Diseño técnico de Procesos batch

de Notificaciones Postales con seguimiento electrónico

FECHA: <01/02/2019>

EJIE, S.A.

Avda. Mediterráneo, 14

01010 - Vitoria-Gasteiz

EJIE, S.A. 2004

Este documento es propiedad de EJIE, S.A. y su contenido es confidencial. Este documento no puede ser reproducido, en su totalidad o parcialmente, ni mostrado a otros, ni utilizado para otros propósitos que los que han originado su entrega, sin el previo permiso escrito de EJIE, S.A.. En el caso de ser entregado en virtud de un contrato, su utilización estará limitada a lo expresamente autorizado en dicho contrato. EJIE, S.A. no podrá ser considerada responsable de eventuales errores u omisiones en la edición del documento.

Contenido

Capítulo/sección Página

PROCESO CONTROL DE ENVÍOS 3

PROCESA ENVÍO DIARO 6

PROCESA IMÁGENES PENDIENTES 12

PROCESA REMESA RESULTADOS 14

PREPARAR REMESAS 16

PROCESO PIF 17

PROCESO CALCULO DE INCUMPLIMIENTOS EN ENTREGA DE NOTIFICACION 23

# PROCESO CONTROL DE ENVÍOS

Trata remesas completas, donde colecciones o notificaciones no entregadas en Correspondencia en el plazo de 7 días desde la fecha de envio de la remesa a Correos. Entregas vencidas y envíos con la marca de “no entregar a Correos.Manda emails a Correos con la información de código de barras no entregado.

FUNCIONAMIENTO:

Por cada cliente:

1.- Se buscan las remesas enviadas y con las colecciones completadas (entregadas a Correos) en los últimos 7 días

*List<Remesa> remesasCompletas = remesaDao.remesasCompletasEnLosUltimos7Dias(cliente.getIcliente()*

SELECT remesacorr,codcliente FROM remesa

INNER JOIN clientes ON fkcliente =icliente

WHERE feenvio >=SYSDATE -7 and remesa.fkcliente=?

AND tiporemesa = 'D'

AND REMESA.FKESTADOREM = 'E'

AND idremesa NOT IN

(SELECT DISTINCT fkremesa

FROM coleccion

WHERE fkestadocol !='J'

and fkestadocol !='S'

and (fkestadocol ='A'

OR (fkestadocol = 'P' AND IDCOLECC NOT IN

(SELECT distinct(idcolecc) FROM HTO\_ENTREGAS,ENVIO,COLECCION WHERE envio.fkcolecc = coleccion.idcolecc AND ENVIO.IDENVIO=hto\_entregas.idenvio) )) )

2.- Se buscan las colecciones no entregadas a Correos, de una remesa enviada hace 7 días y los envíos que el usuario ha decidido no entregar, de su colección entregada.

*List<Envio> enviosSinEntregar=this.envioDao.enviosFueraDePlazo(cliente.getIcliente());*

SELECT remesa.remesacorr,clientes.codcliente, envio.destnomb,envio.destnomb2, envio.CODBARRAS

FROM envio

INNER JOIN coleccion

ON envio.fkcolecc=coleccion.idcolecc

INNER JOIN remesa

ON coleccion.fkremesa=remesa.idremesa

INNER JOIN clientes

ON fkcliente=icliente

WHERE envio.fkremesa IN (SELECT idremesa FROM remesa WHERE remesa.feenvio <SYSDATE - 7 AND tiporemesa = 'D')

AND fkestadocol <> ‘J’ AND fkestadocol <> ‘S’

AND ( FKCOLECC NOT IN ( SELECT distinct(FKCOLECC) FROM ENVIO , HTO\_ENTREGAS

WHERE ENVIO.IDENVIO=HTO\_ENTREGAS.IDENVIO) )

and remesa.fkcliente=?

UNION

SELECT remesa.remesacorr,clientes.codcliente,

envio.destnomb,envio.destnomb2, envio.CODBARRAS

from clientes clientes,

remesa remesa,

coleccion coleccion,

envio envio

WHERE clientes.icliente = ? and

remesa.fkcliente = clientes.icliente and

remesa.feenvio <SYSDATE - 7 and

coleccion.fkremesa = remesa.idremesa and

envio.fkcolecc = coleccion.idcolecc

and envio.noentrega = '1'

and envio.fksitentreg is null

order by codcliente,remesacorr,codbarras

3.- Se pone estado a la colección a S=Sin entregar a Correos a las colecciones no enviadas en esos 7 días

*coleccionDao.marcarNoEntregadas(cliente.getIcliente())*

UPDATE (SELECT fkestadocol

FROM coleccion

INNER JOIN remesa

ON coleccion.fkremesa=remesa.idremesa and remesa.feenvio <SYSDATE - 7 AND tiporemesa = 'D' and REMESA.FKESTADOREM='E'

INNER JOIN clientes

ON fkcliente=icliente

WHERE remesa.fkcliente=?");

AND (fkestadocol = 'A' OR (fkestadocol ='P'

AND IDCOLECC NOT IN (SELECT distinct(idcolecc) FROM HTO\_ENTREGAS,ENVIO,COLECCION

WHERE envio.fkcolecc = coleccion.idcolecc AND ENVIO.IDENVIO=hto\_entregas.idenvio )))

and idcolecc in (select fkcolecc from envio)

) set fkestadocol=’S’

4.- Se pone a los envíos que tienen marca de “no entregar a Correos” la situación de entrega 34

*coleccionDao.marcarSitEnvioaNoEntregadas(cliente.getIcliente())*

UPDATE (SELECT FKSITENTREG

from clientes clientes,

remesa remesa,

coleccion coleccion,

envio envio

WHERE clientes.icliente = ? and

remesa.fkcliente = clientes.icliente and

remesa.feenvio <SYSDATE - 7 and

coleccion.fkremesa = remesa.idremesa and

envio.fkcolecc = coleccion.idcolecc

and envio.noentrega = '1'

