MiddleWare y Applet

Documentación del proceso de instalación, configuración y

puesta en marcha del MiddleWare y Applet

Rolosa HyJ S.A. - MICITT

20 de Enero de 2014

Resumen

El presente manual permite la instalación, configuración y puesta en marcha de la solución de MiddleWare y Applet del Micitt para el DCFD

Tabla de contenido

Introducción

El MiddleWare

Escrito en el lenguaje C++, actualmente comprende un conjunto de librerías utilizables para la interacción con las tarjetas inteligentes de la Firma Digital de Costa Rica. Provee de un acceso nativo a las funcionalidades de la tarjeta inteligente por medio de la librería resultante de la compilación del MiddleWare que sigue el estándar RSA - PKCS#11.

Para una fácil utilización de la librería PKCS#11 del MiddleWare en otros lenguajes de programación como ser C# o JAVA, es recomendable utilizar alguna de las librerías "Wrappers" para PKCS#11 que se pueden encontrar con licencias tanto comerciales como gratuitas.

El presente manual hace uso de la librería Net.Pkcs11 para C#. (ver http://sourceforge.net/projects/pkcs11net)

El Applet

Es un componente para navegadores web que expone funcionalidad de la tarjeta inteligente a las aplicaciones web. En la Figura 1, Applet se puede ver una captura de pantalla del Applet

Figura 1, Applet

- Las principales características del Applet son:
- Fácil integración con las aplicaciones web
- Seguridad y privacidad de la información contenida en la tarjeta
- Manejo Interactivo de la tarjeta

El Applet utiliza tecnología Java, esto minimiza los requerimientos de los navegadores web. En la actualidad, la plataforma Java 6 es soportada por el Applet.

El Applet en la actualidad depende completamente de la librería resultante de la compilación del MiddleWare que sigue el estándar RSA - PKCS#11, para la interacción con las tarjetas inteligentes de la Firma Digital de Costa Rica.

El código de la solución de MiddleWare y Applet

Está actualmente ubicado en un servidor SVN de Google:

http://dcfd-mw-applet.googlecode.com

Para obtener una copia del código se debe utilizar un cliente SVN y realizar la operación de Checkout:

svn checkout http://dcfd-mw-applet.googlecode.com/svn/trunk/

A continuación se muestra como realizar el procedimiento de Checkout del código de la solución.

Para este ejemplo, se utilizará el diente SVN del IDE NetBeans, este procedimiento es válido para todos los Sistemas Operativos actualmente soportados por el IDE NetBeans, y ha sido verificado en Windows (8) y Linux (Ubuntu 12.10). En la Figura 2, NetBeans SVN Checkout, podemos observar cómo realizar el procedimiento de descarga del código.

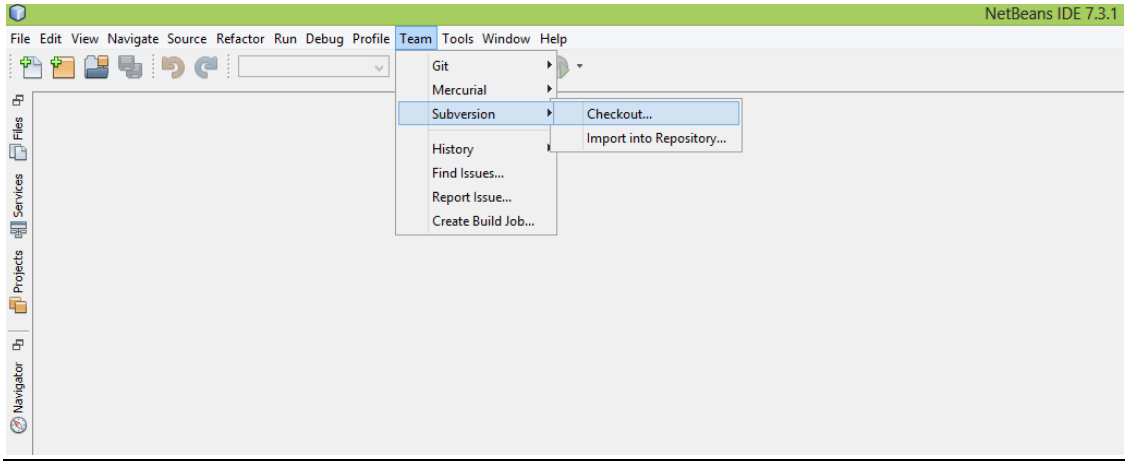

Figura 2, NetBeans SVN Checkout

Debemos especificar el URL para realizar el Checkout , esta es la URL del servidor de Google. Como se aprecia en la Figura 3, NetBeans, SVN URL, no se requiere especificar ningún Usuario ni Contraseña, puesto que el Checkout del código se realiza en modo "read-only".

