JUNTA DE ANDALUCIA

Consejería de Hacienda y Administración Pública

Sistema de Notificaciones Telemáticas

Manual API Entidades Emisoras

Versión: v01r02

Fecha: 27/04/2018

HOJA DE CONTROL

Título	Manual API Entidades Emisoras 3.3		
Entregable	Documentación proyecto Notific@		
Nombre del Fichero	20180427-Notific@_API_Entidades_Emisoras_v3.3.1_v01r02		
Autor	DGPD		
Versión/Edición	v01r02	**Fecha Versión**	**27/04/2018**
Aprobado por		**Fecha Aprobación**	-
		Nº Total Páginas	35

REGISTRO DE CAMBIOS

Versión	Causa del Cambio	Responsable del Cambio	Área	Fecha del Cambio
v01r00	Primera versión del documento	UTE	UTE	26/02/2016
v01r01	Revisión del documento	DGPD	SCAE	27/03/2018
v01r02	Revisión del documento	UTE	UTE	27/04/2018

CONTROL DE DISTRIBUCIÓN

Nombre y Apellidos	Cargo	Área	Nº Copias
Manuel Perera Domínguez	Jefe de Servicio	SCAE	1
José Ignacio Cortés Santos	Gabinete Admón. Electrónica	SCAE	1
Antonio Heredia Rizo		UTE	1
Juan Manuel Gómez Area		UTE	1
Jesús Serrato Mata		UTE	1

ÍNDICE

Definición del Sistema de Notificaciones Telemáticas

El presente documento describe el cliente o API disponible para la interacción de las entidades emisoras con el Sistema Notific@ de Notificaciones Telemáticas de la Junta de Andalucía.

Técnicamente, se trata de un sistema de notificaciones fehacientes mediante la puesta a disposición de notificaciones electrónicas mediante una dirección electrónica habilitada.

La información completa sobre el sistema está disponible en la web de soporte técnico de administración electrónica de la Junta de Andalucía, en la siguiente dirección:

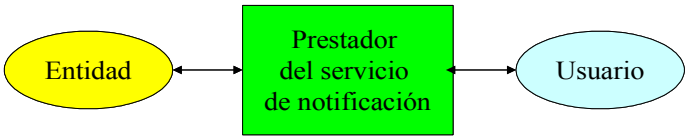
https://ws024.juntadeandalucia.es/ae/adminelec/areatecnica/notifica

El punto de acceso electrónico de la Junta de Andalucía para la práctica de la notificación electrónica es http://www.andaluciajunta.es/notificaciones

Arquitectura general del sistema

El sistema de notificación pone en relación a tres tipos de entidades que se representan en el esquema siguiente y que se designan en lo sucesivo por:

- Entidades: Consejerías, Organismos Autónomos u otras entidades públicas adscritas a la Junta de Andalucía que constituyen los clientes del servicio y que son los emisores de las notificaciones.
- Usuarios: Aquellos que son los destinatarios de las notificaciones: personas físicas, personas jurídicas y representantes de personas jurídicas.
- Prestador del servicio de notificación: La Junta de Andalucía.



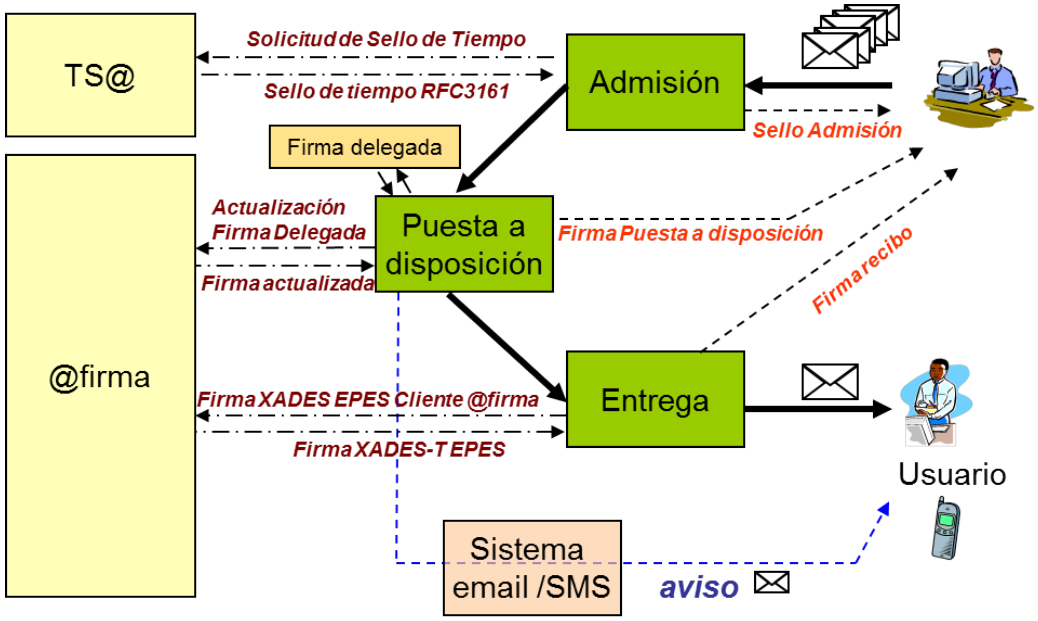
Los tratamientos asociados al envío y entrega de notificaciones pueden representarse mediante una serie de bloques que se indican a continuación:

Proceso de admisión: Entrada al sistema de lotes o remesas de notificaciones provenientes de una entidad determinada, las verificaciones y generación de mensajes de confirmación (acuse de envío).

Proceso de puesta a disposición: Tratamiento que procesa la remesa y deposita cada una de las notificaciones incluidas en ella en la dirección electrónica única del usuario destinatario. Genera acuses de "puesta a disposición" y genera avisos para los usuarios enviando un mensaje a un buzón de correo electrónico o a un teléfono móvil opcionalmente.

Proceso de entrega: El usuario puede acceder por primera vez a una notificación recibida a través de la web de abonado. En dicho proceso proceso se generará la firma de lectura de las notificaciones accedidas.

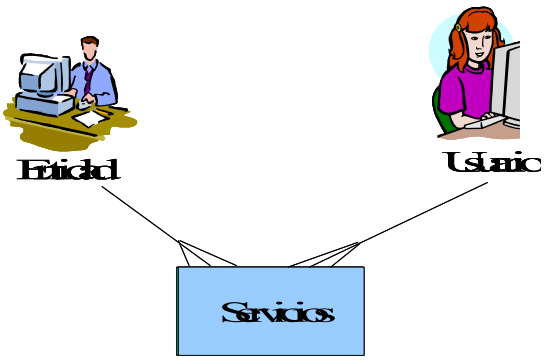
En el siguiente gráfico se muestra un esquema del proceso seguido desde que una entidad emisora envía una remesa (conjunto de notificaciones) hasta que un usuario destinatario recibe una notificación contenida en la remesa. Se observa que cada uno de los procesos se comunica con la TSA o @firma para la generación de un sello de tiempo o actualización de firmas respectivamente, de tal modo que se tenga constancia que cada proceso finalizó correctamente, en un momento dado en el tiempo y con la autorización correspondiente.