and envio.fksitentreg is null

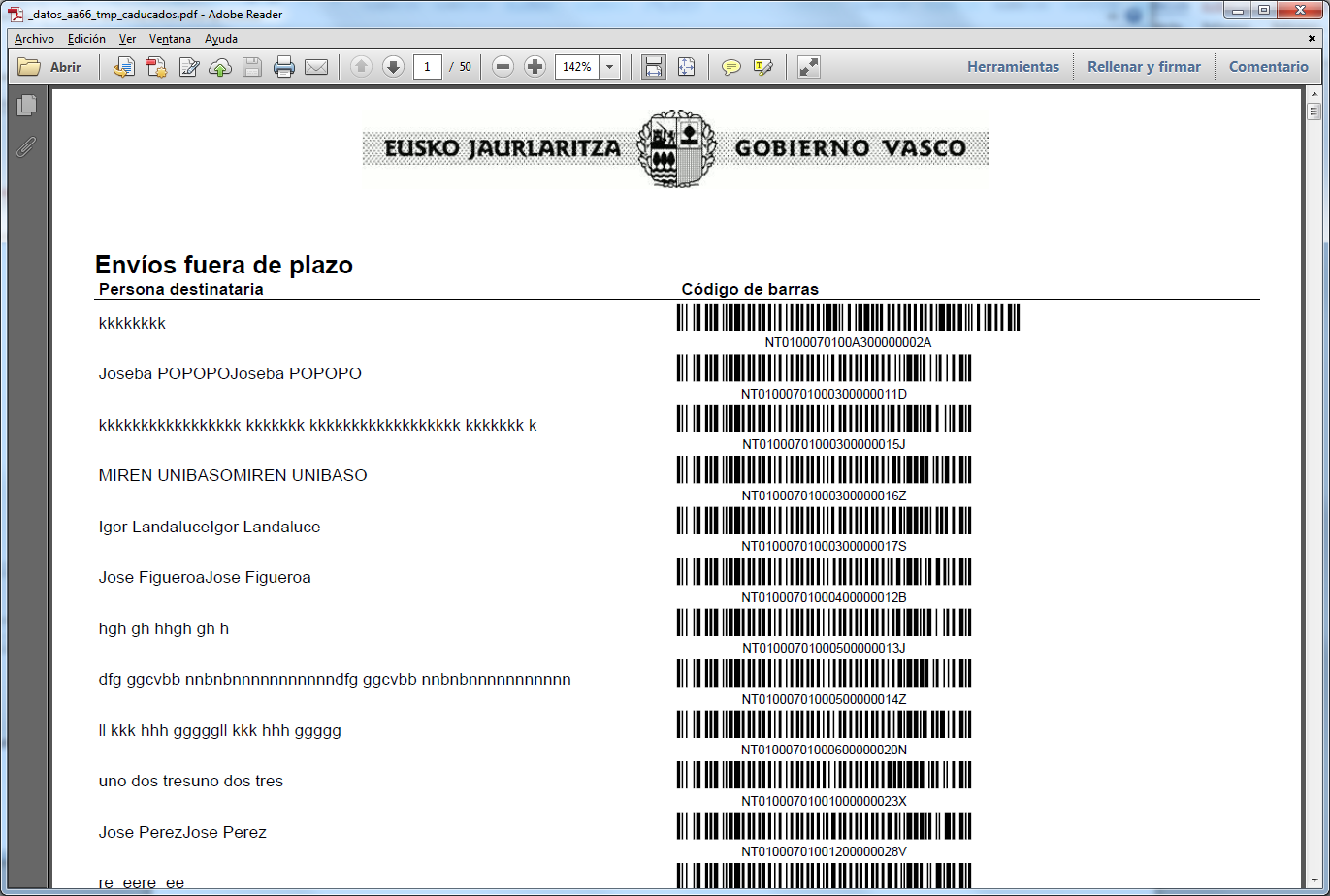
) SET FKSITENTREG = 34 🡪 valor en la tabla de situaciones “no entrega”

5.- Si hay remesas completas (punto 1) o envíos sin Entregar (punto 2) se crea el listado y se manda por email a Correos, separados por cliente de Correos – mail concreto.

|  |  |
| --- | --- |
| CLIENTE | EMAIL |
| LAKUA | ccpc\_vitoria.alava@correos.com |
| GIPUZKOA | ccpc\_sansebastian.guipuzcoa@correos.com |
| BIZKAIA | ccpc\_bilbao.vizcaya@correos.com |

Asunto del email: “Listados de remesas finalizadas y notificaciones que no se enviarán”

Con fichero adjunto:

Santander

# PROCESA ENVÍO DIARO

Recibe los ficheros de situaciones de entrega por parte de Correos y carga las pruebas de entrega de aquellas situaciones que lleguen con ella.

Inicialmente, coge los ficheros que comiencen por “NT” y que no finalicen por “-resultado.txt”,”BAD”,”bad”,”pdf” y “xml”. Eso son los ficheros de envío diario.

Por cada uno de los ficheros, si no contiene el texto “CF”:

1.- Lee las lineas del fichero:

- Si la línea comienza con “C” (cabecera de remesa) lee la línea y luego va leyendo las lineas de detalle :

- Si la línea comienza con “D” (detalle) o “M” (modificación) lee el cuerpo de la línea y crea la situación de entrega correspondiente a ese cuerpo

Con estos datos se obtiene un vector de tipo RemesaDiario que contiene:

*private ArrayList<SituacionEntrega> situaciones;*

*private ArrayList<String> cabecera;*

***Tipo SituacionEntrega:***

*private Integer idEnvio;*

*private String codigoBarras;*

*private Integer coleccion;*

*private String email;*

*private String descripcion;*

*private String codColecc;*

*private String tiposMensaje;*

*private int idHist;*

*private String tiposit;*

*private Date fecha;*

*private Sitentrega situacion;*

*private String inforsit;*

*private String nombrefich;*

Por cada uno de los elementos del vector va tratando la remesa:

*tratarRemesa(ficheroEntrada.get(i))*

Comienza obteniendo de la cabecera el cliente y el remesacorr.

**A partir de la cabecera:**

* Si contiene una “C” en la situación entonces cierra la remesa:

*remesaDao.cerrarRemesa(remesacorr, ccliente, fecha)*

*UPDATE REMESA SET FECIERRE=to\_date(?,'yyyyMMdd')*

*,FKESTADOREM =? ,FKESTADOCORR ='C'*

*WHERE REMESACORR=? AND FKCLIENTE = (SELECT ICLIENTE FROM CLIENTES WHERE CODCLIENTE=? )*

* Si no contiene un “C” entonces cambia la situación de la remesa a la que venga en esa posición, siempre que la remesa no esté ya cerrada:

*remesaDao.cambiarEstadocorrRemesa(remesacorr, ccliente, cabeceraRemesa.get(7))*

*UPDATE REMESA SET FKESTADOCORR =?*

*WHERE REMESACORR=? AND FECIERRE is null AND FKCLIENTE =*

*(SELECT ICLIENTE FROM CLIENTES WHERE CODCLIENTE=? )*

**A partir de la lista de situaciones :**

Trata los envíos de la remesa:

*tratarSituacionesEntrega(remesa.getSituaciones(),remesacorr)*

Pasos en tratar situaciones de entrega:

**1.- Calculamos los ids de los envios de la lista a partir de su código de barras *:***

*envioDao.buscarIds(listaFD);*

*select idenvio,fkcolecc,email,descripcion,colecccorr*

*from envio inner join coleccion on envio.fkcolecc=coleccion.idcolecc*

*where codbarras=? order by fkcolecc*

**2.- Insertamos las situaciones en la tabla SITENTREGAS\_TEMP**

*situacionEntregaDao.insertBatchTemporal(listaFD);*

*INSERT INTO sitentregas\_temp (idenvio,fcarga,fsitentr, situacion,tiposit,procesado,correlativo,codsitentreg)*

*VALUES (?,sysdate,?, (select idsitentreg from sitentrega where codsitentreg=?),?,'0',*

*(SELECT COUNT(IDENVIO) + 1 FROM SITENTREGAS\_TEMP WHERE IDENVIO=?),?)*

**3.- Creamos otra lista (listaTerminados) con los ids de aquellos envíos cuya primera situación de entrega haya llegado antes de tres días laborables.**

*situacionEntregaDao.obtenerIdsTerminadasYCrearLista(listaTerminados);*

*select sitentregas\_temp.IDENVIO,sitentregas\_temp.FSITENTR,sitentregas\_temp.SITUACION,sitentregas\_temp.TIPOSIT,sitentregas\_temp.codsitentreg from sitentregas\_temp,*

*(select idenvio,max(correlativo) correlativo from sitentregas\_temp where idenvio in (*

*select idenvio from sitentregas\_temp where procesado=0 AND CORRELATIVO = 1 AND TRUNC(FCARGA) < (*

*SELECT DECODE ( to\_char(sysdate , 'd'), 1,trunc(sysdate - 5),*

*2,trunc(sysdate - 5),*

*3,trunc(sysdate - 5),*

*4,trunc(sysdate - 3),*

*5,trunc(sysdate - 3),*

*6,trunc(sysdate - 4),*

*7,trunc(sysdate - 5)*

*) from all\_objects where rownum=1)*

*) group by idenvio ) consulta*

*where sitentregas\_temp.idenvio = consulta.idenvio and sitentregas\_temp.correlativo=consulta.correlativo and sitentregas\_temp.tiposit='F'*