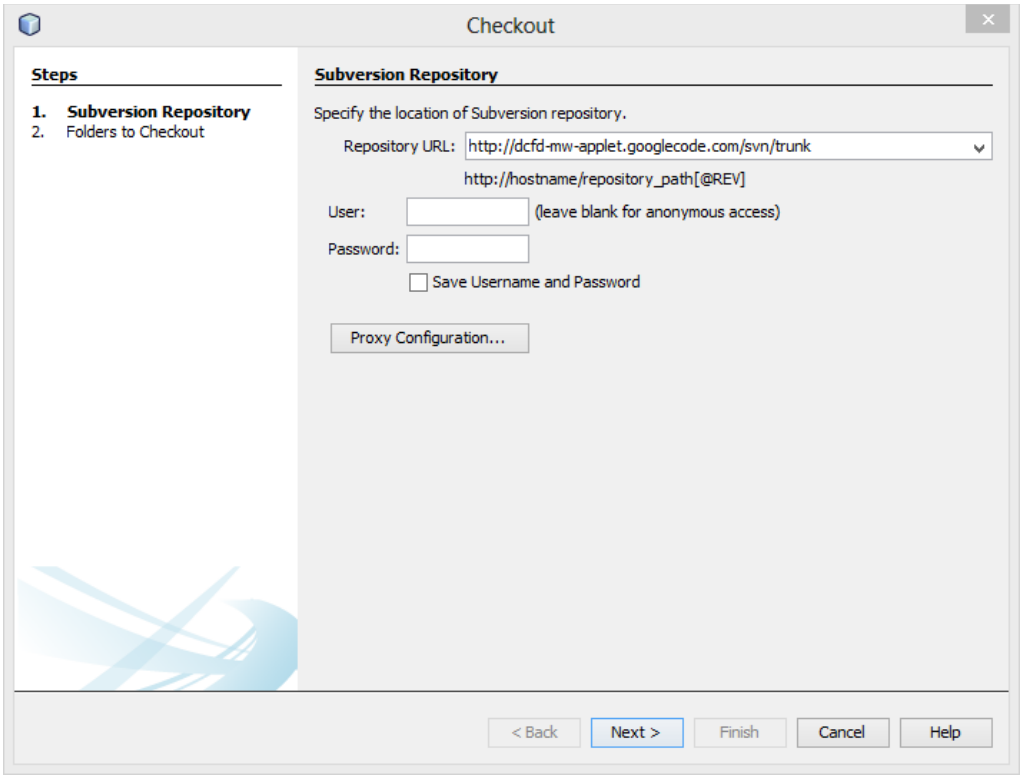

Figura 3, NetBeans, SVN URL

Para finalizar el procedimiento de Checkout es necesario especificar el directorio local donde se realizará la descarga del código. La Figura 4, NetBeans SVN Checkout - local, muestra el último paso para el Checkout.

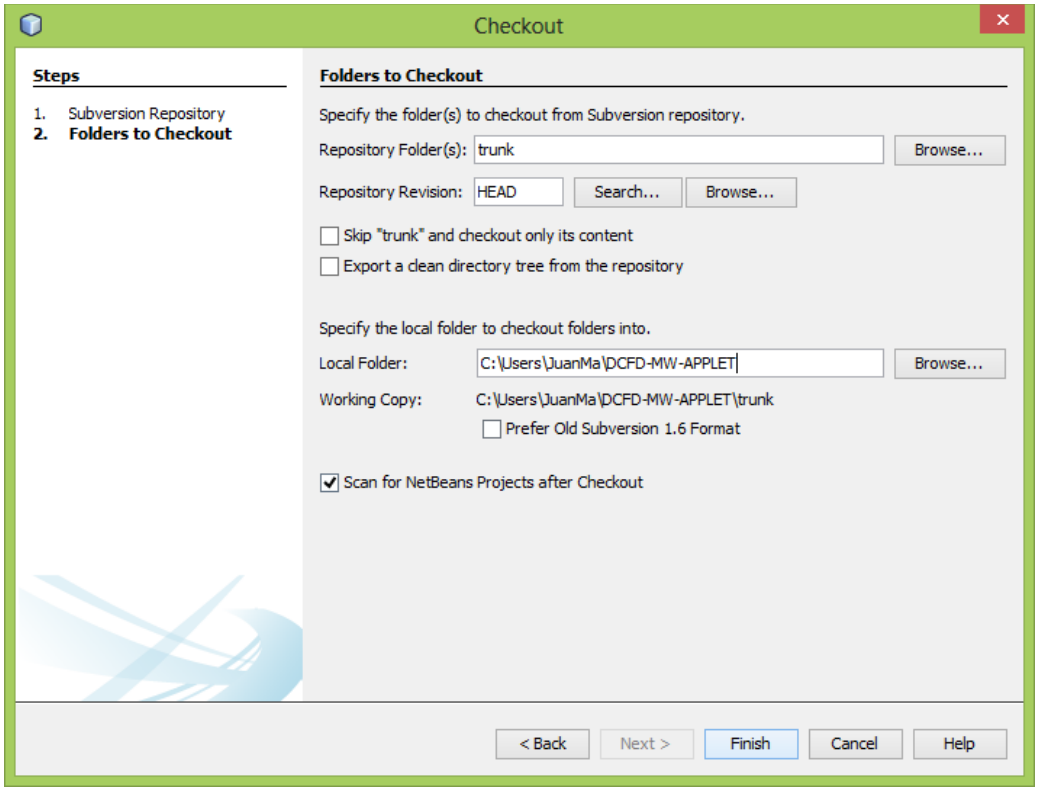

Figura 4, NetBeans SVN Checkout - local

Compilación

MiddleWare

Windows

Para la configuración en Windows se requiere tener instalada una versión del IDE **Microsoft Visual Studio 2010**. *(*http://go.microsoft.com/fwlink/?LinkID=159121&clcid=409*)*

El presente manual supone la utilización de Microsoft Visual Studio 2010 Ultimate, Version 10.0.30319.1

Una vez finalizada la descarga del código de la solución de MiddleWare y Applet, abrir el archivo de solución **beid.sln** desde:

