

Causa raíz NO se envía el folio de orden en setInicioTrans2 hacia Origis

Autor: Huilver Nolasco Aguilar

Fecha: Marzo del 2025

Contenido

Descripción general.....	3
Causa.....	3
Solución.....	10

Descripción general

Recientemente se agregó un nuevo método en origis llamado `setInicioTrans2` el cual soporta nuevos campos relacionados con la orden, en el caso particular del folio de orden el dato que debe mandarse es el campo `sPedidoNumero`, sin embargo se detectó que para algunas ventas que son de SAD, este dato viaja en 0 y para otras dicho dato si viaja correctamente.

Causa

La venta que se tomó para análisis es la venta V10982000088983 que corresponde al pedido 7007373420APP.

Lo primero que se hace es revisar los datos generales del pedido, así como sus formas de pago

```
--consulta de cabecera de venta
select idTransaccion, fechaTransaccion, sadOrdenId from trnTransaccionesCab where folioTransaccion='V10982000088983';
```

idTransaccion	fechaTransaccion	sadOrdenId
596E48EF-5EE2-EF11-95ED-10E7C6374BB8	2025-02-03 12:44:48.247	FF719F1B-87CF-4FC4-892B-83CE3100D617

```
--cabecera de orden asociado a la venta
select sadOrdenId, ordenFolio, codigoFormaPago, monedero from sadOrdenes
where sadOrdenId = (select sadOrdenId from trnTransaccionesCab where folioTransaccion='V10982000088983');
```

sadOrdenId	ordenFolio	codigoFormaPago	monedero
FF719F1B-87CF-4FC4-892B-83CE3100D617	7007373420APP	41	9690000002657

```
--formas de pago de la orden
select sadOrdenId, fp.codigoFormaPago, fpcat.descripcion, fpcat.tipoFormaPago, monedero, referenciaTarjeta from sadCentralOrdenFormasPago fp
join genFormasDePagoCat fpcat on fp.codigoFormaPago=fpcat.codigoFormaPago
where fp.sadOrdenId=(select sadOrdenId from trnTransaccionesCab where folioTransaccion='V10982000088983');
```

sadOrdenId	codigoFormaPago	descripcion	tipoFormaPago	monedero	referenciaTarjeta
FF719F1B-87CF-4FC4-892B-83CE3100D617	41	PAYPAL	SAD3		

Como se puede observar en la imagen anterior esta venta corresponde a una orden que fue pagada con PAYPAL(41) y por lo tanto es correcto que el campo **monedero y referencia tarjeta no tengan dato**

Despues se procede a corroborar que lo reportado en el log de `setInicioTrans2` de origis contra el log de monedero que se tiene para estar seguro que se trata del mismo caso.

	sTarjeta=9690000002657, sSucId=1098, sCajalId=20, sTerminalId=175327, sOrigen=2, sPedidoNumero=0,
setInicioTrans2	sReferenciaNumero=0, sCarritoNumero=0, sAuthCode=d98c9ff72fb6dbd0843d61e81bcd62bd

Dado que en la operación de `commitTrans` se registra de ser el caso el número de monedero, el folio de autorización y el número de venta, con esto se puede corroborar que se trata del mismo caso