1 Servicios o suscripciones

Las notificaciones se adscriben a un "servicio". Un servicio puede ser definido como el nexo de unión entre la entidad emisora y el usuario. A través de éste, el usuario es capaz de recibir notificaciones procedentes de la Entidad Emisora. Antes de que un usuario pueda recibir estas notificaciones es necesario que éste se suscriba al servicio asociado a la Entidad Emisora por propia iniciativa a través de la web de abonado o siendo suscrito por la Entidad Emisora.

El usuario puede suscribirse a determinados servicios de entre la lista de los servicios disponibles que le presentará el sistema, y las entidades emisoras a su vez pueden enviar notificaciones correspondientes a aquellos servicios autorizados por el administrador del sistema. Por tanto, una entidad puede notificar bajo diferentes servicios, y los usuarios pueden suscribirse voluntariamente a un número indeterminado de servicios de notificación.



Funcionalidad del API de Entidades Emisoras

La Junta de Andalucía ha desarrollado un componente software (API de Entidades Emisoras) en Java que permite interactuar con el Sistema de Notificaciones Telemáticas a través del protocolo SOAP (Simple Object Access Protocol) contra un WebService. El API está compuesto por una serie de métodos que permiten realizar las siguientes funcionalidades:

- Suscripción de un usuario a un servicio asociado a la Entidad Emisora.
- Baja de un usuario de un servicio asociado a la Entidad Emisora.
- Construcción de una remesa de notificaciones y envío de la misma a un servicio asociado, previamente contratado con el prestador de servicios de notificación.
- Envío de una circular a todos los usuarios suscritos a un servicio.
- Obtención de información acerca de:
    - Los usuarios que se han dado de alta por iniciativa propia a un servicio asociado de la Entidad Emisora, entre una fecha inicial y final.
    - Los datos de contacto de un usuario concreto dado de alta en un servicio.
    - Los usuarios que se han dado de baja por iniciativa propia de un servicio asociado de la Entidad Emisora, entre una fecha inicial y final.
    - Las remesas enviadas al sistema de notificaciones. Se podrá obtener información sobre la admisión de la remesa, la puesta a disposición de las notificaciones contenidas en la remesa, la lectura o no de una notificación, errores, etc.

Todas las solicitudes que la Entidad Emisora envía al Sistema de Notificaciones a través del API son firmadas por un certificado de componente (tipo sello de entidad) que la Entidad Emisora debe solicitar y ser referenciado en la configuración del componente.

1 Descripción del desplegable

La estructura de directorios del entregable es la siguiente:

1 Directorio **lib**, donde se encuentran todas las librerías o ficheros jar necesarios para poder ejecutar la aplicación. El API de Entidades Emisoras se encuentra en la librería sntja-client-x.y.z.*jar*.

2 Directorio **configuración** con un ejemplo del fichero de configuración de los parámetros del cliente: *mcsn.properties*. Puede renombrarse y reconfigurarse.

3 Directorio **doc**, donde se encuentra el javadoc del componente y el presente documento.

4 Directorio **ejemplos**, donde se ofrece código de ejemplo en la clase *NotificaTest.java*

La librería "sntja-client-x.y.z.jar" también se encuentra disponible en el repositorio maven del Servicio de Coordinación de Administración electrónica en la siguiente ubicación:

https://ws024.juntadeandalucia.es/maven/#artifact/es.juntadeandalucia.notifica/sntja-client/

De este modo, para aplicaciones desarrolladas con maven, la librería "sntja-client" puede ser incluída añadiendo la siguiente dependencia al fichero **pom.xml** de la aplicación:

<dependency>

<groupId>es.juntadeandalucia.notifica</groupId>

<artifactId>sntja-client</artifactId>

<version>3.x.y</version>

</dependency>

Del mismo modo debe incluirse la referencia al repositorio de software donde se encuentra la librería:

<repositories>

<repository>

<id>ArtifactoryRepo</id>

<url>http://ws024.juntadeandalucia.es/maven/repository/internal</url>

</repository>

</repositories>

2 Requisitos previos

A continuación se indican los requisitos previos a tener en cuenta para el correcto funcionamiento del componente:

- Incorporar en el *classpath* del sistema o servidor de aplicaciones todas las librerías que se encuentran en el directorio ***lib***.
- En el caso de configurar el componente para una conexión HTTPS es necesario importar el certificado (clave pública del servidor Web de Notific@) en el almacén de confianza por defecto de la máquina virtual java ubicado en $JAVA_NOME/jre/lib/security/cacerts.

3 Instanciación del API

La invocación de cualquiera de los métodos ofrecidos por el API debe ir precedida por la creación de una instancia de la clase *MCSN*, ubicada en el paquete *es.juntadeandalucia.notifica.cliente.api*. Dicha instancia requiere de una configuración facilitada por la clase

es.juntadeandalucia.notifica.cliente.configuration.Configuration

Un ejemplo de instanciación sería el siguiente:

```
Configuration config = new Configuration().loadProperties( "C:/mcsn.properties" );

MCSN mcsn = new MCSN( config );
```

Como se puede observar en el código anterior la construcción de un objeto de tipo *Configuration* obliga a pasar como parámetro la ruta absoluta del archivo de configuración del módulo cliente del Sistema de Notificaciones Telemáticas. Se ofrece otro constructor del objeto *Configuration* que permite, en vez de establecer la ruta del fichero de configuración, referenciar una instancia de un objeto *java.util.Properties* previamente cargado con los parámetros de configuración. Se

recomienda el uso de este constructor cuando la configuración del componente es dinámica y se almacena por ejemplo en Base de Datos.

En el siguiente cuadro se muestra un ejemplo del fichero de configuración comentado:

```
#Características de la conexión con el sistema de notificaciones.

protocolo=https

direccion_ip=ws031.juntadeandalucia.es

puerto=443

path_acceso=jboss-net/services/ServicioWEBSN


# ------------------

# Conexión con proxy

# ------------------

# Con conexión proxy a true indica que el acceso es via proxy.

conexionproxy = false

# Nombre de servidor proxy o dirección IP:

proxyhost = nombre del HOST proxy o dirección IP

# Puerto del servidor proxy:

proxyport = 8080

# login conexión al proxy (vacio=>sin autenticación):

proxylogin =

# password conexión con proxy:

proxypassword =

# ------------------------------------------------------------------

# PKCS#12 para la firma de las solicitudes SOAP

# ------------------------------------------------------------------

# Fichero PKCS#12 que contiene la pareja de claves

pkcs12.archivo = C:/SoapCertificate.p12



# Metodo de obtención de la contraseña (clase o propiedad). La contraseña se podrá obtener

# a través del atributo pkcs12.pass o desde el método getPrivateKey() de una clase que

# implementa el interfaz IKeySec (Ver descripcion más abajo).
```

```
pkcs12.pass.metodo=clase



# Contraseña del PKCS#12 que protege la clave privada.(Metodo propiedad)

pkcs12.pass = password



# Nombre de la clase junto con el paquete que cumple el interfaz IKeySec (Ver Javadoc) y que
permite implementar el método getPrivateKey() encargado de obtener de una fuente segura la
contraseña. (Método clase)

# Esta clase deberá encontrarse en el classpath del sistema.

pkcs12.clasepwd = mi.paquete.seguridad.PassPrivateKey
```

A continuación se describen los parámetros anteriores:

Nombre	Descripción	Posibles valores
protocolo	Indica el protocolo mediante el cual el componente se comunicará con el servicio web	• http • https (Ver Nota más abajo)
dirección_ip	Host donde se encuentra el servidor del sistema de notificaciones (URL)	• Host del servidor
puerto	Indica el puerto de escucha del servidor web del Sistema de Notificaciones Telemáticas que acepta las solicitudes SOAP.	• puerto no seguro de http • puerto seguro de https (Ver Nota mas abajo)
path_acceso	Contexto donde está publicado el servicio web que atiende las peticiones	• Valor fijo: jboss-net/services/ServicioWEBSN
pkcs12.archivo	Todas las solicitudes enviadas al sistema van firmadas. Por ello es necesario que se indique la ruta absoluta del almacén PKCS#12 que contiene la clave privada.	
pkcs12.pass.metodo	Metodo de obtención de la contraseña. Ésta se puede obtener a través del atributo pkcs12.pass o desde el método getPrivateKey() de una clase que implemente el interfaz IKeySec. Ver propiedad pkcs12.clasepwd	• clase • propiedad

pkcs12.pass	Contraseña de acceso a la clave privada y del almacén PKCS#12	
pkcs12.clasepwd	Clase que cumple el interfaz IKeySec y que permite implementar el método getPrivateKey(), encargado de obtener de una fuente segura la contraseña. Para más información ver en el Javadoc la clase (es.juntadeandalucia.notifica.cliente.security.IKeySec)	

NOTA

En el caso de configurar el componente para una conexión HTTPS es necesario importar el certificado (clave pública del servidor Web de Notific@) en el almacén de confianza por defecto de la máquina virtual java ubicado en $JAVA_NOME/jre/lib/security/cacerts

4 Suscripción de un usuario a un servicio

El proceso de suscripción de un usuario a un servicio puede ser realizado de dos formas diferentes:

1. Por iniciativa del usuario desde la web de abonado del Sistema de Notificaciones Telemáticas.
2. Mediante el API de Entidades Emisoras. La Entidad Emisora puede dar de alta a un usuario en uno de los servicios que tienen asignados haciendo uso API de Entidades Emisoras.

En el segundo caso es necesario tener en cuenta lo siguiente:

- Si el usuario no está dado de alta en Notific@ en el momento de suscribirlo a un servicio, el sistema lo da de alta automáticamente.
- Se debe disponer de los códigos de servicios proporcionados a la Entidad Emisora antes de proceder al alta de usuarios en servicios.
- Los datos que debe disponer la Entidad Emisora acerca del usuario para realizar el alta en un servicio asociado son:

  - **Nombre**. Nombre del usuario.
  - **Apellidos.** En caso de tratarse de un abonado de tipo persona física, será obligatorio indicar los apellidos. En caso de tratarse de un abonado de tipo persona jurídica, no se indicará este campo.
  - **Identificador de Abonado.** El sistema identifica al usuario a través del identificador de abonado. Este identificador debe ser el NIF del abonado en caso de abonado del tipo "persona física" o "persona jurídica" y la concatenación del DNI del representante y NIF del representado cuando el tipo de abonado es "representante de persona jurídica". (*ver Anexo I*).
  - **Teléfono móvil** (OPCIONAL). En el caso de que se indique se envía un mensaje SMS a este móvil, indicando la puesta a disposición de una notificación por parte del sistema de notificaciones.
  - **Correo Electrónico.** Se envía un correo electrónico a esta dirección, indicando la puesta a disposición de una notificación por parte del sistema de notificaciones, así como los recordatorios periódicos de estar la notificación pendiente de lectura.
  - **FormReference** (por compatibilidad con la antigua firma web, puede quedar vacío)

- **TransactionID** (identificador de transacción) devuelto por el servidor de firma electrónica en la custodia de la firma generada por la aceptación de recepción de determinadas notificaciones por vía telemática, según lo dispuesto en el Real Decreto 209/2003 de 21 de febrero, por parte del usuario. En el caso de que la aplicación de la Entidad Emisora tenga su propio sistema de firma, debe suministrar el array de bytes de la firma generada por el usuario aceptando los términos de suscripción a un servicio o procedimiento. Puede ir nulo.

El teléfono y el correo electrónico pueden ser posteriormente modificados por el abonado desde la web de abonado del Sistema de Notificaciones Telemáticas.

NOTA

Si el usuario ya está dado de alta o lo estuvo anteriormente, los datos identificativos del usuario NO son modificados mediante el alta de abonado a través de la API. En este caso el usuario se volvería a dar de alta con los datos originales si estuviese dado de baja y se le suscribiría al servicio, pero NO se modificarían sus datos. Una vez creado un usuario, los datos identificativos y de contacto sólo pueden ser modificados por parte del usuario accediendo a la web de Notific@.

A continuación se muestra un ejemplo de código JAVA que permite dar de alta a un abonado en un servicio:

```
Instancia y configuración del componente MCSN

   Configuration config = new Configuration().loadProperties("C:/mcsn.properties");

   MCSN mcsn = new MCSN(config);

Creación de un Abonado destinatario de la notificación

   Abonado abonado = new Abonado("00000000T"); //Identificador de Abonado (Ver anexo I)

   abonado.setNombre("Juan");

   abonado.setApellidos("Español Español");

   abonado.setTelefonoMovil("666666666"); //Número móvil

   abonado.setEmail("juan.espanol@email.es"); //Correo electrónico

   FirmaInf firma = new FirmaInf("formreference", "TransactionID");

Envío de la solicitud de Alta de un usuario en el servicio con código 1

   mcsn.solicitarAltaAbonado(abonado, firma, 1);
```

5 Baja de un usuario de un servicio

A través de esta funcionalidad la Entidad Emisora puede dar de baja a un usuario de un servicio suscrito previamente. Para ello la Entidad Emisora debe suministrar al API un objeto de tipo Abonado donde se haya indicado el identificador del usuario.

NOTA

No es posible dar de baja a los abonados de persona jurídica, ni a los representantes de persona jurídica. Únicamente se permite la baja de personas físicas, siempre y cuando el servicio del que se vaya a dar de baja permita esta opción.

A continuación se muestra un ejemplo de código JAVA que permite dar de baja a un abonado en un servicio:

```
Instancia y configuración del componente MCSN

   Configuration config = new Configuration().loadProperties("C:/mcsn.properties");

   MCSN mcsn = new MCSN(config);

Creación de un objeto Abonado (Usuario que queremos dar de baja)

  Abonado abonado = new Abonado("00000000T"); //Identificador de Abonado (Ver anexo I)

Envío de la solicitud de Baja de un usuario en el servicio con código 1

   mcsn.solicitarBajaAbonado(abonado,1);
```

6 Envío de Remesas de notificaciones

Las notificaciones se agrupan en remesas. Una remesa es un grupo formado por una o más notificaciones.

Una entidad emisora puede construir una remesa utilizando las clases suministradas en el API de Entidades Emisoras. Una vez construida puede ser enviada al Sistema de Notificaciones Telemáticas mediante el método *enviarRemesa* de la clase MCSN.

6.1 Envío de una remesa con notificaciones incluyendo un cuerpo (OBSOLETO)

NOTA

Este método de envío de una remesa se mantiene por compatibilidad con las versiones anteriores y no debe utilizarse. Es posible que en breve Notific@ rechace las notificaciones enviadas con este método.