DCFD-MW-APPLET\trunk\mw\VS_2010

Con la solución abierta, antes de realizar alguna compilación, es necesario editar el "**Platform ToolSet**" de cada uno de los proyectos y sustituir el valor que viene por defecto **Windows7.1SDK**, por el **ToolSet** disponible en el sistema. Esto puede realizarse ingresando a las Propiedades de Cada Proyecto, haciendo click derecho en el "**Solution Explorer**" y eligiendo "**Properties**". En la Figura 5, Propiedades de Proyecto se muestra claramente cómo se puede realizar el cambio de **Toolset**, asegurarse de que el cambio en cada proyecto sea realizado para Todas las Configuraciones (Debug, Release, etc) de compilación:

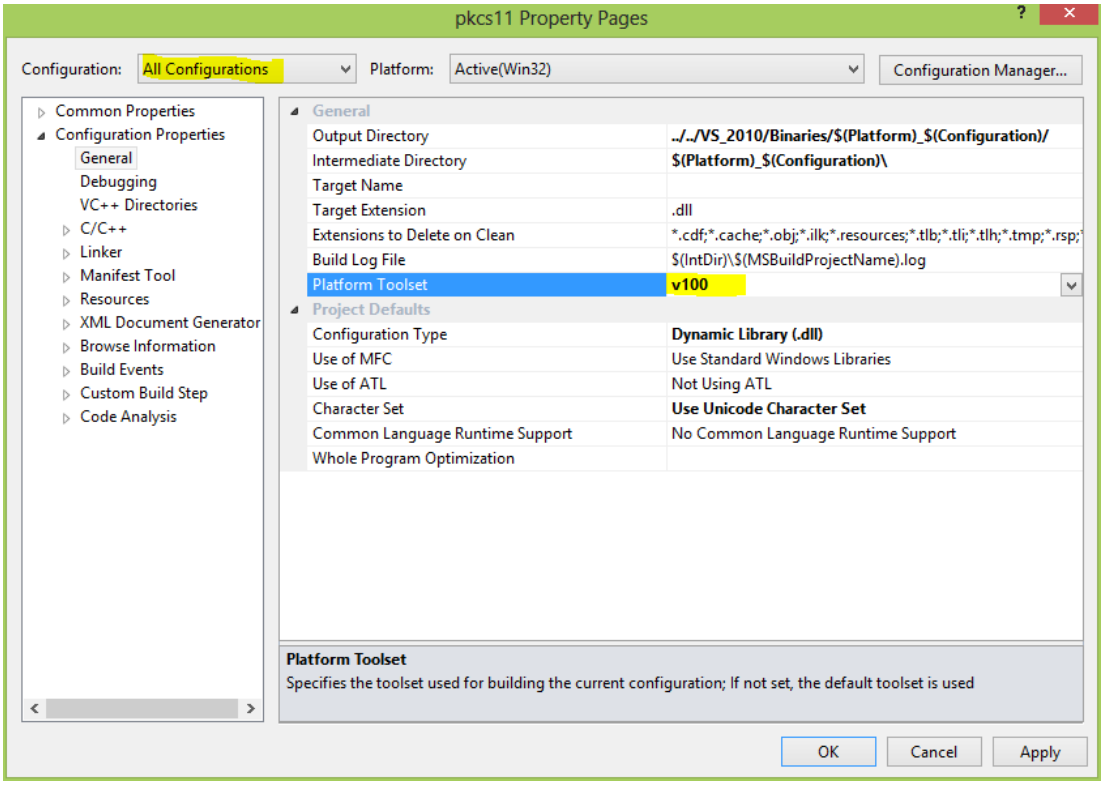

Figura 5, Propiedades de Proyecto

A continuación se puede compilar el proyecto **pkcs11,** como se muestra en la Figura 6, Compilar Proyecto pkcs11, haciendo click derecho en el "**Solution Explorer**" y eligiendo "**Build**".

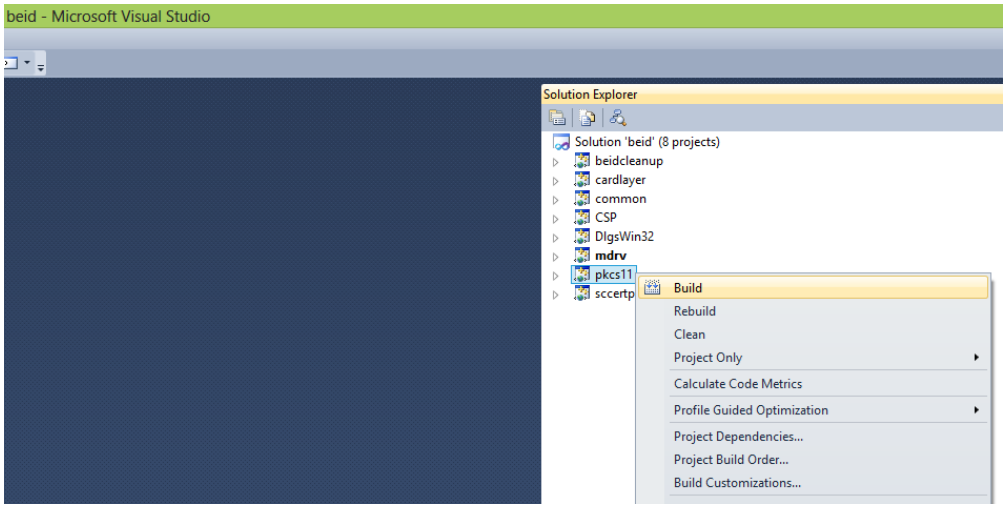

Figura 6, Compilar Proyecto pkcs11

Es posible que la compilación genere algún error relacionado con la svn_revision (ver Figura 7, Error svn_revision), esto se debe a que la solución está buscando algún cliente SVN local para poder determinar el número de revisión SVN. Si esto sucede, instalar algún cliente SVN o, compilar los proyectos uno a uno en el siguiente orden:

- common
- DlgsWin32
- cardLayer
- pkcs11

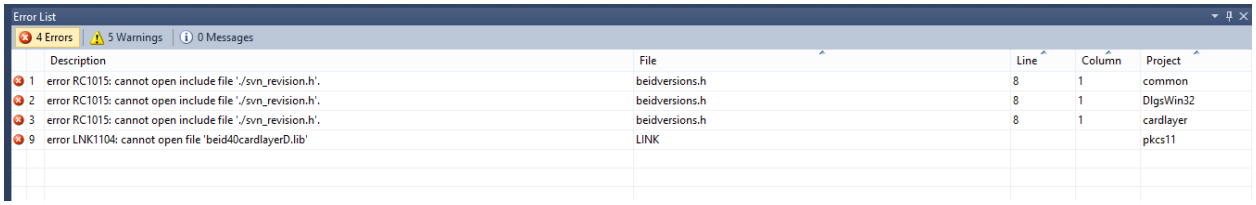

Figura 7, Error svn_revision

Una vez finalizada la compilación, dispondremos de las librerías resultante (en versión Debug o Release) en:

trunk\mw\VS_2010\Binaries\Win32_Debug\beidpkcs11D.dll

trunk\mw\VS_2010\Binaries\Win32_Release\beidpkcs11.dll

Las librerías resultantes podrán ser utilizadas en aplicaciones para Windows, para lo cual se puede registrarlas utilizando la aplicación RegSvr32.exe que viene por defecto en las distribuciones de Windows.

Linux y Mac

Para la configuración en Linux y Mac es necesario disponer del entorno de desarrollo adecuado para la compilación de proyectos en C++. EL código del MiddleWare de la solución, en la actualidad no está ligado a ningún IDE ni bajo Linux ni bajo Mac.

Para el presente manual supone la utilización de Linux Ubuntu 12.04, las instrucciones para la configuración de un entorno de desarrollo para aplicaciones C/C++ en Mac OS X salen del alcance de este documento. A continuación se detalla la configuración de un entorno adecuado para la compilación y configuración del MiddleWare para Linux.

La configuración deberá realizarse desde una ventana de consola.

Como primer paso opcional se puede instalar el cliente SVN para Linux con el siguiente comando:

```
$ sudo apt-get install subversion
```

Si no se desea instalar el cliente SVN, se puede utilizar cualquier otro cliente (como NetBeans) para realizar el Checkout del código de la solución del MiddleWare.

A continuación, se debe instalar la plataforma de compilación para aplicaciones en C/C++:

```
$ sudo apt-get update
$ sudo apt-get upgrade
$ sudo apt-get install build-essential
```

Los comandos de **update** y **upgrade**, sirven para actualizar los repositorios de software de Ubuntu.

Habiendo completado la instalación de **build-essential**, se debe verificar que el compilador haya sido instalado correctamente, utilizando los comandos:

```
$ gcc -v
$ make -v
```

Se requiere instalar los programas utilitarios para la correcta configuración del MiddleWare:

```
$ sudo apt-get install automake autoconf
$ sudo apt-get install libtool
$ sudo apt-get install libgtk2.0-dev
$ sudo apt-get install libpcsclite-dev
$ sudo apt-get install pcscd
```

Luego de haber completado la instalación de todos los paquetes requeridos para la configuración de un entorno de desarrollo para aplicaciones C/C++, ya es posible realizar el Checkout del código de la solución de MiddleWare y Applet utilizando el comando **svn**. Los siguientes pasos son válidos tanto en Linux como en Mac.

```
$ svn checkout http://dcfd-mw-applet.googlecode.com/svn/trunk/
dcfd-mw-applet
```

Una vez completado el Checkout, cambiar al directorio que contiene el código del MiddleWare:

```
$ cd  dcfd-mw-applet/mw
```

Ya dentro del directorio que contiene el código del MiddleWare, se deberá configurar y prepararlos archivos para la compilación:

```
dcfd-mw-apple/mw$ ./bootstrap.sh
```

A continuación es necesario generar los archivos Makefile para la compilación, para lo cual se requiere ejecutar el comando siguiente:

```
dcfd-mw-applet/mw$ ./configure
```

En este punto ya es posible realizar la compilación del Middleware, ejecutar el comando **make**:

```
dcfd-mw-applet/mw$ make
```

EL proceso de compilación podrá consumir varios minutos al final de los mismos ya se puede proceder a la instalación en el sistema de las librerías resultantes para que puedan ser utilizadas por distintas aplicaciones. Para este fin ejecutar el siguiente comando:

```
dcfd-mw-applet/mw$ sudo make install
```

Para finalizar con la instalación de las librerías resultantes, se requiere registrar las mismas para que cualquier aplicación pueda utilizarlas, para esto es requerido, utilizando un editor (en este ejemplo se utiliza el editor: "vi"), crear el archivo middlewareLibs.conf como se muestra a continuación:

```
dcfd-mw-applet/mw$ sudo vi
/etc/ld.so.conf.d/middlewareLibs.conf
```

En este nuevo archivo introducir el texto: "/usr/local/lib" (sin incluir las comillas) y luego ejecutar el comando:

```
dcfd-mw-applet/mw$ sudo ldconfig
```

Librería PKCS11.NET (Wrapper para C#)

La Librería PKCS11.NET - Wrapper para C#, permite la utilización de la librería resultante de la compilación del MiddleWare que sigue el estándar RSA - PKCS#11.

