar el seguimiento de lo que recibe Blat antes de invocarlo en el programa final. En ese caso, VIeCD puede invocar a vtpdice en vez de a Blat, con los mismos argumentos que si se llamara a Blat.

Puede resultar útil llamar a vtpdice, porque se puede configurar vtpdice para que responda de cuatro formas distintas:

| | |
|---|---|
| Caso 1 | Registre los argumentos que se pasan y envíe un valor especificado a VIeCD. |
| Caso 2 | Registre los argumentos que se pasan y emita los contenidos de los archivos especificados a stdout, a stderr o a ambos y envíe un valor de retorno especificado a VIeCD. |
| Caso 3 | Realice un seguimiento del número de veces que se ha llamado a vtpdice, con una respuesta normal, tal y como se describe en el Caso 2. Cuando se haya realizado el número de llamadas especificado, inserte un error en el flujo de respuesta y envíe el valor de retorno de error especificado a VIeCD. |
| Caso 4 | Actuar como un proxy para el programa de servidor objetivo, como blat. En esta configuración, VIeCD llama a vtpdice y vtpdice llama al programa objetivo. vtpdice intercepta y registra los stdout, stderr y retval del programa de destino, y también los reenvía a VIeCD. |

## Uso de vtpdice

Actualmente vtpdice solo puede utilizarse en sistemas Windows.

vtpdice será transparente en lo referente a los argumentos que se le pasan, en comparación con los argumentos pasados de forma potencial a un determinado programa objetivo. Esto significa que la información o la configuración de metadatos de prueba no se puede pasar a vtpdice a través de la línea de comandos de vtpdice. Como consecuencia:

- Cada prueba de vtpdice tiene su propio directorio único.
- Cada directorio de prueba debe contener como mínimo un archivo denominado `vtpdice.tst`.
- vtpdice lee `vtpdice.tst` para la información de configuración de la prueba.
- vtpdice adjunta toda la información registrada en el archivo `vtpdice.log` en el directorio de prueba, a menos que exista una variable de entorno `vtpdice_test_log`. En cuyo caso, toda la información registrada se anexa al archivo especificado a través de dicha variable de entorno.
- Después de cada llamada, vtpdice incrementa el número de veces que se ha llamado al archivo `vtpdice.rct` para realizar el seguimiento del recuento de registros. Este es el número de veces que se ha llamado a vtpdice en el contexto de dicho directorio de prueba. vtpdice realiza el seguimiento de este número para determinar cuándo se debe generar un error después de un determinado número de llamadas, si esa es parte de la llamada de prueba. Para restablecer el número de registros, elimine este archivo.

Al realizar la llamada, vtpdice examina el contenido de la variable de entorno `vtpdice_test_dir`. Si la variable de entorno existe, usa el contenido como la ruta del directorio para usarlo en la prueba. Si esa variable de entorno no existe, busca en su directorio de trabajo actual un archivo llamado `vtpdice.ini`. Si la variable de entorno existe, usa el contenido como la ruta del directorio de prueba. Si no existe la variable de entorno ni el archivo `vtpdice.ini`, o si el directorio de prueba especificado en estos no existe, la prueba se anula.

Si la ruta del directorio de prueba es una ruta relativa, se considera relativa con respecto al directorio de trabajo actual de vtpdice.

Con una ruta de directorio de prueba válida, vtpdice abre y lee el archivo `vtpdice.tst` en el directorio. Si el archivo no existe, se anula la prueba.

Los Casos 1–4 son descripciones de cómo configurar el contenido del archivo `vtpdice.tst` para obtener el comportamiento de prueba deseado de vtpdice. Para los casos siguientes, se presupone que:

- **vtpdice.exe** se encuentra en el directorio `C:\vtpdice`
- Un directorio de prueba `C:\vtpdice\test`
- Un archivo de configuración de prueba `C:\vtpdice\test\vtpdice.tst`
- Un archivo `C:\vtpdice\vtpdice.ini` que contiene la línea:

  ```
  prueba
  ```

### CASO 1

En el caso 1, se registran los argumentos que se pasan, y se devuelve un valor de retorno especificado a VIeCD.

`Contenido de vtpdice.tst`:

```
retvalNormal:0
```

Llamar a vtpdice:

```
vtpdice a b c
```

Contenido de `vtpdice.log`:

```
Test run #0, time: Fri Jun 13 13:12:23 2003
Invocation argc, argv:
argv[0]: 'vtpdice'
argv[1]: 'a'
argv[2]: 'b'
argv[3]: 'c'
No se especificó ningún archivo de origen de stdout. No se envió nada a stdout.
No se especificó ningún archivo de origen de stderr. No se envió nada a stderr.
Valor devuelto: 0
```

### CASO 2

En el caso 2 se registran los argumentos que se pasan y se emiten los contenidos de los archivos especificados a stdout, a stderr o a ambos, además de enviar un valor de retorno especificado a VIeCD.