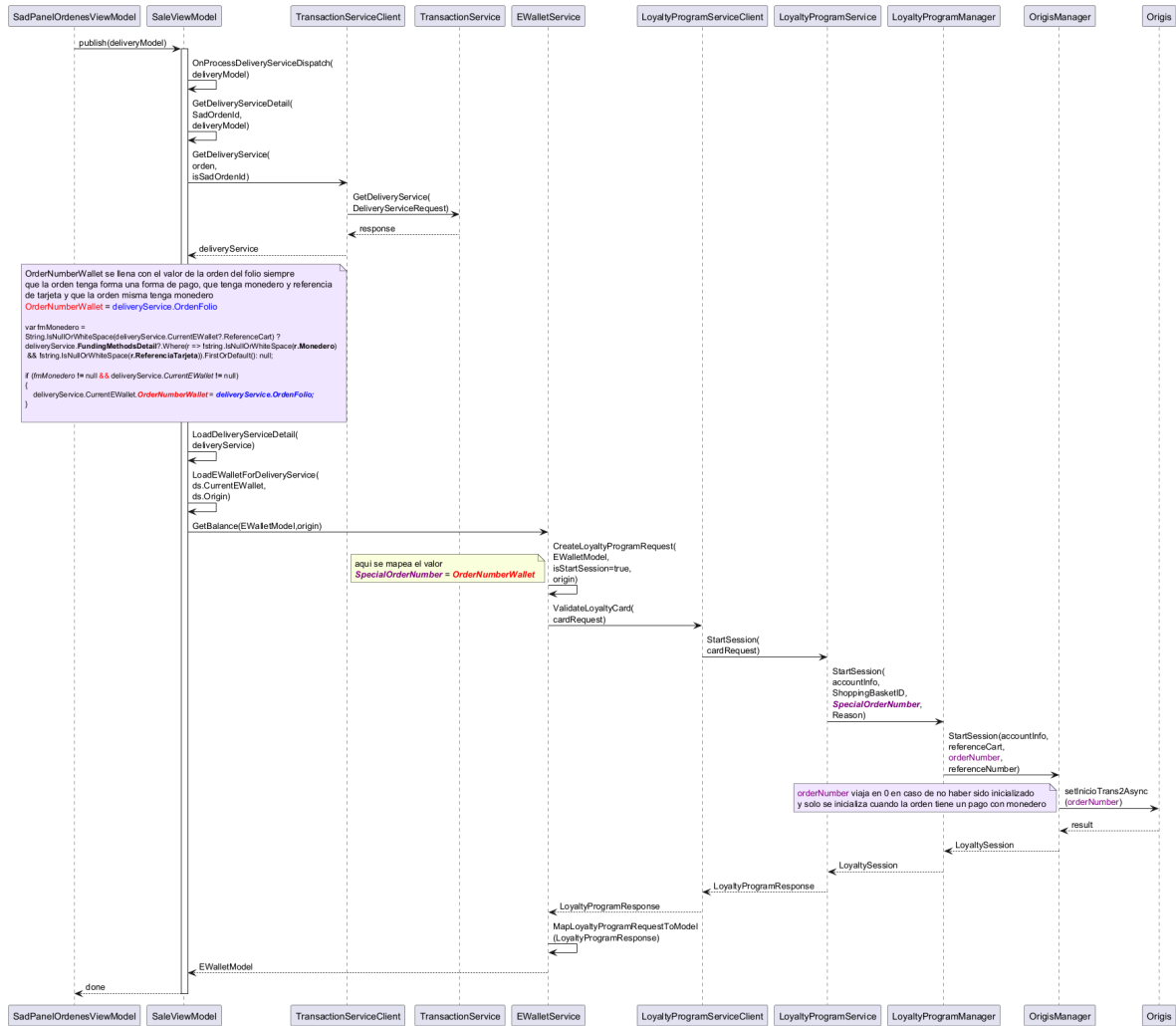
```
[2025-02-03 12:42:47:236],FARMAX_WSC,MONEDERO,INFO,,Request:<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <Action s:mustUnderstand="1" xmlns="http://schemas.microsoft.com/ws/2005/05/addressing/none">https://pos.monederodelahorro.net/online2/commitTrans</Action>
  </s:Header>
  <s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <commitTrans xmlns="https://pos.monederodelahorro.net/online2/">
      <Tarjeta>9690000002657</Tarjeta>
      <SucId>1098</SucId>
      <CajaId>20</CajaId>
      <TerminalId>175327</TerminalId>
      <SessionId>14352336926</SessionId>
      <Articulos>7501288209067,1,747,747|0650240057946,2,22.90002400,45.80004800|7506472801310,1,10.0000,10.0000</Articulos>
      <NumTicket>V10982000088983</NumTicket>
      <SaldoRedimido>0</SaldoRedimido>
      <ImporteTotal>802.8000</ImporteTotal>
      <FechaOperacion>20250203</FechaOperacion>
      <EntregaDomicilio>2</EntregaDomicilio>
      <FormaPago>0</FormaPago>
      <AuthCode>d98c9ff72fb6dbd0843d61e81bcd2bd</AuthCode>
    </commitTrans>
  </s:Body>
</s:Envelope>
```

Se observa que el número de tarjeta 9690000002657 y el numero de autorizacion d98c9ff72fb6dbd0843d61e81bcd2bd coinciden, por lo tanto la venta V10982000088983 es la correcta para el análisis.