Es necesario hacer algunos ajustes al código fuente de la librería PKCS11.NET para que pueda trabajar correctamente con la librería PKCS11 del MiddleWare.

El código esta actualmente ubicado en un servidor SVN de SourgeForge:

https://svn.code.sf.net/p/pkcs11net/code

Para obtener una copia, utilizar un cliente SVN y realizar la operación de Checkout:

Luego de realizar la descarga, abrir el archivo de solución utilizando Microsoft Visual Studio 2010, luego es necesario ajustar el "**Calling Convention**" del **PKCS11NET** para poder utilizar ese wrapper con la librería resultado pkcs11 del MiddleWare, esto debido a que el Calling Convention que se utiliza en el MiddleWare es el **CDECL**.

Para esta tarea, se tiene que agregar antes de cada función "**delegate**" el siguiente código:

[System.Runtime.InteropServices.UnmanagedFunctionPointerAttribute(System.Runtime.InteropServices.CallingConvention.Cdecl)]

Todas las funciones "**delegate**" están listadas en el directorio "**delegate**" dentro del directorio de la solución. La Figura 8, CDECL - Delegate C_CLoseAllSessions, muestra un ejemplo del cambio que se requiere.

```
[System.Runtime.InteropServices.UnmanagedFunctionPointerAttribute(System.Runtime.InteropS
ervices.CallingConvention.Cdecl)]
        internal delegate CKR C_CloseAllSessions(
                uint slotID
        );
```

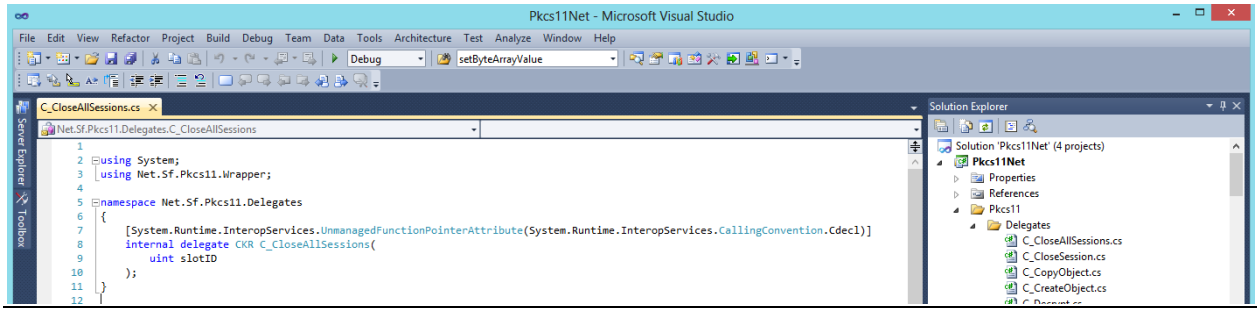

Figura 8, CDECL - Delegate C_CLoseAllSessions

También es necesario ajustar el " **alignment** " de las estructuras que se manejan en la librería pkcs11net, puesto que algunas de las estructuras que se utilizan en la librería no están con el " **alignment** " de valor "1".

Todas las estructuras del Middleware están con el "**alignment**" de valor "1" y cualquier librería Wrapper que se trate de utilizar con la librería resultante de la compilación del MiddleWare que sigue el estándar RSA - PKCS#11, debe exponer todas sus estructuras con el valor de " **alignment** " correcto.

Para esta tarea, se tiene que agregar:

Pack=1

para cada una de la estructuras que están listadas en el directorio Wrapper. La Figura 9, Alignment - "Pack = 1", muestra un ejemplo del cambio que se requiere.

Para la estructura CK_ATTRIBUTE en CK_ATTRIBUTE.cs cambiar:

```
[StructLayout(LayoutKind.Sequential, CharSet = CharSet.Unicode)]
```

por:

```
[StructLayout(LayoutKind.Sequential, CharSet = CharSet.Unicode, Pack = 1)]
```

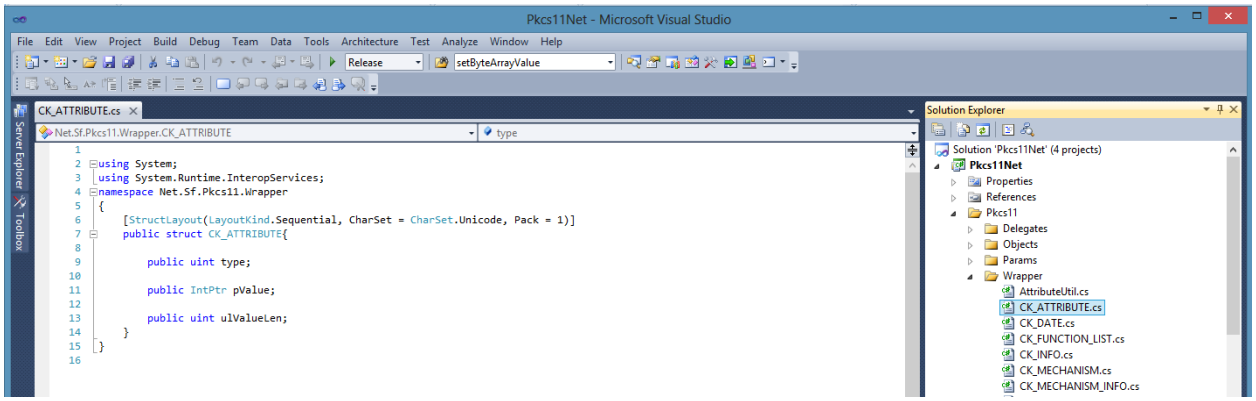

Figura 9, Alignment - "Pack = 1"