**4.- Obtenemos los envíos de esta listaTerminados a partir del id**

*envioDao.buscarIdsConIdEnnvios(listaTerminados);*

*select idenvio,fkcolecc,email,descripcion,colecccorr,codbarras from envio inner join coleccion on envio.fkcolecc=coleccion.idcolecc where idenvio=? order by fkcolecc*

**3.- Guardamos las situaciones de envio en la tabla hto\_entregas**

*situacionEntregaDao.insertBatch(listaTerminados);*

*MERGE INTO HTO\_ENTREGAS h USING DUAL*

*ON(h.IDENVIO=? and h.TIPOSIT=?)*

*WHEN NOT MATCHED THEN*

*INSERT (IDHIST,IDENVIO,TIPOSIT,FECHA,SITUACION,INFORSIT)*

*VALUES (AA6620Q00.NEXTVAL,?,?,?, (select idsitentreg from sitentrega where codsitentreg=?),?)*

**4.- Actualizamos las situaciones de entrega en los envíos**

*envioDao.updateSitEntrega(listaTerminados);*

*UPDATE ENVIO SET FKSITENTREG=(select idsitentreg from sitentrega where codsitentreg=?), ENVIADOPIF='0' WHERE IDENVIO=?*

**5.- Se tratan los ficheros PEE, si los hubiera de la lista de envíos terminados.**

*tratarFicherosFirma(listaTerminados);*

Por cada situación de entrega (envio) :

* Obtiene el código de la tabla de situaciones de entrega para montar el nombre del fichero que debería recibir
  + *situacionEntregaDao.obtenerCodXml(elemento.getSituacion().getCodsitentreg())*
* Si el código es null, no se espera fichero, por lo que no se hace nada.
* Si no es null, entonces se espera fichero
  + Busca en el directorio si existen los ficheros pdf y xml para ese código de barras.
    - Si no existen los ficheros con ese nombre, se monta el nombre del fichero alternativo con el otro código- codxmlpdf de la tabla de situaciones de entrega (10/16).
      * Se busca en el directorio si existen los ficheros pdf y xml para ese código de barras.
      * Si existen los dos:
        + Guardamos la situación y fecha de entrega en tabla SITENTREGAS\_TEMP, Procesado a 6, fecha de carga del día
        + Guardamos la situación de entrega en la tabla HTO\_ENTREGAS
        + Actualizamos la situación de entrega en el ENVÍO.
        + Se guardan en la BD y se mueven al backup

***pifToBD****(nombreCorto, idenvio,pdfMinuscula,xmlMinuscula);*

* + - Si existen los dos:
      * Se guardan en la BD y se mueven al backup

***pifToBD****(nombreCorto, idenvio,pdfMinuscula,xmlMinuscula);*

**FUNCION PIFTOBD**

1º Obtiene los ficheros del directorio y los inserta en la tabla ENVIO\_DOC

2ª Borra los ficheros del pif y los mueve al backup

3º Marca el envío como pendiente de imprimir

UPDATE *ENVIO SET PENDIMPRIMIR=1, NOMFICH=? WHERE IDENVIO=?*

4º Lee el fichero xml y obtiene el csv y la fecha de caducidad y los guarda en la tabla Envio

UPDATE ENVIO SET csv=?,FECADUCACSV=to\_date(?,'yyyy-MM-dd HH24:MI:SS'),REVISENTREG=0 WHERE IDENVIO=?

* + - * Se obtiene la aplicación del envío y se mira si tiene marca de envío pif automático en la tabla de aplicaciones. Si es así, se suben los ficheros desde el backup a su zona pif.
      * Se guardan en la tabla imagenpendiente con todos los datos

*imagenPendienteDao.insertImagenPendienteCompleto(elemento)*

*MERGE INTO imagenpendiente b*

*USING dual on( b.IDENVIO = ?)*

*WHEN NOT MATCHED THEN*

*INSERT (IDENVIO,NOMFICH,FSITENTR,FENTREGA)*

*VALUES (?,?||(select codxmlpdf from sitentrega where codsitentreg=?),SYSDATE,SYSDATE)*

*WHEN MATCHED THEN*

*UPDATE SET NOMFICH=?||(select codxmlpdf from sitentrega where codsitentreg=?),FSITENTR=SYSDATE,FENTREGA=SYSDATE*

Si ya existía de antes registro y ahora viene también situacion de entrega con fichero, actualiza la fecha de entrega

* + - Si alguno de los dos ficheros no existe
      * Se guardan en la tabla imagenpendiente sin la fecha de recepción

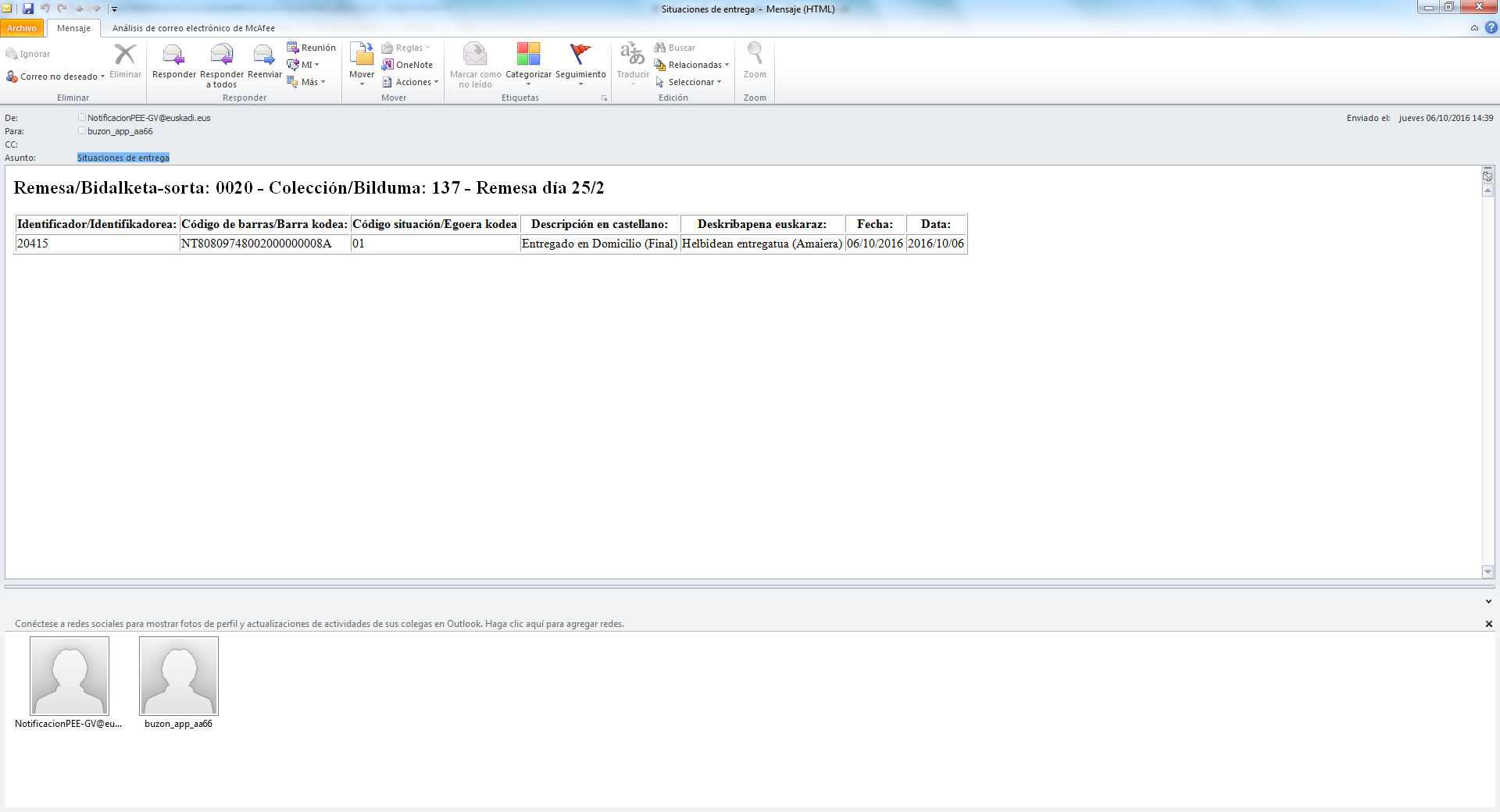
*MERGE INTO imagenpendiente b USING dual on( b.IDENVIO = ?)*