A continuación se muestra un ejemplo de envío de una remesa que contiene una notificación.

```
Instancia y configuración del componente MCSN

   Configuration config = new Configuration().loadProperties("C:/mcsn.properties");

   MCSN mcsn = new MCSN(config);
```

```
Creación de un Abonado destinatario de la notificación

   Abonado abonado = new Abonado("00000000T"); //Identificador de Abonado (Ver anexo I)

Lectura del fichero que se va a adjuntar a la notificación

   java.io.File adjunto = new java.io.File("C:/adjunto.pdf");  //Documentación adjunta

   if (!adjunto.exists()) {

        throw new Exception("No se ha encontrado el adjunto en " + pathAdjunto);

   }

   java.io.FileInputStream fis = new java.io.FileInputStream(adjunto);

   java.io.ByteArrayOutputStream baos = new java.io.ByteArrayOutputStream();

   byte[] buffer = new byte[1024];

   int i = 0;

   while ((i = fis.read(buffer)) != -1) {

        baos.write(buffer,0, i);

    }

    fis.close();

Creación del adjunto

    Adjunto adj1 = new Adjunto(baos.toByteArray(), adjunto.getName());

    baos.close();

Creación de la notificación

    Notificacion notif1 = new Notificacion(false); /*El parámetro indica si la
notificacion debe enviarse como una circular (true), es decir a todos los Abonados
suscritos a un servicio especifico o no (false)*/

    notif1.setAsunto("Asunto de la notificación");

    notif1.setCuerpo("Texto que forma el cuerpo de la notificación");

    notif1.addAdjunto(adj1) ; //Se puede adjuntar tantos adjuntos como se requiera

    notif1.addDestinatario(abonado); //Se pueden añadir varios destinatarios

Construcción de la remesa que contiene el adjunto creado

    Remesa remesa = new Remesa(1); /*El parámetro indica el código del servicio
remitente de la notificación*/

    remesa.addNotificacion(notif1); //Se pueden añadir varias notificaciones a la remesa
```

```
Envío de la remesa al Sistema de Notificaciones

    mcsn.enviarRemesa(remesa);
```

6.2 Envío de una remesa con notificaciones (incluyendo documento PDF y metadatos)

A continuación se muestra un ejemplo de envío de una remesa que contiene una notificación cuyo contenido debe ser un documento PDF y los metadatos que son de carácter opcional:

```
Instancia y configuración del componente MCSN

   Configuration config = new Configuration().loadProperties("C:/mcsn.properties");

   MCSN mcsn = new MCSN(config);

Creación de un Abonado destinatario de la notificación

   Abonado abonado = new Abonado("00000000T"); //Identificador de Abonado (Ver anexo I)

   abonado.setEmail("empresa.pruebas.opcional@pruebas.es"); //opcional, email
alternativo al que se enviaran los correos de aviso de la notificación

   abonado.setTelefonoMovil("555555555"); //opcional, teléfono alternativo al que se
enviaran los mensajes de aviso de la notificación

Lectura del fichero que se va a adjuntar a la notificación

   java.io.File doc = new java.io.File("C:/documento.pdf");

   if (!doc.exists()) {

        throw new Exception("No se ha encontrado el doc en " + pathAdjunto);

   }

   java.io.FileInputStream fis = new java.io.FileInputStream(doc);

   java.io.ByteArrayOutputStream baos = new java.io.ByteArrayOutputStream();

   byte[] buffer = new byte[1024];

   int i = 0;

   while ((i = fis.read(buffer)) != -1) {

        baos.write(buffer,0, i);

    }

    fis.close();
```

Creación de la notificación

```
    NotificacionDocMetadatos notif1 = new NotificacionDocMetadatos(false); /*El
parámetro indica si la notificacion debe enviarse como una circular (true), es decir a
todos los Abonados suscritos a un servicio especifico o no (false)*/

    notif1.setAsunto("Asunto de la notificación"); //Asunto de la notificación

    notif1.setCodigoRpaSia("0123"); /*Código numérico de RPA*/

    notif1.setCodigoExpediente("EXPEDIENTE-1"); /*Código del expediente asociado a la
notificación*/

    notif1.setDocNotificacion(baos.toByteArray()); //Documento de la notificación

  notif1.setEnidocFechaCaptura("23/12/2015 09:08:11");

notif1.setEnidocIdentificador( "ES_A01002823_2015_3ZS9A000000000000000000000315");

    notif1.setEnidocOrgano( "A01002823" ); //Metadato


notif1.setEnidocVersionNti( "http://administracionelectronica.gob.es/ENI/XSD/v1.0/docume
nto-e" ); //Metadato

    notif1.setEniDocOrigenCiuAdmin( "1" ); //Metadato

    notif1.setEniDocTipoDocumental( "TD11" ); //Metadato

    notif1.setEniDocValorEstElab( "EE03" ); //Metadato

    String dir3Abonado = "E00000000"; //obligatorio para abonado empleado publico.

    Boolean obligacionRelacionarseEletronicamente = false; //opcional

    Boolean notificacionEnPapel = false; //opcional

    Titular titular = new Titular("28149938P"); //opcional

    titular.setNombre("Nombre Titular");

    titular.setApellidos("Apellidos Titular");

    //Se pueden añadir varios destinatarios

    //notif1.addDestinatario(abonado); //Obsoleto

    notif1.addDestinatarioV3( abonado, dir3Abonado,
obligacionRelacionarseEletronicamente, notificacionEnPapel, titular );
```

Construcción de la remesa que contiene el adjunto creado

```
    // ID servicio y Codigo DIR3 del remitente

    RemesaDocMetadatos remesa = new RemesaDocMetadatos(1, "A01004512"); //Código DIR3
del órgano remitente de la notificación.
```

```
    remesa.addNotificacion(notif1); //Se pueden añadir varias notificaciones a la remesa
Envío de la remesa al Sistema de Notificaciones
    mcsn.enviarRemesaEni(remesa);
```

A continuación se describen los parámetros anteriores:

Nombre	Descripción	Posibles valores
Asunto	Asunto de la notificación	
Código RpaSia	RPA (Registro de Procedimientos de la Junta de Andalucía) es el código del procedimiento asociado a la notificación. Obligatorio. Este código se le suministra al gestor al registrar el procedimiento, puede comprobarse si esta registrado en el siguiente enlace: https://ciudadania.junta-andalucia.es/ciudadania/web/guest/procedimientos).	Valor numérico del código de procedimento RPA.
Código Expediente	Código del expediente de la notificación	
Documento de la notificación	Documento adjunto de la notificación	• .pdf • .xml
Fecha de captura	Fecha de inicio.	• Fecha • Fecha + Hora
Identificador	Identificador único de la notificación	Identificador ENI del documento.