Contenido de `vtpdice.tst`:

```
retvalNormal:0
stdoutSourcePathNormal:stdoutNorma1.txt
stderrSourcePathNormal:stderrNorma1.txt
```

Las rutas relativas especificadas en el archivo `vtpdice.tst` lo son en relación al directorio de prueba, no al directorio de trabajo vtpdice actual. Dos archivos, `stdoutNormal.txt` y `stderrNormal.txt`, con información que se enviará a stdout y stderr como parte de la prueba, deben figurar en el directorio de prueba.

Llamar a vtpdice:

```
vtpdice d e f
```

Contenido de `vtpdice.log`:

```
Test run #0, time: Fri Jun 13 13:12:23 2003
Invocation argc, argv:
argv[0]: 'vtpdice'
argv[1]: 'd'
argv[2]: 'e'
argv[3]: 'f'
```

Contenido del archivo `'C:\vtpdice\test\stdoutNormal.txt` enviado a stdout, como se indica a continuación:

```
--- start stdout stream ---
```

```
 el contenido de stdout normal va aquí...
--- end stdout stream ---
```

Contenido del archivo `'C:\vtpdice\test\stderrNormal.txt'` enviado a stder, como se indica a continuación:

```
--- start stderr stream ---
 el contenido de stderr normal va aquí...
--- end stderr stream ---
Valor devuelto: 0
```

### CASO 3

El caso 3 realiza el seguimiento del número de llamadas a vtpdice con una respuesta normal (como el caso 2). Sin embargo, después de un determinado número de llamadas, se inserta un error en el flujo de respuesta. Los contenidos de los archivos de error especificados se envían a stdout, stderr o a ambos, y se envía el valor de retorno de error especificado a VIeCD. Las llamadas posteriores devolverán el comportamiento normal de stdout/stderr/retval.

En este ejemplo, vtpdice inserta un error en el registro 2, el tercer registro, ya que vtpdice empieza a contar desde cero. En el error, vtpdice devuelve un valor `-1`, así como información distinta en stdout/stderr.

`Contenido de vtpdice.tst`:

```
retvalNormal:0
stdoutSourcePathNormal:stdoutNormal.txt
stderrSourcePathNormal:stderrNormal.txt
faultRecord:2
retvalFault:-1
stdoutSourcePathFault:stdoutFault.txt
stderrSourcePathFault:stderrFault.txt
```

Para restablecer el número de registros a cero, borre vtpdice.rct, si existe el parámetro.

Llamar a vtpdice:

Llamar a vtpdice cuatro veces:

```
vtpdice g h i
vtpdice j k l
vtpdice m n o
vtpdice p q r
```

`Contenido de vtpdice.log`:

```
Test run #0, time: Fri Jun 13 13:12:23 2003
Invocation argc, argv:
argv[0]: 'vtpdice'
argv[1]: 'g'
argv[2]: 'h'
```

```
argv[3]: 'i'
```

Contenido del archivo `'C:\vtpdice\test\stdoutNormal.txt` enviado a stdout, como se indica a continuación:

```
--- start stdout stream ---
 el contenido de stdout normal va aquí...
--- end stdout stream ---
```

Contenido del archivo `'C:\vtpdice\test\stderrNormal.txt'` enviado a stderr, como se indica a continuación:

```
--- start stderr stream ---
 el contenido de stderr normal va aquí...
--- end stderr stream ---
Valor devuelto: 0
Test run #1, time: Fri Jun 13 13:12:23 2003
Invocation argc, argv:
argv[0]: 'vtpdice'
argv[1]: 'j'
argv[2]: 'k'
argv[3]: 'l'
```

Contenido del archivo `'C:\vtpdice\test\stdoutNormal.txt` enviado a stdout, como se indica a continuación:

```
--- start stdout stream ---
 el contenido de stdout normal va aquí...
--- end stdout stream ---
```

Contenido del archivo `'C:\vtpdice\test\stderrNormal.txt'` enviado a stderr, como se indica a continuación:

```
--- start stderr stream ---
 el contenido de stderr normal va aquí...
--- end stderr stream ---
Valor devuelto: 0
Test run #2, time: Fri Jun 13 13:12:23 2003
Invocation argc, argv:
argv[0]: 'vtpdice'
argv[1]: 'm'
argv[2]: 'n'
argv[3]: 'o'
ERROR INSERTADO en el registro 2:
```