*WHEN NOT MATCHED THEN*

*INSERT (IDENVIO,NOMFICH,FSITENTR,FENTREGA)*

*VALUES (?,?||(select codxmlpdf from sitentrega where codsitentreg=?),SYSDATE,NULL)*

**6.- Obtenemos los envíos que están en la lista de terminados y mandamos emails dividiéndolas por colección con emails**

Asunto del mail :” Situaciones de entrega” 

**7.- Busca por aplicación los envíos en los que ha cambiado la situación de entrega y monta un fichero de situaciones que deja en la zona pif de aquellas aplicaciones que tengan marcado el campo enviopifauto a 1 en la tabla de aplicaciones.**

*crearFicherosSituacionesEntregaAutomaticas();*

Marca como enviadas las situaciones de entrega para no volver a enviarlas:

*envioDao.updateSitEntregaAEnviadoPif()*

*UPDATE ENVIO SET ENVIADOPIF='1' WHERE ENVIADOPIF='0'*

# PROCESA IMÁGENES PENDIENTES

Este proceso busca los ficheros pdf y xml que se encuentran en la tabla imagenpendiente pendientes de recibir. Si los encuentra los carga.

* Obtiene una lista de los ficheros pendientes de recibir

*ImagenPendienteDao.obtenerPendientes()*

*select \* from imagenpendiente where FENTREGA is null*

* Por cada elemento:

Por cada situación de entrega (envio) :

* Obtiene el código de la tabla de situaciones de entrega para montar el nombre del fichero que debería recibir
  + *situacionEntregaDao.obtenerCodXml(elemento.getSituacion().getCodsitentreg())*
* Si el código es null, no se espera fichero, por lo que no se hace nada.
* Si no es null, entonces se espera fichero
  + Busca en el directorio si existen los ficheros pdf y xml para ese código de barras.
    - Si no existen los ficheros con ese nombre, se monta el nombre del fichero alternativo con el otro código- codxmlpdf de la tabla de situaciones de entrega (10/16).
      * Se busca en el directorio si existen los ficheros pdf y xml para ese código de barras.
      * Si existen los dos:
        + Guardamos la situación y fecha de entrega en tabla SITENTREGAS\_TEMP, Procesado a 6, fecha de carga del día
        + Guardamos la situación de entrega en la tabla HTO\_ENTREGAS
        + Actualizamos la situación de entrega en el ENVÍO.
        + Se guardan en la BD y se mueven al backup

***pifToBD****(nombreCorto, idenvio,pdfMinuscula,xmlMinuscula);*

* + - Si existen los dos:
      * Se guardan en la BD y se mueven al backup

***pifToBD****(nombreCorto, idenvio,pdfMinuscula,xmlMinuscula);*

**FUNCION PIFTOBD**

1º Obtiene los ficheros del directorio y los inserta en la tabla ENVIO\_DOC

Primero busca en la tabla envio\_doc si existe registro. Si existe hace update con lso ficheros. Si no existe, entonces hace insert con los blob a null y luego hace update de ese registro con los ficheros.

*try {*

*boolean auto=conn.getAutoCommit();*

*conn.setAutoCommit(false);*

*String query = "SELECT F\_PDF, F\_XML FROM ENVIO\_DOC WHERE idEnvio=? FOR UPDATE";*

*lstm = conn.prepareStatement(query);*

*lstm.setLong(1, doc.getEnvio());*

*rs=lstm.executeQuery();*

*if(rs.next()){*

*blobXML=rs.getBlob("F\_XML");*

*blobPDF=rs.getBlob("F\_PDF");*

*}*

*else{*

*lstm.close();*

*query = "INSERT INTO ENVIO\_DOC (idEnvio, F\_PDF, F\_XML) VALUES (?, EMPTY\_BLOB(), EMPTY\_BLOB())";*

*lstm2 = conn.prepareStatement(query);*

*lstm2.setLong(1, doc.getEnvio());*

*lstm2.executeQuery();*

*lstm2.close();*

*query = "SELECT F\_PDF, F\_XML FROM ENVIO\_DOC WHERE idEnvio=? FOR UPDATE";*

*lstm = conn.prepareStatement(query);*

*lstm.setLong(1, doc.getEnvio());*

*rs=lstm.executeQuery();*

*if(rs.next()){*

*blobXML=rs.getBlob("F\_XML");*

*blobPDF=rs.getBlob("F\_PDF");*

*}*

*}*

2ª Borra los ficheros del pif y los mueve al backup

3º Marca el envío como pendiente de imprimir

UPDATE *ENVIO SET PENDIMPRIMIR=1, NOMFICH=? WHERE IDENVIO=?*

4º Lee el fichero xml y obtiene el csv y la fecha de caducidad y los guarda en la tabla Envio

UPDATE ENVIO SET csv=?,FECADUCACSV=to\_date(?,'yyyy-MM-dd HH24:MI:SS'),REVISENTREG=0 WHERE IDENVIO=?

* + - * Se obtiene la aplicación del envío y se mira si tiene marca de envío pif automático en la tabla de aplicaciones. Si es así, se suben los ficheros desde el backup a su zona pif.
      * Se actualiza en la tabla imagenpendiente con la fecha de entrega

*imagenPendienteDao.actualizaImagenPendiente(elemento)*

*UPDATE imagenpendiente SET FENTREGA=SYSDATE WHERE IDENVIO=?*

# PROCESA REMESA RESULTADOS

Trata los ficheros de resultado que vienen de Correos

Inicialmente, se separan en dos listas los ficheros que acaban en “-resultado” y los ficheros que acaban en “.bad”.

**Tratamiento del fichero de resultado:**

* Obtiene el fichero desde pif y lee las lineas del fichero
* Lee la línea de la remesa separando sus campos
  + Si el resultado es incorrecto (tiene N) se pone estado reabierta
  + Si es correcto (S) se pone estado enviada y no hace nada.
* Si el estado es “reabierta” busca el fichero .bad

**Tratamiento del fichero .bad ( si el resultado es incorrecto)**

*tratarFicheroResultadoBad(String nombre)*

Lee las lineas y obtiene la lista de errores de envio o errores de remesa.