Nombre	Descripción	Posibles valores
Órgano	Órgano que crea o captura el documento o el órgano responsable de la tramitación del procedimiento.	Código DIR3 del órgano
Versión NTI	Identificador normalizado de la versión de la Norma Técnica de Interoperabilidad de Documento electrónico conforme a la cual se estructura el documento electrónico.	Actualmente, valor fijo *http://administracionelectronica.gob.es/ENI/XSD/v1.0/documento-e*
OrigenCiuAdmin (Origen del document0)	Indica si el contenido del documento fue creado por el ciudadano o por una administración.	0 = Ciudadano; 1 = Administración
Tipo Documental	Modelo estructurado y reconocido que adopta un Documento, en el desarrollo de una competencia concreta, en base a una Regulación y cuyo formato, contenido informativo o soporte son homogéneos.	Valor normalizado: TDxx
Valor Estado Elaboración	Indicación del estado de la situación de elaboración de un documento, a saber, original o los distintos tipos identificados de copia.	Valor normalizado EExx
Destinatario	Abonado al que va dirigida la notificación. Pueden añadirse mas de uno. Se puede indicar de forma opcional los parámetros email y teléfono.	Objeto “abonado”
Dir3Abonado	Indica el código DIR3 del abonado, este campo solo es necesario para los abonados que son empleados públicos.	Código DIR3 del abonado.
Obligación Relacionarse Eletrónicamente	Indica si la notificación electrónica va dirigida a un destinatario obligado a relacionarse electrónicamente con las Administraciones Públicas.	true/false
Notificación en papel	Indica si la notificación electrónica también se practica en papel.	true/false
Titular	Indica el titular de la notificación. Opcional.	Objeto “titular”

6.3 Envío de una remesa con notificaciones (incluyendo documento ENI)

A continuación se muestra un ejemplo de envío de una remesa que contiene una notificación cuyo contenido es un documento ENI. El componente valida el documento ENI y extrae el documento PDF y los metadatos que se envían al servidor.

```
Instancia y configuración del componente MCSN

   Configuration config = new Configuration().loadProperties("C:/mcsn.properties");

   MCSN mcsn = new MCSN(config);

Creación de un Abonado destinatario de la notificación

   Abonado abonado = new Abonado("00000000T");  //Identificador de Abonado (Ver anexo I)

   abonado.setEmail("empresa.pruebas.opcional@pruebas.es"); //opcional, email
alternativo al que se enviaran los correos de aviso de la notificación

   abonado.setTelefonoMovil("555555555"); //opcional, teléfono alternativo al que se
enviaran los mensajes de aviso de la notificación

Lectura del fichero que se va a adjuntar a la notificación

   java.io.File doc = new java.io.File("C:/documento_eni.xml");

   if (!doc.exists()) {

        throw new Exception("No se ha encontrado el doc en " + pathAdjunto);

   }

   java.io.FileInputStream fis = new java.io.FileInputStream(doc);

   java.io.ByteArrayOutputStream baos = new java.io.ByteArrayOutputStream();

   byte[] buffer = new byte[1024];

   int i = 0;

   while ((i = fis.read(buffer)) != -1) {

        baos.write(buffer,0, i);

    }

    fis.close();

Creación de la notificación

    NotificacionEniDoc notif1 = new NotificacionEniDoc(false); /*El parámetro indica si
la notificacion debe enviarse como una circular (true), es decir a todos los Abonados
suscritos a un servicio especifico o no (false)*/

    notif1.setAsunto("Asunto de la notificación"); //Asunto de la notificación
```

```
    notif1.setCodigoRpaSia("0123"); // Código numérico RPA

    notif1.setCodigoExpediente("EXPEDIENTE-1"); // Código Expediente

    notif1.setDocEni(baos.toByteArray()); // Documento ENI de la notificación

    String dir3Abonado = "E00000000"; //obligatorio para abonado empleado publico.

    Boolean obligacionRelacionarseEletronicamente = false; //opcional

    Boolean notificacionEnPapel = false; //opcional

    Titular titular = new Titular("28149938P"); //opcional

    titular.setNombre("Nombre Titular");

    titular.setApellidos("Apellidos Titular");

    //Se pueden añadir varios destinatarios

    //notif1.addDestinatario(abonado); //Obsoleto

    notif1.addDestinatarioV3( abonado, dir3Abonado,
obligacionRelacionarseEletronicamente, notificacionEnPapel, titular );

Construcción de la remesa que contiene el adjunto creado

    // ID servicio y Codigo DIR3 del remitente

    RemesaEniDoc remesa = new RemesaEniDoc(1, "A01004512"); //Código DIR3 del órgano
remitente de la notificación.

    remesa.addNotificacion(notif1); //Se pueden añadir varias notificaciones a la remesa

Envío de la remesa al Sistema de Notificaciones

    mcsn.enviarRemesaEni(remesa);
```

NOTA

- El código DIR3 utilizado debe de estar asignado al servicio indicado
- En el caso de no estar asignado, se asignará automáticamente tras realizar la operación, siempre y cuando este código DIR3 pertenezca a la jerarquía de códigos DIR3 de la entidad emisora que envía la notificación.

El formato del documento ENI debe ser el siguiente:

```
<?xml version="1.0" encoding="UTF-8"?>
<enidoc:documento
    xmlns:enidocmeta="http://administracionelectronica.gob.es/ENI/XSD/v1.0/documento-e/metadatos"
    xmlns:enidoc="http://administracionelectronica.gob.es/ENI/XSD/v1.0/documento-e"
    xmlns:enifile="http://administracionelectronica.gob.es/ENI/XSD/v1.0/documento-e/contenido"
    xmlns:enids="http://administracionelectronica.gob.es/ENI/XSD/v1.0/firma"
```

```
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://administracionelectronica.gob.es/ENI/XSD/v1.0/documento-
e/documentoEni.xsd">
    <enifile:contenido Id="ID_01">
        <enifile:ValorBinario>[DOCUMENTO_PDF]</enifile:ValorBinario>
        <enifile:NombreFormato>PDF</enifile:NombreFormato>
    </enifile:contenido>
    <enidocmeta:metadatos>
        <enidocmeta:VersionNTI>http://administracionelectronica.gob.es/ENI/XSD/v1.0/documento-
e</enidocmeta:VersionNTI>
        <enidocmeta:Identificador>[IDENTIFICADOR]</enidocmeta:Identificador>
        <enidocmeta:Organo>[ORGANO]</enidocmeta:Organo>
        <enidocmeta:FechaCaptura>[FECHA]</enidocmeta:FechaCaptura>
        <enidocmeta:OrigenCiudadanoAdministracion>true</enidocmeta:OrigenCiudadanoAdministracion>
        <enidocmeta:EstadoElaboracion>

<enidocmeta:ValorEstadoElaboracion>[ESTADO_ELABORACION]</enidocmeta:ValorEstadoElaboracion>
        </enidocmeta:EstadoElaboracion>
        <enidocmeta:TipoDocumental>[TIPO_DOCUMENTAL]</enidocmeta:TipoDocumental>
    </enidocmeta:metadatos>
    <enids:firmas>
        <enids:firma>
            <enids:TipoFirma>[TIPO_FIRMA]</enids:TipoFirma>
            <enids:ContenidoFirma>
                <enids:FirmaConCertificado>
                    <enids:FirmaBase64>[FIRMA_BASE64]</enids:FirmaBase64>
                </enids:FirmaConCertificado>
            </enids:ContenidoFirma>
        </enids:firma>
    </enids:firmas>
</enidoc:documento>
```

NOTA

- Es posible enviar una notificación a todos los usuarios suscritos a un servicio, es decir una ***circular***, indicando ***true*** en el constructor de la notificación.

7 Sistema de información para entidades

Este subsistema agrupa los procesos destinados a extraer y servir informaciones del Sistema de Notificaciones Telemáticas a las entidades que lo soliciten.