Con este último cambio, ya se puede compilar la , dispondremos de las librerías resultante (en versión Debug o Release) en:

PKCS11NET\trunk\pkcs11Wrapper\bin\Debug\Net.Pkcs11.dll

PKCS11NET\trunk\pkcs11Wrapper\bin\Release\Net.Pkcs11.dll

Applet

El Applet ha sido concebido para soportar una plataforma especifica de Java sin depender del Sistema Operativo en el que se desee utilizar el mismo. Es en este sentido que en la actualidad, solamente esta soportada la plataforma Java 6.

El presente manual detalla la configuración, instalación y puesta en marcha del Applet considerando el soporte a la plataforma Java 6 en el sistema operativo que se desee y que cumpla con los pre-requisitos que se listan a continuación.

Pre-requisitos

1. Plataforma Java 6
    a. Oracle Java Runtime Environment 6 (update 45)
    b. Oracle Java Development Kit 6 (update 45)

    Es imprescindible asegurarse que las distribuciones de JRE y JDK correspondan a las provistas por SUN - Oracle. En varias distribuciones Linux (Ubuntu 12.04, Fedora 9, 10, 11, 12), el JRE por defecto es el IcedTea JRE basado en OpenJDK; el Applet no tiene soporte funcional completo para OpenJDK.

    Apple solamente soporta Java 6 runtime en las versiones de Mac OS X desde Snow Leopard.

    Los plugins para soporte de Java en el explorador web que se utilice, tienen que ser compatibles con Java 6

    La instalación y configuración de Oracle Java 6u45 en algún sistema operativo sale del alcance de este manual. Como referencia es posible revisar el siguiente tutorial de instalación de Java6u45 en Ubuntu 12:

    http://hendrelouw73.wordpress.com/2013/05/07/how-to-install-oracle-java-6-update-45-on-ubuntu-12-10-linux/

2. Apache Maven
    a. Apache Maven 3.0.4 o superior

    La administración del código fuente del Applet ha sido realizada utilizando Apache Maven. La instalación y configuración de Apache Maven en algún sistema operativo sale del alcance de este manual. Existen numerosos tutoriales en Internet al respecto.

IDE - NetBeans

Para hacer más eficiente el trabajo con el proyecto del Applet es recomendable utilizar el **IDE NetBeans** el cual en su **versión 6.9.1 para Java EE**, contiene soporte nativo para **Maven 3** y de igual forma incluye de una instancia de **GlassFish 3 Server**, servidor de aplicaciones web, el cual será utilizado para poder desplegar aplicaciones web que utilicen el Applet y consuman los servicios que el framework del Applet provee.

El presente manual detalla la configuración, instalación y puesta en marcha del Applet considerando el IDE NetBeans 6.9.1 para Java EE, este puede ser obtenido desde:

https://netbeans.org/downloads/6.9.1/index.html

Una vez instalado el IDE, abrir NetBeans 6.9.1, y luego abrir el proyecto Applet que se encuentra dentro del directorio que se acaba de descargar del repositorio SVN (**Menu File > Open Project..)** como se indica en la Figura 10, Abrir proyecto - Applet.

Con el proyecto abierto, seleccionar el Proyecto **eID Applet Project** y luego presionar la teda [**F11**] o hacer click derecho sobre el proyecto y elegir "**Build**" del menú contextual, **Netbeans** realizará el proceso de compilación, para lo cual también realizará varias descargas de las dependencias que el presente proyecto tiene (ver Figura 11, Compilación del Applet - Descarga de Dependencias) este proceso puede demorar varios minutos dependiendo de la velocidad de su conexión de Internet.