* + Si hay errores de envío obtiene la lista de ids de envío e inserta en la tabla remesabad el envio con errores:

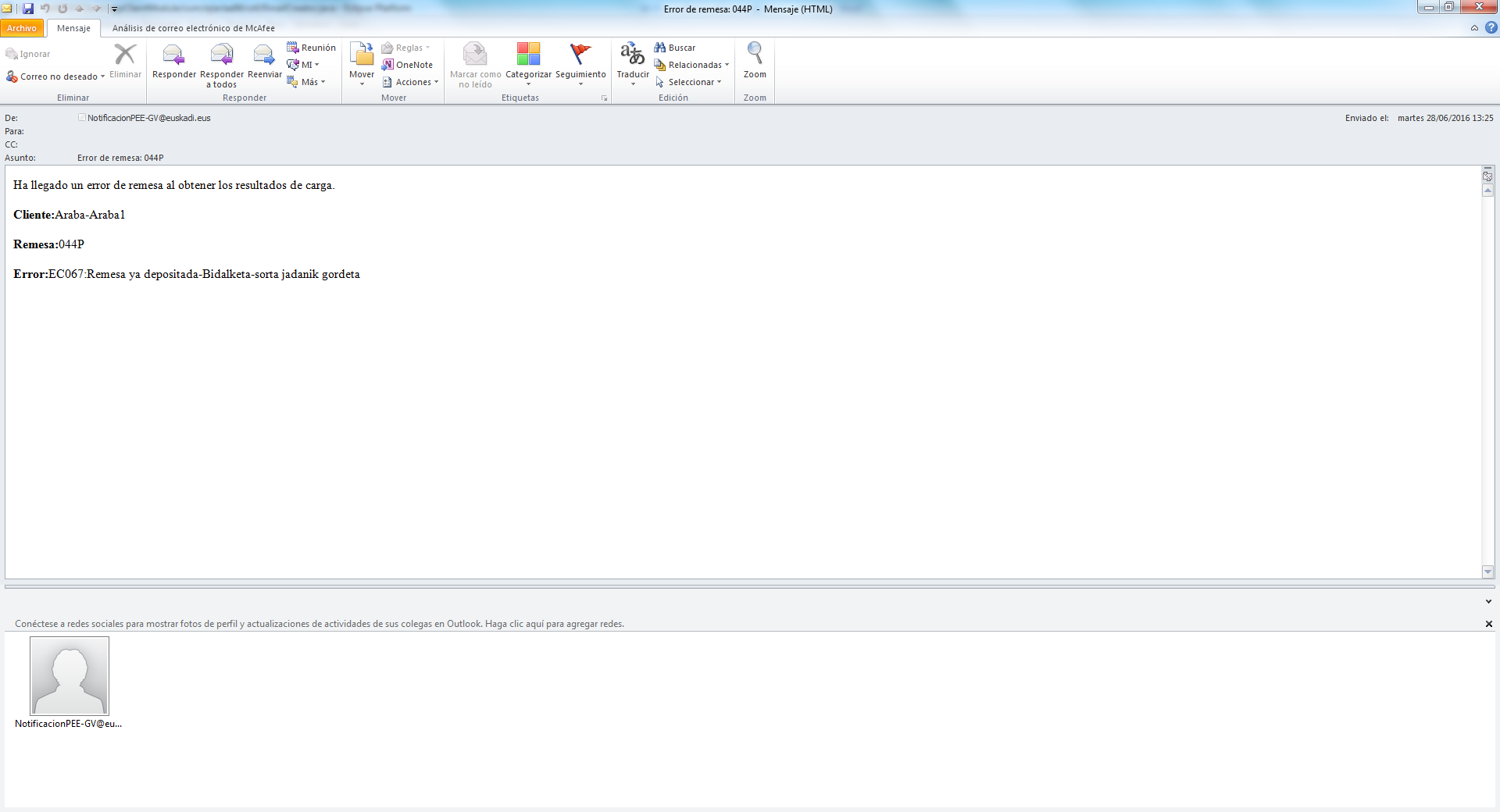
*INSERT INTO REMESABAD (NOMBREFIC,NROENVIO,FEEMISION,CODERROR,DETALLE,FKLOTE,FKENVIO) SELECT ?,nroenvio,?,?,?,fklote,? from remesa where idremesa=(select fkremesa from envio where idenvio=?)*

Y modifica en la tabla envío los envíos:

*UPDATE ENVIO SET corregido=0 WHERE IDENVIO IN (Select FKENVIO from remesabad inner join errorenvio on remesabad.coderror =errorenvio.coderror where errorenvio.estado='R' and remesabad.nombrefic=?)*

* + Si hay errores de remesa
    - Obtiene el cliente y envía mail a los email que hay en el properties que deben recibir los errores de remesa:

[*emailCorreoErrorRemesa=a-psantos@euskadi.eus##J-Estebanez@euskadi.eus##MJ-Mesanza@ejie.eus##L-Cabezudo@ejie.eus*](mailto:emailCorreoErrorRemesa=a-psantos@euskadi.eus##J-Estebanez@euskadi.eus##MJ-Mesanza@ejie.eus)



Mueve el fichero al backup

# PREPARAR REMESAS

Proceso que busca las remesas diarias abiertas y las pone como preparadas, abriendo una nueva remesa diaria para cada cliente.

Por cada la remesa diaria en estado preparada, se borran las colecciones que no tienen notificaciones, se crea el fichero de remesa/lote, lo dejamos en zona PIF de EDITRAN y cambiamos a estado Enviada.

Se lanza el proceso “PrepararRemesa” los días laborales, excluyendo los días festivos para la comunidad autónoma del Pais Vasco”

Busca las remesas abiertas

*remesaDao.buscaRemesasDiariasAbiertas()*

*select idRemesa,fkcliente,fkzona from remesa where fkestadorem='A' and tiporemesa='D'*

Por cada una de las remesas:

* Comprueba si es día festivo local para el cliente de la remesa.

*select count(1) as numFechas from diasfestivos where trunc(diafestivo) = trunc(sysdate) and idcliente=?*

* Si no es festivo:
  + Prepara la remesa

*UPDATE REMESA SET FKESTADOREM =?,FEPREPARADA=SYSDATE WHERE idremesa=?*

* + Borra las colecciones sin envíos de la remesa preparada

*DELETE coleccion*

*WHERE idcolecc IN (SELECT idcolecc FROM coleccion c*

*LEFT JOIN envio ON idcolecc=fkcolecc*

*WHERE fkcolecc IS NULL and c.fkremesa =?)*

* + Abre una nueva remesa en ese cliente

*INSERT INTO REMESA (FKCLIENTE, FKZONA, FKESTADOREM, REMESACORR, NROENVIO, FECREA, TIPOREMESA,IDREMESA) VALUES (?,?,?,?,0,SYSDATE,?,?)*

* + Borra colecciones sin notificaciones de remesa diaria en estado preparada
  + Crea fichero de remesa/lote con formato Sicer y se deja en zona PIF de EDITRAN. En tabla LOTE se guarda el nombre de fichero generado para la remesa, con fecha y hora de generación.
  + Cambiar estado de remesa a Enviada, con fecha de envío de remesa.

# PROCESO PIF

Inicialmente obtiene la lista de aplicaciones de la tabla aplicaciones que tienen el campo pif a 1.

Por cada una de las aplicaciones:

* Recorre el directorio en pif de esa aplicación. Si encuentra ficheros:
* Por cada uno de los ficheros llama a la función “procesarFichero”

Función procesarFichero:

* Comprueba el nombre del fichero y en función de cómo empieza realiza su tratamiento:
  + *Carga 🡪 procesaRemesa*
  + *RPEE 🡪 procesaPruebasEntregaRemesa*
  + *PEE 🡪 procesaPruebasEntregaEnvio*
  + *RSitEntrega 🡪 procesaSituacionesEntregaRemesa*
  + *ESitEntrega 🡪 procesaSituacionesEntregaEnvio*
  + *Borrar 🡪 procesaEliminarEnvio*
  + *Prepara 🡪 procesaPrepararColeccion*
  + *EstadoR 🡪 procesaEstadoRemesa*

**procesaRemesa**

* Obtiene los datos del fichero y comprueba en las validaciones que todo es correcto. Si hay errores monta el fichero de errores y lo devuelve.
* Xsd de validación: /datos/aa66/file/xsd/cargarremesa.xsd
* Si es correcto llama al método de create de RemesaCreator que obtiene todos los datos que se van a dar de alta.
* Guarda la remesa, las colecciones y los envíos. Si es remesa diaria busca la que esté abierta y ahí guarda los datos. Si no hay ninguna remesa diaria abierta se crea remesa diaria.  Si es remesa masiva crea una remesa.
* Por cada envío obtiene el código de barras,y valida que tiene el código de remesa de usuario (que va en el código de barras), antes de generar el envio y formar el código de barras.
* Una vez insertado todo, por cada envío obtiene el código de barras, monta la imagen y genera el xml de respuesta.