La información disponible para las entidades es la siguiente:

- Usuarios suscritos por iniciativa propia, entre una fecha inicial y final, a servicios de notificaciones contratados por la entidad.
- Usuarios dados de baja por iniciativa propia, entre una fecha inicial y final, a servicios de notificaciones contratados por la entidad.
- Acuses de admisión. Mediante estos acuses se informará tanto de las remesas admitidas por el sistema de notificación como de los posibles errores detectados en la admisión.
- Firma de puesta a disposición de los mensajes. Asimismo se generarán errores de puesta a disposición de aquellos mensajes que no se puedan depositar en el buzón del usuario, indicando el error detectado (usuario desconocido, usuario no suscrito al servicio, etc).

- Acuses de recibo de los mensajes de aquellos servicios que lo requieran.
- Estado de un conjunto de usuarios en un servicio asociado.
- Certificado de un conjunto de usuarios, obtenido por el sistema en el momento de autenticación del usuario en el Sistema de Notificaciones.
- Número y fecha de registro de salida en el registro @ries de la notificación.
- Información de un abonado suscrito a un servicio.

7.1 Información acerca de los usuarios dados de alta en un servicio

El Sistema de Notificaciones permite a un usuario previamente dado de alta en el sistema la suscripción a un servicio. La entidad emisora puede identificar nuevos abonados que se han dado de alta en un servicio asociado mediante el siguiente método del API:

AbonadoInf[] solicitarInformacionAltasIniciativaAbonado(Date fechaIni, Date fechaFin,int cod_Servicio)

El método anterior requiere la inclusión del código de servicio (suministrado por el Administrador del Sistema de Notificaciones) del cual se quieren obtener los Abonados dados de alta por iniciativa propia. Además, es necesario que indique el rango de fechas en los que los abonados se dieron de alta en el servicio indicado.

Como resultado de la invocación del método se obtiene información de los usuarios dados alta por propia iniciativa. Dicha información está compuesta por:

- Nombre del usuario
- Apellidos del usuario
- Identificador del usuario en el sistema

7.2 Información acerca de los usuarios dados de baja de un servicio

El Sistema de Notificaciones permite a un usuario previamente dado de alta en el sistema darse de baja de un servicio. La entidad emisora puede identificar los abonados que se han dado de baja de un servicio asociado mediante el siguiente método del API:

AbonadoInf[] solicitarInformacionBajasAbonado(Date fechaIni, Date fechaFin,int cod_Servicio)

El método anterior requiere la inclusión del código de servicio (suministrado por el Administrador del Sistema de Notificaciones) del cual se quieren obtener los Abonados dados de baja por iniciativa propia. Además es necesario que se indique el rango de fechas en los que los abonados se dieron de baja del servicio indicado.

Como resultado de la invocación del método se obtiene la siguiente información de los usuarios dados baja por iniciativa propia:

- Nombre del usuario
- Apellidos del usuario
- Teléfono móvil (Opcional)
- Correo electrónico (Opcional)

7.3 Información acerca del estado de un conjunto de abonados en un servicio

Las Entidades Emisoras pueden en todo momento conocer si un conjunto de usuarios están suscritos a un servicio asociado a la entidad, gracias a la invocación del método siguiente:

public int[] solicitarEstadoAbonadoServicio(String[] anagramasAbons, int cod_Servicio)

En la invocación del anterior método es necesario que la Entidad Emisora indique los siguientes parámetros:

- AnagramasAbons: Es el conjunto de identificadores unívocos de los usuarios que queremos conocer su estado en un servicio (compuestos por DNI o NIE + CIF).
- Código de servicio, suministrado por el Administrador del Sistema de Notificaciones, del cual se quiere comprobar si el usuario está suscrito.

Como resultado de la invocación del método se obtendrá un *array* de tipo entero por cada identificador consultado. Dependiendo de su valor indicará:

- **-1**, El usuario **no** está dado de alta en el sistema
- **0**, El usuario **no** está suscrito al servicio indicado.
- **1**, El usuario **está suscrito** al servicio indicado

A continuación se muestra un ejemplo que permite la invocación de esta funcionalidad:

```
Instancia y configuración del componente MCSN

   Configuration config = new Configuration().loadProperties("C:/mcsn.properties");

   MCSN mcsn = new MCSN(config);

Envío de la solicitud de estado de un abonado en el servicio con código 1

  String[] ids = new String[1];

   Ids[0]= "00000000T"; //Identificador del abonado (ver Anexo I)

   Int[] res = mcsn.solicitarEstadoAbonadosServicio(ids, 1);

Se procesa el resultado

   if (res[0] == -1) {

          logger.info("El usuario NO está dado de alta en el sistema");

   }

  else if (res[0] == 0) {

          logger.info("El usuario NO está suscrito en el SERVICIO " + 1 + ", pero si
esta dado de alta en el sistema");

   }
```

```
else if (res[0] == 1) {

    logger.info("El usuario SI está suscrito al SERVICIO " + 1);

}
```

7.4 Información de contacto de un abonado suscrito a un servicio

Las entidades emisoras pueden en todo momento obtener la información de contacto de un abonado suscrito a un servicio que está asociado a dicha entidad emisora, gracias a la invocación del método siguiente:

public Abonado obtenerInfAbonadoSuscrito (String identificadorAbonado, int codServicio)

En la invocación del anterior método es necesario que la Entidad Emisora indique los siguientes parámetros:

- Identificador de Abonado: Es el identificador unívoco del usuario del que se desean obtener sus datos (compuesto por DNI o NIE + CIF).
- Código de servicio, suministrado por el Administrador del Sistema de Notificaciones, es necesario para comprobar si el usuario está suscrito a dicho servicio.

La operación devuelve un objeto de tipo **es.juntadeandalucia.notifica.cliente.estructuras.Abonado** que contiene los siguientes métodos para consultar los datos del abonado solicitado:

- getIdAbonado()
- getNombre()
- getApellidos()
- getTelefonoMovil()
- getEmail()
- getDireccion()
- getLocalidad()
- getProvincia()
- getZip()

La operación de solicitud de información de contacto de un abonado debe realizarse previa comprobación de la suscripción de éste en el servicio en cuestión mediante la operación *solicitarEstadoAbonadoServicio*. En caso de que se realice la invocación de la operación *obtenerInfAbonadoSuscrito* y el abonado no estuviese suscrito al servicio enviado como parámetro, la operación elevará una excepción con un mensaje descriptivo de la imposibilidad de realizar la operació

7.5 Información de datos de un titular

Las entidades emisoras pueden en todo momento consultar los datos de un titular, gracias a la invocación del método siguiente:

public Titular obtenerInfTitular(String nifTitular);

En la invocación del anterior método es necesario que la Entidad Emisora indique el siguiente parámetro:

- NIF Titular: Es el identificador unívoco del usuario del que se desean obtener sus datos (DNI o NIF del titular).

La operación devuelve un objeto de tipo **es.juntadeandalucia.notifica.cliente.estructuras.Titular** que contiene los siguientes métodos para consultar los datos del titular solicitado:

- getNif()
- getNombre()
- getApellidos()

La operación de solicitud de información de titular debe realizarse si en el proceso de envío de una notificación se ha indicado de forma opcional el titular. Si un titular se da de alta en Notific@ como abonado, se procederá al alta, completando los datos que faltan del titular y se modificando su tipo.