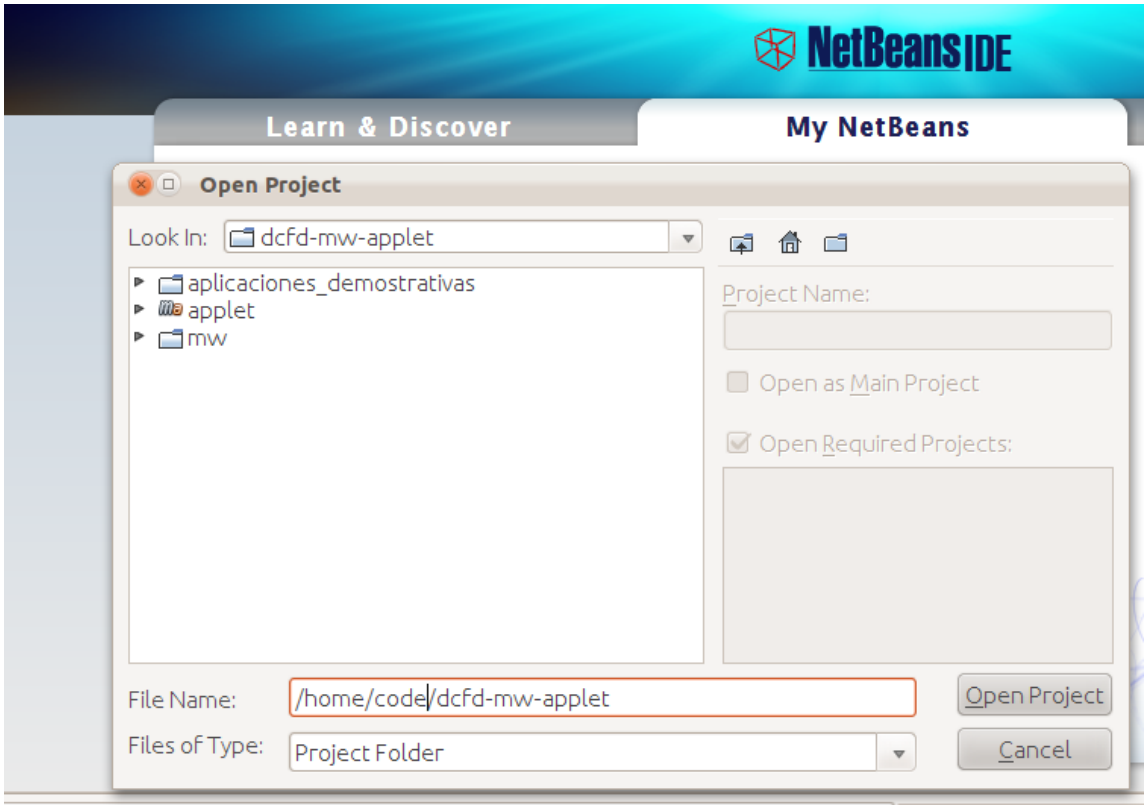

Figura 10, Abrir proyecto - Applet

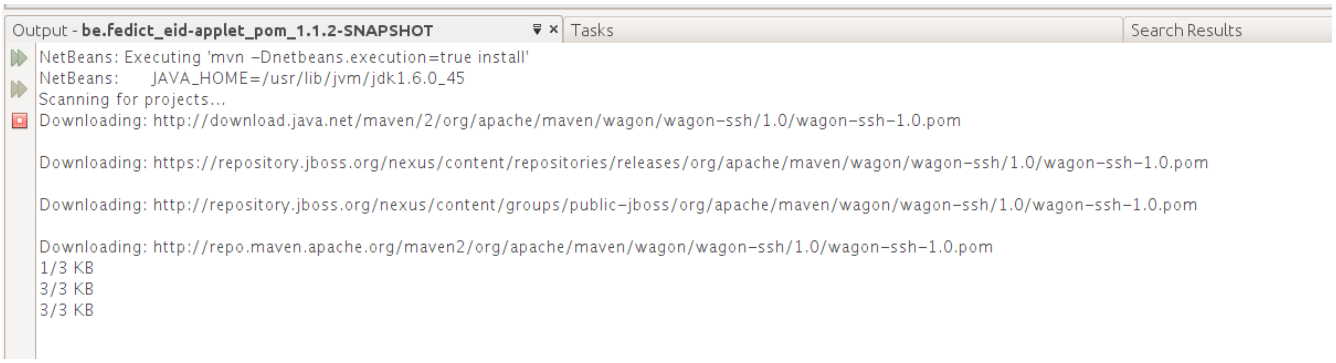

Figura 11, Compilación del Applet - Descarga de Dependencias

Una vez finalizada la compilación del proyecto, se obtendrá un resumen como el que se muestra en la Figura 12, Compilación del Applet - Exitosa.