**ProcesaPruebasEntregaRemesa**

A partir del id de una remesa proporciona los ficheros pdf y xml de prueba de entrega de los envíos de la remesa que los contengan.

* Obtiene los datos del fichero y comprueba en las validaciones que todo es correcto. Si hay errores monta el fichero de errores y lo devuelve.
* Xsd de validación: /datos/aa66/file/xsd/PPEEremesa.xsd
* Se leen los datos del xml y se guardan en una remesaPif (id de remesa y uidpuestotramitador)
* Se busca la remesa por id. Si no la encuentra devuelve error.

Si la encuentra:

* + Se obtiene el puesto de las tablas de n38 🡪 si no lo encuentra : error

*SELECT N38GRUPOORGANICOUID,N38CODESTRUS,DISPLAYNAME,OU,n38extension FROM PUESTOPERSONA WHERE N38UIDPUESTO=?*

* + Se obtienen las aplicaciones del puesto de n38 🡪 si no coinciden con la aplicación del directorio : error

*SELECT DISTINCT(NVL(SUBSTR(N38UIDGRUPOUSUARIOS, 0, INSTR(N38UIDGRUPOUSUARIOS, '-')-1), N38UIDGRUPOUSUARIOS)) FROM GRUPOSPUESTO WHERE n38uidpuesto = ?*

* + Si no hay error, busca los envíos de la remesa. Por cada envío:
    - Busca los ficheros

*Select F\_PDF, F\_XML from ENVIO\_DOC where idenvio=?*

* + - Si hay ficheros, los agrega a un zip
  + Cuando termina con los envíos coloca el zip en la ruta pif de la aplicación:
    - /datos/aa66/tmp/procesoVueltaPif/app/idremesa.zip
  + Monta el xml de respuesta con el id de remesa y la ruta del fichero zip.

**ProcesaPruebasEntregaEnvio**

A partir del id de un envío proporciona los ficheros pdf y xml de prueba de entrega ( si los hubiera)

* Obtiene los datos del fichero y comprueba en las validaciones que todo es correcto. Si hay errores monta el fichero de errores y lo devuelve.
* Xsd de validación: /datos/aa66/file/xsd/PPEEenvio.xsd
* Se leen los datos del xml y se guardan en un envioPif (id de envio y uidpuestotramitador)
* Se busca si existe el envío. Si no existe : error

Si lo encuentra:

* + Se obtiene el puesto de las tablas de n38 🡪 si no lo encuentra : error

*SELECT N38GRUPOORGANICOUID,N38CODESTRUS,DISPLAYNAME,OU,n38extension FROM PUESTOPERSONA WHERE N38UIDPUESTO=?*

* + Se obtienen las aplicaciones del puesto de n38 🡪 si no coinciden con la aplicación del directorio : error

*SELECT DISTINCT(NVL(SUBSTR(N38UIDGRUPOUSUARIOS, 0, INSTR(N38UIDGRUPOUSUARIOS, '-')-1), N38UIDGRUPOUSUARIOS)) FROM GRUPOSPUESTO WHERE n38uidpuesto = ?*

* + - Si no hay error, busca los ficheros

*Select F\_PDF, F\_XML from ENVIO\_DOC where idenvio=?*

* + - Si no hay ficheros : error
    - Si hay ficheros, los agrega a un zip
  + Coloca el zip en la ruta pif de la aplicación:
    - /datos/aa66/tmp/procesoVueltaPif/app/idenvio.zip
  + Monta el xml de respuesta con el id de envio y la ruta del fichero zip.

**procesaSituacionesEntregaRemesa**

A partir de un id de remesa envía las situaciones de entrega de sus envíos

* Obtiene los datos del fichero y comprueba en las validaciones que todo es correcto. Si hay errores monta el fichero de errores y lo devuelve.
* Xsd de validación: /datos/aa66/file/xsd/ situacionesRemesa.xsd
* Se leen los datos del xml y se guardan en una remesaPif (id de remesa ,uidpuestotramitador y todas-si se quieren recibir todas o sólo las que no se han recibido anteriormente)
* Se busca la remesa por id. Si no la encuentra devuelve error.

Si la encuentra:

* + Se obtiene el puesto de las tablas de n38 🡪 si no lo encuentra : error

*SELECT N38GRUPOORGANICOUID,N38CODESTRUS,DISPLAYNAME,OU,n38extension FROM PUESTOPERSONA WHERE N38UIDPUESTO=?*

* + Se obtienen las aplicaciones del puesto de n38 🡪 si no coinciden con la aplicación del directorio : error

*SELECT DISTINCT(NVL(SUBSTR(N38UIDGRUPOUSUARIOS, 0, INSTR(N38UIDGRUPOUSUARIOS, '-')-1), N38UIDGRUPOUSUARIOS)) FROM GRUPOSPUESTO WHERE n38uidpuesto = ?*

* + Si no hay error, obtiene las situaciones de entrega de la remesa:

*obtenerSituacionesEntrega(idRemesa,remesaEntradaPif.isTodas(),codApp)*

*select envio.idenvio,*

*case when CODSITENTREG is not null then CODSITENTREG else '00' end as CODSITENTREG,*

*case when csitentreg is not null then csitentreg else 'Sin respuestas de correos' end as csitentreg,*

*case when esitentreg is not null then esitentreg else 'Correos-en erantzunik gabe' end as esitentreg,*

*to\_char(hto\_entregas.FECHA,'dd/MM/yyyy') as fecha from envio left join sitentrega on fksitentreg=idsitentreg*

*left join hto\_entregas on envio.idenvio=hto\_entregas.IDENVIO*

*where fkremesa=?*

*and envio.aplic=?*

*if (!todas)*

*and envio\_situ='0'*

*order by idhist desc*

* + Si se han encontrado situaciones de entrega las marca como enviadas

*UPDATE envio set envio\_situ='1' where fkremesa=? and aplic=? and envio\_situ='0' and fksitentreg is not null*

* Se monta el xml de respuestacon el id de remesa y por cada envío el id de envío, el código de situación y las descripciones en euskera y en castellano.

**procesaSituacionesEntregaEnvio**

A partir de un id de envio envía su situación de entrega

* Obtiene los datos del fichero y comprueba en las validaciones que todo es correcto. Si hay errores monta el fichero de errores y lo devuelve.
* Xsd de validación: /datos/aa66/file/xsd/situacionesEnvio.xsd
* Se leen los datos del xml y se guardan en un envioPif (id de envío ,uidpuestotramitador y todas-si se quieren recibir todas o sólo las que no se han recibido anteriormente)
* Se busca el envío por id. Si no lo encuentra devuelve error.

