# CONTROL DE CAMBIOS

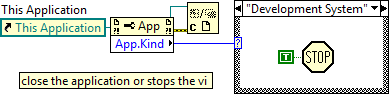
|  |  |  |
| --- | --- | --- |
| **Fecha** | **Actor** | **Acción** |
| 30-11-2012 | Dictino | Crea el documento inicial. |

# Cambios de la versión

Esta versión de JiL parte de la versión 2010 original de Hector y realiza los siguientes cambios:

**1)** Se resuleve el problema del Path cuando se compila, para ello se utiliza el bloque  que devuelve el path del proyecto o del ejecutable en lugar de obtener el path del vi (que incluye el nombre del ejecutable)

**2)** Se resuleve el problema del cierre en la versión compilada para ello se usa el siguiente código en el tratamiento del evento del menú “exit”:



El código comprueba si la aplicación es un ejecutable o no y ejecuta la acción de cierre que corresponde.

**3)** Se centraliza la carga de las configuraciones desde un vi ConfParameters.vi  este vi admite como entrada un comando y un cluster de configuración y devuelve el cluster de configuración y un indicador de error.

Los comandos son “load” que carga la configuración desde el archivo config.XML “save” que guarda la configuración entrante y “defParam” que devuelve la configuración por defecto.

Si se produce un error en la carga de la configuración se notifica a través del indicador de error, además se reconstruye el archivo de configuración usando los parámetros por defecto (la siguiente vez que se ejecute no habrá error).

**4)** Se centraliza la escritura en el “connection repport”, para ello se crea Report.vi que admite como entrada dos strings que contiene el cliente IP:puerto y el mensaje. Este subVi escribe en la variable global “ConnectionReport.vi” que se gestiona de forma similar a ErrorsReport.

**5)** Se establece un sistema simple de autenticación con usuario y contraseña que se guarda en la configuración. La autenticación se activa/desactiva con la opción useAuthentication del archivo de configuración.

**6)** Se elimina código muerto: Lectura/escritura de la base de datos, Comprobación de la BD, envío de correos electrónicos, Smart device así como muchos controles e indicadores asociados que no son necesarios. Para que dicho código no se pierda se conserva una copia en JilBackup26-11-2012.zip en la carpeta sources/OLD.

**7)** Se resuelve el problema de la ayuda contextual (el enlace <http://lab.dia.uned.es/personalweb/jiloverview.html> está roto) en su lugar redirijo a <http://profesores.eie.ucv.cl/hvargas/jil/jil.html>

**8)** Se oculta el indicador “Absolute Appication path” (habrá cambios en el futuro cuando haya varias conexiones y se necesitará otra forma de mostrar la información, además es redundante ya que el conection report muestra los vi que se han abierto)

**9)** La mayoría de los terminales de los controles e indicadores se mueven a INITIALIZATION para que sea más fácil encontrarlos (ya que algunos de ellos son ocultos y conviene saber que variables se están usando)

**10)** Cambios menores en el “comand parser” para resolver errores (sobre todo al quitar código muerto) y un posible fallo que encontré en “getFolatArray” que ahora está conectado a “indicators formated” y no a “controls formated” así como muchos pequeños cambios comentados en el código.

**11)** Este es **el cambio más grande de la versión**. Se crea el subVi “Parser.vi”  que se encarga de leer y ejecutar todos los comandos que llegan dese una conexión. Esto permite separar totalmente el mantenimiento del protocolo de JiL (en el que va a trabajar Jesús para implementar en Ejs algunos de los métodos nuevos) de la interfaz gráfica de modo que los nuevos cambios

El comando admite como entradas un indicador de error y un cluster ParserState y los devuelve a la salida. El cluster “ ParserState” contiene todos los datos que el Pharser necesita para hacer su trabajo:

-vi reference: es la referencia del Vi abierto

-absolute path: es el path del Vi abierto

-User y Pasword: son los datos de autenticación en el servidor JiL

-UserRecieved y PaswordRecieved: Son el usuario y contraseña recibidos del cliente (que se comparan con los del servidor para autenticar)

-Exec.State: Indicador del estado de ejecución del Vi

-Indicators formated y Controls formated: Arrays con los nomres y descripciones de tipo de los indicadores y controles del vi abierto

-Disconect: Booleano que cuando se pone a “true” indica que hay que cerrar la conexión

-connID: identificador del socket de conexión TCP/IP que permite enviar y recibir datos del cliente

-Package Size y Senditn Rate (ms): Parámetros que usa TCPAS.vi (aunque no se si realmente funciona la conexión asíncrona por el comentario que ponía hector)

-useAuthentication: indica si hay que usar autenticación

# Funcionamiento básico

# 1) Se inicia la aplicación leyendo los datos del XML y cargándolos en los indicadores correspondientes, se construye también el menú contextual (esta parte para mi es algo confusa jajajaja) y se activan o desactivan características.

**2)** Se inician 3 bucles en paralelo (en realidad se inician cuando llega un evento de cambio en el botón “start”):

**A)** Uno de ellos “UPDATING INTERNAL ERRORS” se encarga de actualizar “error report” y “connection repport” en la vista.

**B)** El otro se “MAIN EVENTS OF THE APPLICATION” ocupa de atender los eventos del menú (entre ellos el cambio en el botón start que genera una ocurrencia de evento que activa el bloque C)

**C)** El último “ESTABLISH CONNECTIONS AND EXECUTE THE PARSER” escucha la conexión en el Puerto de JiL y cuando la abre inicializa “PharserState” con datos de conexión y parámetros y luego entra en un bucle en el que se ejecuta “Parser.vi” actualizando el estado. Cuando el cliente se desconecte el cambie el estado de “Disconect” a “true” en el “ParserState” y eso cierra el bucle.

**3)** si se pulsa “stop” se desencadena un evento que cambia el valor del estado de  y se encarga de cerrar los bucles.

**4)** Durante todo el proceso se escriben mensajes de conexión con  y mensajes de error con .

# PROBLEMAS CONOCIDOS:

**1)** No he sabido quitar la opción muerta del menú contextual (Smart device)

**2)** No se en que estado se encuentra TCPAS.vi (hay un comentario de hector que dice que no funciona adecuadamente)

**3)** La autenticación no está probada y el método no es el más adecuado ya que si el cliente solicita la autenticación en ejs el servidor no se si haría ninguna comprobación… esto hay que pensarlo con calma. Para esto he dejado “UserRecieved” y ”PaswordRecieved”.

**4)** Deberíamos documentar el protocolo de comunicación de JiL (no se si Héctor lo tendrá hecho en la tesis). Al menos los comandos más usados (conectar, desconectar, arrancar, parar y los get y set de los tipos de datos básicos)

**5)** JILdist produce un error

6) Hay errores en la consola de “internal errors” que habrá que solucionar en el futuro (ya estaban antes)