A continuación se describe de manera general cómo es que con una orden SAD se interactúa con Orígenes enfocado al método setInicioTrans2 .

Después de que la orden ha llegado a la sucursal para su atención, dicha orden puede ser visualizada en el panel de órdenes de las cajas, una vez que se atiende la orden esta debe generar una comanda pero no sin antes haber obtenido la información de la orden, es decir su información general, el detalle de productos así como sus formas de pago. Después de obtener dicha información esta se coloca dentro del flujo del POS para la captura de productos y su despacho, justo en ese mapeo de los datos de la orden hacia los datos de la venta y al ejecutar la comanda, se dispara el inicio de sesión con orígenes y es ahí donde el número de pedido(folio orden) se notifica, sin embargo en el mapeo establecido entre la orden y la venta asociada, el número de pedido se mapea solo bajo ciertas condiciones que básicamente son, que la orden tenga un monedero asociado y que tenga una forma de pago mda, si esto no se cumple la consecuencia es que termina mandando 0 por defecto.

Se agrega un diagrama de secuencia para tener más detalle:



La imagen original se puede ver aquí [sad_enviofolioorden_setiniciotrans.png](#)

En código los lugares claves son

1.-Generación de comanda de la orden

```

C# ITTransactionPayment  C# BankTender  C# GenericTender  C# ILoyaltyProgramService  C# SadPanelOrdenesViewModel x
> Q- publis  x  Ce W .*  1/1  ↑ ↓  ⋮
354
355
356 >  /// Procesa la acción para el boton de acción 1 de las ordenes ...
357
358
359
360
361  @!usage  & Juan Murillo
362  public void ProcessButton1Action(SadOrderDisplayModel order)
363  {
364      switch (order.Status)
365      {
366          // orden de servicio pendiente
367          case "P":...
368
369          // orden de servicio con comanda impresa, y pendiente de despachar
370          case "I":
371              if (order.OrderType == SADORdenes.TIPO_ORDEN_VALUES.SERVICIO_DOMICILIO_PROGRAMADA
372                  && DateTime.Now.AddMinutes(PosAppContext.Instance.SystemConfiguration.TiempoDespachoOrdenProgramada) < order.OrdersScheduleDateTime){...}
373              else if (order.OrderType == SADORdenes.TIPO_ORDEN_VALUES.SERVICIO_DOMICILIO && order.ContainsVouchers){...}
374
375              this._eventAggregator.GetEvent<ProcessDeliveryServiceDispatchEvent>().Publish((new DeliveryServiceModel() { SadOrdenId = order.OrderID, Origin = order.OriginDisplay }));
376              this.Close();
377
378              break;
379
380          // orde de servicio en despacho
381          case "D":...
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396

```

2.-Inicia la atención de la orden

```

C# SaleViewModel x  C# PaymentMethodViewModel  C# TenderManager  C# LoyaltyProgramServiceClient  C# TenderServiceClient  C# PaymentMethodModel  C#
C# ITTransactionPayment  C# BankTender  C# GenericTender  C# ILoyaltyProgramService  C# SadPanelOrdenesViewModel
10385
10386  /// </summary>
10387  /// <param name="deliveryModel">identificador del SAD</param>
10388  @!usage  & Juan Murillo
10389  private void OnProcessDeliveryServiceDispatch(DeliveryServiceModel deliveryModel)
10390  {
10391      var confirmed_bool = this.Sale.SaleDetailList.Count > 0;
10392      if (confirmed){...}
10393      else{...}
10394
10395      if (confirmed)
10396      {
10397          this.ClearSaleVm();
10398          this.LoadDeliveryServiceDetail(this.GetDeliveryServiceDetail(deliveryModel.SadOrdenId.ToString(), isSadOrdenId: true, deliveryModel.Origin));
10399      }
10400
10401
10402
10403
10404
10405
10406
10407
10408
10409
10410
10411