Contenido del archivo `'C:\vtpdice\test\stdoutFault.txt'` enviado a stdout, como se indica a continuación:

```
--- start stdout stream ---
```

```
 el contenido de stdout 'Fault' va aquí...
--- end stdout stream ---
```

Contenido del archivo `'C:\vtpdice\test\stderrFault.txt'` enviado a stderr, como se indica a continuación:

```
--- start stderr stream ---
 el contenido de stderr 'Fault' va aquí...
--- end stderr stream ---
Valor devuelto: -1
Test run #3, time: Fri Jun 13 13:12:23 2003
Invocation argc, argv:
argv[0]: 'vtpdice'
argv[1]: 'p
argv[2]: 'q'
argv[3]: 'r'
```

Contenido del archivo `'C:\vtpdice\test\stdoutNormal.txt` enviado a stdout, como se indica a continuación:

```
--- start stdout stream ---
 el contenido de stdout normal va aquí...
--- end stdout stream ---
```

Contenido del archivo `'C:\vtpdice\test\stderrNormal.txt'` enviado a stderr, como se indica a continuación:

```
--- start stderr stream ---
 el contenido de stderr normal va aquí...
--- end stderr stream ---
Valor devuelto: 0
```

### CASO 4

El caso 4 actúa como un proxy para el programa de back-end objetivo real. En esta configuración, VIeCD llama a vtpdice y vtpdice llama al programa objetivo. vtpdice intercepta y registra los stdout, stderr y retval del programa de destino, y también los reenvía a VIeCD. Por lo tanto, VIeCD cree que se realizó una llamada directa al programa objetivo y que, de hecho, este realizó el trabajo. De esta manera, vtpdice actúa como un proxy o shim que se puede insertar entre VIeCD y cualquier programa de back-end objetivo para realizar tareas de seguimiento y diagnóstico e, incluso, para recopilar información de resolución de problemas del cliente.

El VIeCD SDK contiene un simple programa de bucle para probar este caso. El programa de bucle actúa como una aplicación llamada por vtpdice.

Contenido de `vtpdice.tst`:

```
pathToProxyFor:../../../bin/loopback.exe
```

Esta ruta puede ser absoluta o relativa. Si es relativa, lo es en relación al directorio de prueba, no al directorio de

trabajo actual de vtpdice.

Llamar a vtpdice:

```
vtpdice s t u
```

Contenido de vtpdice.log:

```
Test run #0, time: Fri Jun 13 13:12:23 2003
Invocation argc, argv:
argv[0]: 'vtpdice'
argv[1]: 's'
argv[2]: 't'
argv[3]: 'u'
actuando como proxy para: 'C:\vtpdice\test\../../../bin/loopback.exe'
Argumentos pasados:
argv[0]: 'C:\vtpdice\test\../../../bin/loopback.exe'
argv[1]: 's'
argv[2]: 't'
argv[3]: 'u'
--- start received stdout stream ---
Hola, soy el programa situado en
'C:\vtpdice\test\../../../bin/loopback.exe'.
Se me llamó con 3 argumentos, de esta manera:argv[1]: 's'argv[2]:
't'argv[3]: 'u'End of argument list.
Se envía 'Hola, stderr!' a stderr...
Y ahora envía '123' como valor de retorno...Adios!
--- end received stdout stream ---
--- start received stderr stream ---
Hola, stderr!
--- end received stderr stream ---
Valor devuelto: 123
```

## Uso de vtpdice en modo por lotes

vtpdice se puede ejecutar en modo por lotes desde la línea de comandos. Este tipo de ejecución se realiza para usar vtpdice y programas de back-end especificados para garantizar que las pruebas se realizan de la forma esperada antes de ejecutar trabajos de diagnóstico reales a través de VIeCD. Puede utilizar el modo por lotes para realizar una verificación preparada para garantizar que las pruebas funcionan correctamente antes de que sean invocadas por VIeCD y vtpdice.

Mediante la configuración o el ajuste de las variables de entorno *vtpdice_test_dir* y *vtpdice_test_log* podrá ejecutar diversas pruebas por lotes.

Busque el archivo `vipodsdk/bin/runtests.bat`,el resultado de ejecutar la prueba por lotes aparecerá en el archivo `alltest.log`. Al analizar el archivo por lotes, los directorios de prueba están ubicados en `vipodsdk\src\examples` y el contenido del directorio de prueba proporciona detalles del uso de vtpdice en modo por lotes y sin lotes.

Para ejecutar la prueba por lotes, cambie la ruta absoluta de las variables de entorno para que coincida con la ubicación de `vipodsdk`.