Si lo encuentra:

* + Se obtiene el puesto de las tablas de n38 🡪 si no lo encuentra : error

*SELECT N38GRUPOORGANICOUID,N38CODESTRUS,DISPLAYNAME,OU,n38extension FROM PUESTOPERSONA WHERE N38UIDPUESTO=?*

* + Se obtienen las aplicaciones del puesto de n38 🡪 si no coinciden con la aplicación del directorio : error

*SELECT DISTINCT(NVL(SUBSTR(N38UIDGRUPOUSUARIOS, 0, INSTR(N38UIDGRUPOUSUARIOS, '-')-1), N38UIDGRUPOUSUARIOS)) FROM GRUPOSPUESTO WHERE n38uidpuesto = ?*

* + Si no hay error, obtiene la situacion de entrega del envío:

*obtenerSituacionEntrega(idEnvio,envioEntradaPif.isTodas()*

*select envio.idenvio,*

*case when CODSITENTREG is not null then CODSITENTREG else '00' end as CODSITENTREG,*

*case when csitentreg is not null then csitentreg else 'Sin respuestas de correos' end as csitentreg,*

*case when esitentreg is not null then esitentreg else 'Correos-en erantzunik gabe' end as esitentreg,*

*to\_char(hto\_entregas.FECHA,'dd/MM/yyyy') as fecha*

*from envio left join sitentrega on fksitentreg=idsitentreg left join hto\_entregas on*

*envio.idenvio=hto\_entregas.IDENVIO*

*where envio.idenvio=?*

***if (!todas)***

*and envio\_situ='0'*

*order by idhist desc*

* + Si se han encontrado situación de entrega la marca como enviadas

*UPDATE envio set envio\_situ='1' where idenvio=? and envio\_situ='0' and fksitentreg is not null*

* + Si no se ha encontrado situación de entrega : error
* Se monta el xml de respuesta con el id de envío, el código de situación y las descripciones en euskera y en castellano.

**procesaEliminarEnvio**

Borra un envío a partir de su id, siempre que la colección se encuentre en estado abierta

* Obtiene los datos del fichero y comprueba en las validaciones que todo es correcto. Si hay errores monta el fichero de errores y lo devuelve.
* Xsd de validación: /datos/aa66/file/xsd/eliminarEnvio.xsd
* Se leen los datos del xml y se guardan en un envioPif (id de envío y uidpuestotramitador)
* Se busca el envío por id. Si no lo encuentra devuelve error.

Si lo encuentra:

* + Se obtiene el puesto de las tablas de n38 🡪 si no lo encuentra : error

*SELECT N38GRUPOORGANICOUID,N38CODESTRUS,DISPLAYNAME,OU,n38extension FROM PUESTOPERSONA WHERE N38UIDPUESTO=?*

* + Se obtienen las aplicaciones del puesto de n38 🡪 si no coinciden con la aplicación del directorio : error

*SELECT DISTINCT(NVL(SUBSTR(N38UIDGRUPOUSUARIOS, 0, INSTR(N38UIDGRUPOUSUARIOS, '-')-1), N38UIDGRUPOUSUARIOS)) FROM GRUPOSPUESTO WHERE n38uidpuesto = ?*

* + Si no hay error, busca si la remesa a la que pertenece el envío está abierta
    - Si no está abierta: error
    - Si está abierta elimina el envío

*DELETE FROM ENVIO WHERE IDENVIO=?*

* Se monta el xml de respuesta con el idenvio y el resultado eliminar

**procesaPrepararColeccion**

Cambia el estado de una colección a preparada a partir de su id

* Obtiene los datos del fichero y comprueba en las validaciones que todo es correcto. Si hay errores monta el fichero de errores y lo devuelve.
* Xsd de validación: /datos/aa66/file/xsd/prepararColeccion.xsd
* Se busca la colección por id. Si no la encuentra devuelve error.

Si la encuentra:

* + Se obtiene el puesto de las tablas de n38 🡪 si no lo encuentra : error

*SELECT N38GRUPOORGANICOUID,N38CODESTRUS,DISPLAYNAME,OU,n38extension FROM PUESTOPERSONA WHERE N38UIDPUESTO=?*

* + Se obtienen las aplicaciones del puesto de n38 🡪 si no coinciden con la aplicación del directorio : error

*SELECT DISTINCT(NVL(SUBSTR(N38UIDGRUPOUSUARIOS, 0, INSTR(N38UIDGRUPOUSUARIOS, '-')-1), N38UIDGRUPOUSUARIOS)) FROM GRUPOSPUESTO WHERE n38uidpuesto = ?*

* Comprueba si la remesa a la que pertenece la colección está en estado abierta o reabierta.
  + Si no está : error
  + Si está :
    - Se prepara la colección

*UPDATE COLECCION SET FKESTADOCOL =?, FEPREPARADA=SYSDATE,UIDPUESTOPREP=?,TRAMITAPREP=?,TELEFTRAPREP=?,FKDEPARPREP=?,FKCORGPREP=?*

*WHERE idcolecc=?*

* + - Si la remesa no es diaria, también se prepara la remesa

*UPDATE REMESA SET FKESTADOREM =?,FEPREPARADA=SYSDATE WHERE idremesa=?*

* Se monta el xml de respuesta con el idcoleccion y el resultado preparar

**procesaEstadoRemesa**

A partir del id de una remesa se envía la situación de la remesa

* Obtiene los datos del fichero y comprueba en las validaciones que todo es correcto. Si hay errores monta el fichero de errores y lo devuelve.
* Xsd de validación: /datos/aa66/file/xsd/estadoRemesa.xsd
* Se leen los datos del xml y se guardan en una remesaPif (id de remesa y uidpuestotramitador )
* Se busca la remesa por id. Si no la encuentra devuelve error.

Si la encuentra:

* + Se obtiene el puesto de las tablas de n38 🡪 si no lo encuentra : error

*SELECT N38GRUPOORGANICOUID,N38CODESTRUS,DISPLAYNAME,OU,n38extension FROM PUESTOPERSONA WHERE N38UIDPUESTO=?*

* + Se obtienen las aplicaciones del puesto de n38 🡪 si no coinciden con la aplicación del directorio : error

*SELECT DISTINCT(NVL(SUBSTR(N38UIDGRUPOUSUARIOS, 0, INSTR(N38UIDGRUPOUSUARIOS, '-')-1), N38UIDGRUPOUSUARIOS)) FROM GRUPOSPUESTO WHERE n38uidpuesto = ?*

Si no hay error, monta el xml con el estado de la remesa ( lo tenemos en la remesa que hemos buscado anteriormente)

# PROCESO CALCULO DE INCUMPLIMIENTOS EN ENTREGA DE NOTIFICACION

Tratamos todos los envíos con PEE y sin cálculo el incumplimiento de entrega.

De cada PEE de fichero con formato .xml, obtenemos la fecha de admisión del sobre en Correos, las situaciones intermedias de entrega (primer intento, segundo intento) o/y situación final de entrega.

Ordenadas las situaciones por fecha, se calcula si existe incumplimiento entre:

* “Admitida y primer reparto” Diferencia de días entre la primera fecha de admisión y siguiente fecha, supera los días indicados en el mantenimiento de incumplimiento
* “Primer reparto y (segundo reparto o entrega final). Diferencia de días y/o horas entre segunda fecha y tercera fecha, donde la situación de entrega final sea diferente a Entregada/Devolución en Oficina y
  + - supera los días indicados en el mantenimiento de incumplimiento
  + - o supera a las horas indicadas en el mantenimiento
  + - o las horas de entrega están en mismo horario de mañana o mismo horario de tarde. En horario de mañana hasta las 12:00 y de tarde a partir de las 15:00.
* Cálculo entre fechas:
* No se tienen en cuenta en el cómputo de días, los días incluidos como festivos y nacionales, incluidos desde el mantenimiento de festivos

Se marcan los incumplimientos existentes y envío calculado el incumplimiento.

FUNCIONAMIENTO:

Por cada envío con PEE sin calculo de incumplimiento:

1.- Se obtienen los envios de la tabla ENVÍO\_DOC donde procesado está a 0 y certificado a 0

Y por cada envío se lee el PEE.xml para obtener las fechas de entrega

Se guardan las fechas con horas de todos los codrazon del fichero xml y la fecha de entrega del sobre en Correos, para calcular y marcar los incumplimientos, dejando el envio como PROCESADO.