```

3.-Se consulta la información de la orden mediante la solicitud de un servicio de till service

```

C# SaleViewModel x  C# PaymentMethodViewModel  C# TenderManager  C# LoyaltyProgramServiceClient  C# TenderServiceClient
C# ITTransactionPayment  C# BankTender  C# GenericTender  C# ILoyaltyProgramService  C# SadPanelOrdenesViewModel
9964
9965  private DeliveryServiceModel GetDeliveryServiceDetail(string orden, bool isSadOrdenId = false, string origin = "")
9966  {
9967      try
9968      {
9969          using (var service = new TransactionServiceClient())
9970          {
9971              deliveryService = service.GetDeliveryService(orden, isSadOrdenId);
9972              deliveryService.Origin = origin;
9973          }
9974      }
9975      return deliveryService;
9976
9977
9978
9979
9980
9981
9982
9983
9984
9985
9986
9987
9988
9989
9990
9991
9992
9993
9994
9995
9996
9997
9998
9999

```

4.-Se consulta el servicio

```

C# ITransactionPayment  C# BankTender  C# GenericTender  C# ILoyaltyProgramService  C# SadPanelOrdenesViewModel  C# TransactionServiceClient.partial x
81 >      /// Metodo que obtiene una orden de servicio.....
82 >      [usage Juan Murillo]
83 >      public DeliveryServiceModel GetDeliveryService(string orden, bool isSadOrdenId = false)
84 >      {
85 >      ...
86 >      DeliveryServiceResponse response = this.UseService((ITransactionService transactionService) => transactionService.GetDeliveryService(request));
87 >
88 >      return this.MapDeliveryServiceResponse(response);
89 >      }

```

5.-Se hace el mapeo de los datos de la orden hacia un response, haciendo notar el mapeo relevante de los datos que más adelante se usan, por ejemplo está el monedero de la forma de pago de la orden, así como su referencia de tarjeta y los datos del monedero de la orden en el currentEwallet

```

C# SaleViewModel  C# PaymentMethodViewModel  C# TenderManager  C# LoyaltyProgramServiceClient  C# TenderServiceClient  C# PaymentMethodModel  C# TenderProviderFactory  C# EwalletTender
C# ITransactionPayment  C# BankTender  C# GenericTender  C# ILoyaltyProgramService  C# SadPanelOrdenesViewModel  C# TransactionServiceClient.partial x
143 private DeliveryServiceModel MapDeliveryServiceResponse(DeliveryServiceResponse response)
144 {
145     var ordenDeliveryService = response.DeliveryServices[0];
146     string ordenNumberWallet = orden.OrdenFolio;
147     foreach (DeliveryServiceFundingMethod fm in orden.DeliveryServiceFundingMethods ?? new List<DeliveryServiceFundingMethod>())
148     {
149         var item = new DeliveryServiceFundingMethodModel
150         {
151             Monedero = fm.Monedero,
152             ReferenciaTarjeta = fm.ReferenciaTarjeta,
153         };
154         dsm.FundingMethodsDetail.Add(item);
155     }
156     if (!string.IsNullOrEmpty(orden.Monedero))
157     {
158         dsm.CurrentEwallet = new EwalletModels.EwalletModel() { FolioEwallet = orden.Monedero, ReferenceCart = referenceCart, OrderNumberWallet = orden.NumberWallet, ReferenceNumberWallet = referenceWallet };
159     }
160     return dsm;
161 }