7.6 Información de remesas enviadas

El API permite a la Entidad Emisora obtener información acerca de las remesas enviadas y las notificaciones contenidas en las mismas. Esta información puede ser obtenida a través de un conjunto de funciones filtro que se indican a continuación:

a) Información acerca de las remesas enviadas a partir de sus códigos devueltos en el envío de remesa. El parámetro *conAcuses* indica si se desea obtener los acuses generados, por el sistema y por el abonado en la lectura de la notificación, en el *array* de *RemesaInf*.

```
RemesaInf[] obtenerInfRemesas(int[] remesas, bolean conAcuses)
```

b) Información ampliada acerca de las remesas enviadas a partir de sus códigos devueltos en el envío de remesa. **Incluye para cada una de las notificaciones de cada una de las remesas el número y la fecha del registro de salida en el registro @ries.** Las clases que almacenan los datos devueltos por esta operación se denominan RemesaInfCompleta y NotificacionInfCompleta que heredan a RemesaInf y NotificacionInf respectivamente.

```
RemesaInfCompleta[]obtenerInfCompletaRemesas (int[] remesas)
```

c) Información acerca de las remesas asociadas a un servicio y procesadas entre una fecha inicial y final, que contengan notificaciones leídas por el usuario.

```
RemesaInf[] obtenerInfRemesaConNotifLeidas(Date fechaIni, Date fechaFin, int
                                cod_Suscripcion)
```

d) Información acerca de las remesas asociadas a un servicio y procesadas entre una fecha inicial y final, que contienen notificaciones no leídas por el usuario. El parámetro *cumplidoPlazo* es un modificador del método que permite, si es *true,* obtener información acerca de las remesas y notificaciones que no han sido leídas en el plazo máximo exigible de 10 días en el Real Decreto 209/2003 desde la recepción de la notificación, siendo notificada o avisada por correo electrónico o por el envío de un SMS dicha recepción. En el caso de que su valor sea *false,* se obtienen todas las notificaciones no leídas, tanto rechazadas como no.

```
RemesaInf[] obtenerInfRemesaConNotifNoLeidas(Date fechaIni, Date fechaFin, int
                    cod_Suscripcion, boolean cumplidoPlazo)
```

e) Obtención de información acerca de las remesas entregadas al Sistema de Notificaciones Telemáticas y procesadas por el mismo entre una fecha inicial y final, y asociadas a una suscripción concreta de la entidad.

```
RemesaInf[] obtenerInfRemesa(Date fechaIni, Date fechaFin, int cod_Suscripcion)
```

f) Información acerca de las remesas entregadas al Sistema de Notificaciones Telemáticas y procesadas por el mismo entre una fecha inicial y final, asociadas a una suscripción concreta de la entidad, y que contienen notificaciones destinadas a un grupo de abonados.

```
RemesaInf[] obtenerInfRemesa(Date fechaIni, Date fechaFin, int cod_Servicio,
                                Abonado[] abonados)
```

NOTA

Sólo con los métodos *obtenerInfRemesas(int[], boolean conAcuses)* y *obtenerInfCompletaRemesas(int[])* es posible obtener los acuses generados por el sistema.

Integración de Notific@ en una aplicación de un Organismo

La integración de una aplicación con Notific@ a través del API de Entidades Emisoras requiere de un pequeño desarrollo de integración. Para llevar a cabo dicha integración es necesario tener presente los puntos tratados a continuación.

1 Suscripción de usuarios a servicios

La plataforma permite suscribir un usuario a un servicio asociado a una Entidad Emisora de formas distintas:

a. **Usuario suscrito mediante API**. En este caso es necesario disponer de los siguiente datos del usuario:

   i. Identificador, que permite determinar al usuario de forma unívoca. Esta cadena se obtiene concatenando el DNI o NIE del abonado con el CIF en caso de usuarios que deseen acceder mediante su certificado de representante de persona jurídica, para abonados que accedan mediante certificado de persona física el identificador debe estar compuesto únicamente por el DNI o NIE del usuario. Para abonados que accedan mediante certificado de persona jurídica el identificador debe estar compuesto únicamente por el NIF de la empresa (ver Anexo I). **Es muy importante verificar que el abonado que se va a dar de alta está en posesión de un certificado digital reconocido por la plataforma @firma. Así mismo, se debe generar un identificador acorde con el tipo de certificado del que va a hacer uso el usuario en la plataforma: persona física o jurídica.** El identificador que gestionaba anteriormente la plataforma continua siendo compatible en esta versión de Notific@.

   ii. Nombre

   iii. Apellidos (en caso de tratarse de persona física)

   iv. Correo Electrónico

   v. Opcionalmente SMS, si se desea que el sistema automáticamente envíe un mensaje de texto informando de la llegada de una notificación.

> **NOTA**
>
> Para la suscripción de un usuario mediante el API, es necesario que el usuario firme un formulario dando su consentimiento expreso de que desea que le notifiquen vía telemática todas las notificaciones generadas por el procedimiento en cuestión.

b. **Usuario suscrito por iniciativa propia**. Se debe invocar el método *AbonadoInf[] solicitarInformacionAltasAbonado(Date fechaIni, Date fechaFin,int cod_Servicio)* para obtener los abonados dados alta en un servico entre dos fechas. Este método devolverá información (Nombre, apellidos, identificador) de todos los usuarios suscritos al servicio indicado.

Es responsabilidad de la aplicación cliente comprobar si los usuarios a los que se les va a remitir las notificaciones están suscritos al servicio asociado, mediante los métodos que proporciona la API. Si el abonado no se encuentra dado de alta en el sistema o no está suscrito al servicio, se deberá dar de alta y/o suscribirlo al servicio antes de enviarle la notificación.

2 Envío de Remesa de Notificaciones

Una vez comprobada la existencia de un abonado y su suscripción al servicio requerido, la aplicación cliente puede enviar a estos abonados notificaciones mediante la formación de una remesa.

> **NOTA**
>
> Antes de enviar una remesa con notificaciones la aplicación debe comprobar que los destinatarios están suscritos al servicio que envía la remesa. Para ello se debe invocar el método *int[] solicitarEstadoAbonadosServicio(String[] anagramasAbons, int cod_Servicio).* Sólo deben enviarse notificaciones a aquellos destinatarios en los que el método devuelva '1'. Para aquéllos que devuelva '0' o '-1' será necesario darles de alta antes de enviarles una notificación.

Es responsabilidad de la aplicación almacenar en base de datos el código devuelto por el sistema en el envío de la remesa, ya que este código permite verificar posteriormente si la remesa se ha procesado correctamente y que las notificaciones contenidas en la misma, han sido leídas o rechazadas por los usuarios o rechazadas por el sistema transcurridos el plazo estipulado desde la puesta a disposición y la no lectura por el destinatario.

Es conveniente registrar las notificaciones que se envían para poder consultar cualquier evento que se produzca sobre la misma, ya sea la puesta a disposición, su lectura o rechazo. Se recomienda identificar la notificación en el origen, utilizando el método *setId(int)* de la clase *Notificacion*, para mantener la trazabilidad de las notificaciones.

3 Obtener información de remesas enviadas

La obtención de información de las remesas enviadas permite verificar que el procesamiento de las remesas ha sido satisfactorio. El procesamiento de una remesa se realiza de forma asíncrona, por lo que no es conveniente verificar una remesa nada más enviarla.