Grabar en tabla FECHATRAZABIL , las fechas de fichero PEE en formato xml de:

* ***Fecha admitida en Correos***

*Con codrazon a A*

De la PEE en formato xml, valor de fecha en fechaadmisionasociado

**fechaadmisionasociado** = “2017-09-27t00:00:00+01:00”

* ***Fechas de reparto***

Se obtienen todas las fechas y horas de entrega del fichero PEE, excluyendo las duplicadas.

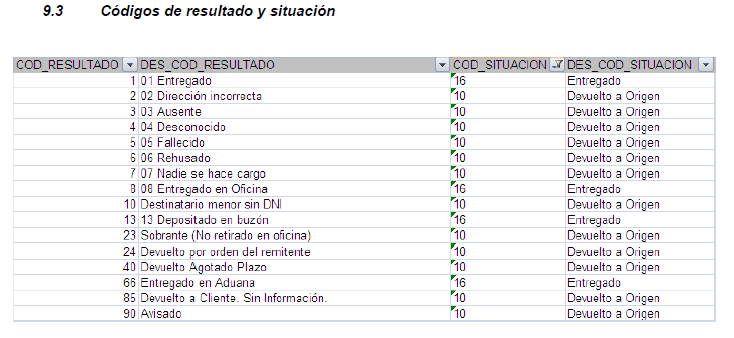
*Con codrazon a valor de codrazon en xml*

*Fecha y hora de cada codrazon*

Todos los codrazon que existen con su fecha y hora que existen en <eventosenvio> :

fechahora=”2018-04-24T12:44:00+01:00”

* + codrazon= “1”



**Cálculo de incumplimiento**

Una vez cargadas las fechas de la notificación, se ordenan por orden ascendente.

La primera fecha es fecha de admisión y segunda fecha y hora será la primera (entrega final, devolución, intento de entrega) y la última fecha es la entrega final o devolución. Las fechas y horas intermedias se consideran intentos de entrega.

Calcular los posibles incumplimientos y se graba:

* el código de incumplimiento en el último registro que se compara
* nro. de días entre fechas cuando la comparación es entre fechas excluyendo los sábados, domingos y días festivos definidos en la tabla de Festivos de cada zona,
* nro. de horas cuando la comparación es entre horas
* Marca mismo horario, si los intentos de entrega están realizados los 2 en mismo horario (mañana hasta las 15:00 o tarde desde las 15:00)
* IDINCUMPLI a 2, cuando :
  + Ordenados los registros del IDENVIO por FECHA y HORA.

La diferencia entre fecha de codrazon A y el siguiente registro y es mayor al valor indicado en tabla INCUMPLIPLAZO.ENTREFECHA y se graba el código de incumplimiento en el último registro comparado.

* IDCINCUMPLI a 3, cuando :
  + Ordenados los registros del IDENVIO por FECHA y HORA.
* La diferencia entre la fecha y hora de primer reparto y siguiente fecha y hora o segunda fecha de reparto y siguiente fecha de reparto, siempre que está siguiente fecha no sea la última fecha de reparto con situación de entrega/devolución en oficina.
* Es incumplimiento de fecha entre intentos de reparto, cuando la diferencia entre días es mayor a la especificada en el mantenimiento de incumplimientos.
* Es incumplimiento de horas entre intentos de reparto, cuando la diferencia entre horas es mayor a la especificada en el mantenimiento de incumplimientos.
* Es incumplimiento de horarios entre intentos de reparto, cuando están en mismo horario. En horario de mañana hasta las 15:00 y de tarde a partir de las 15:00.
  + Se graba el código de incumplimiento en el último registro comparado.

**Excel con incumplimientos** De todos los envios con algún incumplimiento y sin la marca de “generadoexcel” y sin fecha de generación del excel, sacar el envio  con incumplimientos y los siguientes datos:

* Código de barras
* Fecha de admisión del envio en correos
* Fecha de admisión
* Fecha 1º intento: 2º fecha, si no es la última de la notificación.
* Fecha 2º intento: 3º fecha, si no es la última de la notificación.
* Fecha 3º intento: 4º fecha, si no es la última de la notificación.
* Fecha entrega: última fecha con codrazon (1,10,18)
* Fecha entrega en oficina: última fecha con codrazon (8)
* Fecha de devolución en oficina: última fecha con codrazon(9/23)
* Fecha devolución: última fecha con  codrazon diferente a (1,3,8,9,23,10,12,14,18)
* Observaciones: descripción de última fecha con codrazon diferente a (1,3,8,9,23,10,12,14,18)
* Dias de reparto, nro. de dias de  Incumplimiento a 2
* Dias de intento, nro. de días de incumplimiento a 3
* Horas intento de entrega, nro. de horas de incumplimiento a 3
* Check de incumplimiento en reparto con Incumplimiento a 2
* Check de incumplimiento en intento de entrega con Incumplimiento a 3

Los check de incumplimientos mostrados en el Excel se suman y se graba el sumatorio al final de los incumplimientos:

En cada columna de incumplimiento se muestra el precio de penalización del incumplimiento

Y el resultado de precio penalización por nro. de incumplimientos

Y una suma total de incumplimientos

Al pintar el número de dias de incumplimiento cuando el incumplimiento es 3, si el número de días es 0 o null no pintar nada para que no se machaque por si hay pintado un incumplimiento anterior.

Extraer los incumplimientos de la tabla FECHATRAZABIL.

Columnas a grabar:

Código de barras

                Fecha Admisión Correos (codRazon = “A”)

                Fecha Reparto 1 (Fecha con hora) 2º fecha, hora, en el caso que no sea la última fecha, hora.

                Fecha Reparto 2 (Fecha con hora) 3º fecha, hora, en el caso que no sea la última fecha, hora.

  Fecha Reparto 3 (fecha con hora) 4º fecha, hora, en el caso que no sea la última fecha, hora.

  Fecha Entrega (codrazon 1,10,18), de la última fecha, hora. Cuando tiene fecha entrega oficina, se pinta también aquí.

                Fecha Entrega Oficina ( codrazon 8),de  la última fecha, hora

                Fecha Devolucion Oficina (codrazon 9/23), de la última fecha, hora

  Fecha Devolución (codrazon diferente a 1,3,8,10,18,9/23), de la última fecha, hora . a. Cuando tiene fecha devolución en oficina, se pinta también aquí.

                Observaciones (descripción de sitentrega dif.a codrazon 1,3,8,10,18,9/23).buscamos por el código de Correos)

                Dias Reparto (fkincumplimiento = 2)

                Dias Intento Entrega (fkincumplimiento = 3)

                Horas Intento Entrega (fkincumplimiento = 3)

                Incumplimiento Reparto (Donde codrazon=”A ponemos un 1)

                Incumplimiento Intento Entrega (Donde fkincumplimiento = 3 ponemos a 1)

En columna “Observaciones” se muestra :

* descripción de situación de entrega diferente a codrazon 1,3,8,9/23.

: Listado de incumplimiento que se muestra desde “Consultas y Listados”    orden de las columnas a mostrar.

- fecha, acción, incumplimiento, dias dif., horas dif., mismo periodo horario

En vez de poner el nro. De dias y horas de diferencia y “Si” en mismo periodo horario, poner la marca “X” en cada uno de ellos..

Mantener el texto de la primera acción Admitida, de la última y las intermedias poner en orden secuencial “1er. Intento de entrega” y “2º intento de entrega”

Los check de incumplimientos mostrados en el Excel se suman y se graba el sumatorio al final de los incumplimientos:

En cada columna de incumplimiento se muestra el precio de penalización del incumplimiento

Y el resultado de precio penalización por nro. de incumplimientos

Y una suma total de incumplimientos