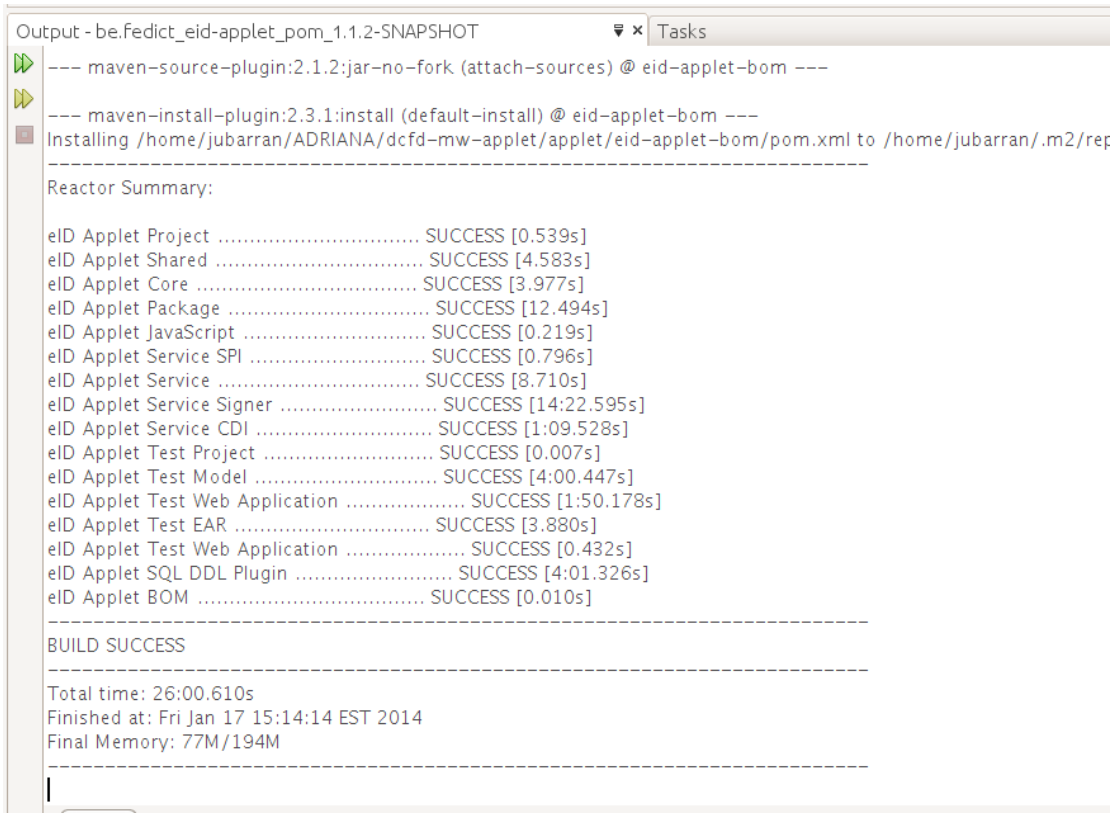

Figura 12, Compilación del Applet - Exitosa

Si por alguna motivo, la compilación es detenida a la mitad del proceso, ya sea por acción del usuario o problemas con la conexión de Internet, sin permitir que todos los sub-proyectos sea compilados, es muy probable que la firma del Applet ya haya sido realizada (esto se realiza en el sub proyecto eid Applet Package); En cuyo caso, si se trata de realizar la compilación nuevamente al seleccionar el Proyecto **eID Applet Project** y luego presionar la tecla [**F11**] o hacer click derecho sobre el proyecto y elegir "**Build**" del menú contextual, es muy probable que el error que se muestra en la Figura 13, Compilación del Applet - Error aparezca, para solucionar dicho error se debe seleccionar el Proyecto eID Applet Project y luego presionar la tecla [**Shift+F11**] o hacer click derecho sobre el proyecto y elegir "**Clean and Build**" del menú contextual, NetBeans realizará el proceso de borrado de los archivos de la compilación anterior e iniciara una compilación nueva, para lo cual realizará varias descargas de las dependencias que el presente proyecto tiene (ver Figura 11, Compilación del Applet - Descarga de Dependencias) este proceso puede demorar varios minutos dependiendo de la velocidad de su conexión de Internet.

```
--- maven-jar-plugin:2.3.1:jar (default-jar) @ eid-applet-package ---
Building jar: /home/jubarran/ADRIANA/dcfd-mw-applet/applet/eid-applet-package/target/eid-applet-package-1.1.2-SNAPSHOT.jar

--- keytool-maven-plugin:1.0:genkey (default) @ eid-applet-package ---
keytool error: java.lang.Exception: Key pair not generated, alias <SIGN> already exists
java.lang.Exception: Key pair not generated, alias <SIGN> already exists
      at sun.security.tools.KeyTool.doGenKeyPair(KeyTool.java:1129)
      at sun.security.tools.KeyTool.doCommands(KeyTool.java:786)
      at sun.security.tools.KeyTool.run(KeyTool.java:172)
      at sun.security.tools.KeyTool.main(KeyTool.java:166)
------------------------------------------------------------------------
Reactor Summary:

eID Applet Project ..... SUCCESS [0.570s]
eID Applet Shared ..... SUCCESS [2.867s]
eID Applet Core ..... SUCCESS [2.556s]
eID Applet Package ..... FAILURE [1.718s]
eID Applet JavaScript ..... SKIPPED
eID Applet Service SPI ..... SKIPPED
eID Applet Service ..... SKIPPED
eID Applet Service Signer ..... SKIPPED
eID Applet Service CDI ..... SKIPPED
eID Applet Test Project ..... SKIPPED
eID Applet Test Model ..... SKIPPED
eID Applet Test Web Application ..... SKIPPED
eID Applet Test EAR ..... SKIPPED
eID Applet Test Web Application ..... SKIPPED
eID Applet SQL DDL Plugin ..... SKIPPED
eID Applet BOM ..... SKIPPED
------------------------------------------------------------------------
BUILD FAILURE
------------------------------------------------------------------------
```

Figura 13, Compilación del Applet - Error

El resultado de la compilación del código del proyecto del Applet es un conjunto de archivos JAR los cuales podrán ser utilizados para la creación de aplicaciones web.

El Applet como tal sería utilizado dentro de una página web como se muestra en el ejemplo de la Figura 14, Script para el uso del Applet.

```
<script src="https://www.java.com/js/deployJava.js"></script>
<script>
    var attributes = {
        code :'be.fedict.eid.applet.Applet.class',
        archive :'eid-applet-package.jar',
        width :400,
        height :300
    };
    var parameters = {
        TargetPage :'identity-result.jsp',
        AppletService : 'applet-service',
        BackgroundColor : '#ffffff'
    };
    var version = '1.6';
    deployJava.runApplet(attributes, parameters, version);
</script>
```

Figura 14, Script para el uso del Applet

La aplicación web en la cual se utiliza el Applet tiene que utilizar SSL para la seguridad en la comunicación entre el navegador web y el servidor de aplicaciones web. El Applet no continuara con su ejecución si detecta una sesión de navegador sin SSL.

El Applet tampoco continuará con su ejecución cuando detecte que no tiene privilegios suficientes para hacerlo. Esto implica que el archivo JAR del Applet debe estar firmado adecuadamente. Si el Applet no está firmado adecuadamente, este no podrá utilizar recursos locales de la maquina cliente.