3.1 Tiempo para verificar el procesamiento de una remesa

Se recomienda verificar una remesa enviada a partir de las **cinco horas** desde que se envió, permitiendo de esta manera dejar suficiente tiempo al sistema para procesarla adecuadamente. La forma de verificar su procesamiento es invocar al método *RemesaInf[] obtenerInfRemesas(int[] remesas, bolean conAcuses).*

3.2 Estados de las notificaciones

Cada objeto de tipo *RemesaInf* obtenido contiene un array de *NotificacionInf.* Los objetos *NotificacionInf* tienen un atributo estado accesible mediante el método *getEstado(). Dicho atributo* puede devolver cualquiera de las constantes definidas en la clase *es.juntadeandalucia.notifica.common.util.EstadoNotificacion. L*os estados que puede devolver se muestran en la siguiente tabla:

Estado	Descripción
0	Notificación puesta a disposición del abonado.
5	Notificación puesta a disposición y leída por el usuario.
6	Notificación puesta a disposición y rechazada por el usuario.
7	Notificación puesta a disposición y rechazada por el sistema automáticamente, por no lectura de la notificación por el usuario durante los diez días desde su puesta a disposición.
-1	Se ha enviado una notificación a un destinatario no dado de alta en el sistema.
-2	Se ha enviado una notificación a un destinatario no suscrito al servicio destino de la remesa.
-3	La notificación no se pudo poner a disposición por un problema técnico

3.3 Tiempo para verificar la lectura o rechazo de una notificación por el destinatario

La verificación de la lectura o rechazo de la notificación contenida en la remesa debe realizarse mediante el método *obtenerInfRemesas* descrito. Una vez transcurrido el tiempo establecido para la lectura de notificaciones (valor establecido por cada servicio en Notifica, con un valor por defecto de **10 días naturales**) desde el envío de la remesa, el sistema rechaza automáticamente aquellas notificaciones que no hayan sido leídas o rechazadas por el usuario. Por tanto, los posibles estados de las notificaciones pueden ser:

- Notificación leída por el usuario.
- Notificación rechazada por el usuario.
- Notificación rechazada por el sistema (caducada).

ANEXO I: Identificación del abonado

Se debe tener siempre en cuenta que el Sistema de Notificaciones Telemáticas distingue entre el alta de abonados que acceden mediante certificado de persona física y abonados que acceden mediante certificado de persona jurídica, considerando a ambos abonados distintos aun siendo la misma persona. Es decir, un abonado dado de alta para acceder mediante su certificado de persona física no puede ver las notificaciones que se hayan enviado a la misma persona a través de su identificador de persona jurídica (DNI + CIF).

a) Anagrama fiscal (OBSOLETO)

En versiones anteriores a Notific@ 2.2, la identificación de los abonados en las llamadas al API de Entidades Emisoras del Sistema de Notificaciones se realiza mediante la inclusión del anagrama fiscal del abonado, compuesto por:

DNI/NIE + 4 primeros caracteres del primer apellido

+ 3 primeros caracteres del segundo apellido + primer carácter del nombre + NIF en su caso

Este modo de identificación era utilizado por @firma4 y fue eliminado en @firma5.

b) Identificador del abonado

A partir de la versión 2.2 de Notific@, se mantiene la compatibilidad con el sistema de identificación descrito (anagrama fiscal), sin embargo, la plataforma permite utilizar un nuevo modelo de identificador compuesto por:

DNI/NIE para personas físicas

DNI/NIE+NIF para representantes de personas jurídicas

NIF para personas jurídicas

Este modelo no da lugar a equívocos, simplifica la identificación de los abonados y se ha establecido para identificar a los nuevos abonados dados de alta. Dependiendo del modelo que adopte el integrador (anagrama fiscal o identificador) para la identificación de los abonados, el sistema realiza diversas acciones para obtener el abonado requerido.

Este proceso es transparente para el integrador y se detalla a continuación:

1) Abonado registrado con certificado de persona física

- **Identificación mediante Anagrama:** Igual que en la versión 2.0.

- **Identificación mediante DNI/NIE:** El sistema busca la existencia de los siguientes patrones para la identificación:

- ***DNI/NIE (identificador persona física)***
- ***DNI/NIE + 8 letras (anagrama fiscal persona física)***

2) Abonado registrado con certificado de representación persona jurídica

- **Identificación mediante Anagrama:** Igual que en la versión 2.0.

- **Identificación mediante DNI/NIE+NIF:** El sistema buscará la existencia de los siguientes patrones para la identificación:

- ***DNI/NIE + NIF (identificador persona jurídica)***
- ***DNI/NIE + 8 letras + NIF (anagrama fiscal persona jurídica)***
- ***NIF (de la entidad representada)***

3) Abonado registrado con certificado de persona jurídica

- **Identificación mediante NIF:** El sistema busca la existencia de los siguientes patrones para la identificación:

  - ***NIF (identificador persona jurídica)***

Métodos del API afectados

Los métodos del API de Entidades Emisoras del Sistema de Notificaciones a los que se pasa como parámetro el anagrama fiscal o el nuevo identificador de abonado son los siguientes:

1. ***solicitarAltaAbonado(Abonado abonado, FirmaInf firma, int cod_servicio)***
2. ***solicitarBajaAbonado(Abonado abonado, int cod_servicio)***

3 int idRemesa enviarRemesa (Remesa remesa)

4 AbonadoInf[] solicitarInformacionAltasAbonado(Date fechaIni, Date fechaFin,int cod_Servicio)

5 ***Abonado obtenerInfAbonadoSuscrito (String identificadorAbonado, int codServicio)***

6 AbonadoInf[] solicitarInformacionBajasAbonado(Date fechaIni, Date fechaFin,int cod_Servicio)

7 int[] solicitarEstadoAbonadoServicio(String[] anagramasAbons, int cod_Servicio)

8 CertificadoInf[] solicitarCertificadoAbonados(String[] anagramasAbons)

Los métodos *setIdAbonado(String id)* y *getIdAbonado()* de la clase *Abonado* son los encargados de modificar o consultar respectivamente el anagrama fiscal o el nuevo identificador del abonado.

IMPORTANTE

Es **OBLIGATORIO** el uso del nuevo identificador en el alta de nuevos abonados en el sistema (*solicitarAltaAbonado*). Si se intenta dar de alta con el anagrama fiscal, el sistema lo registrará con el nuevo identificador.

Resumen

La siguiente tabla muestra un resumen de los expuesto anteriormente respecto a la identificación de los usuarios en el Sistema de Notificaciones Telemáticas.

Método invocado	Entrada	Comprobación de registro	Registro en el sistema si procede
solicitarAltaAbonado	**12345678Z**	**12345678Z** o 12345678ZXXXXXXXX (anagrama fiscal persona física)	**12345678Z**
solicitarAltaAbonado	12345678ZROMAMARA	**12345678Z** o 12345678ZROMAMARA	**12345678Z**
solicitarAltaAbonado	**12345678ZA99999999**	**12345678ZA99999999** o 12345678ZXXXXXXXXA99999999 (anagrama fiscal persona jurídica)	**12345678ZA99999999**
solicitarAltaAbonado	12345678ZROMAMARAA99999999	**12345678ZA99999999** 0	**12345678ZA99999999**

		12345678ZROMAMARAA99999999 (anagrama fiscal persona jurídica)	
Resto de métodos	Mismas entradas	Misma comprobación	No procede