```

6.-Se cargan los datos de la orden para que sea procesada por el POS, en donde se puede observar que el llenado de OrdenNumberWallet depende de que exista una forma de pago de la orden que tenga monedero y que tenga referenciaTarjeta y que la orden tenga los datos del monedero en currentEwallet. Sin embargo la Orden en cuestión fue pagada con PayPal y **su forma de pago no tiene monedero ni referencia de tarjeta** así que el dato OrdenNumberWallet **no se llena**

```

C# SaleViewModel x  C# PaymentMethodViewModel  C# TenderManager  C# LoyaltyProgramServiceClient  C# TenderServiceClient  C# PaymentMethodModel  C# TenderProviderFactory  C# EwalletTender
C# ITransactionPayment  C# BankTender  C# GenericTender  C# ILoyaltyProgramService  C# SadPanelOrdenesViewModel  C# TransactionServiceClient.partial
9991 private void LoadDeliveryServiceDetail(DeliveryServiceModel deliveryService)
9992 {
9993     if (deliveryService == null) { ... }
9994     // Si no hay numero de carrito buscamos si en la forma de pago de monedero se cuenta con alguno
9995     var fmMonedero = deliveryService.FundingMethodsDetail?.Where(r => !string.IsNullOrEmpty(r.Monedero) || !string.IsNullOrEmpty(r.ReferenciaTarjeta))
9996     : null;
9997     if (fmMonedero != null || deliveryService.CurrentEwallet != null)
9998     {
9999         deliveryService.CurrentEwallet.OrderNumberWallet = deliveryService.OrdenFolio;
10000     }
10001     if (!this.LoadEwalletForDeliveryService(deliveryService.CurrentEwallet, deliveryService.Origin))
10002     {
10003         Log.Write(string.Format("Servicio a domicilio FOLIO: {0}, no se logro procesar monedero", deliveryService.OrdenFolio));
10004     }
10005     this.Sale.CurrentDeliveryService = deliveryService;
10006 }

```

7.-Se consultan los datos del monedero hacia Origis en caso de que se cuente con el monedero, para despues colocar dicha información en el POS

```

10066 private bool LoadWalletForDeliveryService(EWalletModel ewallet, string origin)
10067 {
10068     if (ewallet != null && !string.IsNullOrEmpty(ewallet.FolioEwallet))
10069     {
10070         using (var service = new EWalletService())
10071         {
10072             ewalletInfo = service.GetBalance(new EWalletModel { BusinessUnitId = PosAppContext.Instance.BusinessUnit.BusinessUnitId, EmployeeId = PosAppContext.Instance.Cashier.Id, TillId = PosAppContext.Instance.TillId });
10073         }
10074     }
10075     if (ewalletInfo != null && (ewalletInfo.ErrorCode == EWallet.Models.Enum.EWalletErrorCode.None || ewalletInfo.ErrorCode == EWallet.Models.Enum.EWalletErrorCode.LockedCardToRedeem))
10076     {
10077         wereCorrectlyLoaded = true;
10078         this.SetEwallet(ewalletInfo, handleError: true);
10079     }
10080     else if (ewalletInfo != null)
10081     {
10082     }
10083     return wereCorrectlyLoaded;
10084 }

```

```

52 #region Service Methods
53 /// Obtiene una lista de productos ...
54 [13 usages] Juan Murillo +1
61 public EWalletModel GetBalance(EWalletModel purse, bool handleErrorByUser = false, string origin = "")
62 {
63     var purseRequest = this.CreateLoyaltyProgramRequest(purse, isStartSession: true, origin);
64     using (var loyaltyService = new LoyaltyProgramServiceClient())
65     {
66         purseResponse = loyaltyService.ValidateLoyaltyCard(purseRequest);
67     }
68     return ewallet;
69 }
70
127
128

```

Pero antes de se llamar al servicio se genera el request mediante un mapeo en el que ahora el OrdenNumberWallet es depositado en SpecialOrdenNumber

```

202 /// Crea un LoyaltyProgramRequest. ...
203 [3 usages] Juan Murillo +2
208 private LoyaltyProgramRequest CreateLoyaltyProgramRequest(EWalletModel purse, bool isStartSession = false, string origin = "")
209 {
210     return new LoyaltyProgramRequest
211     {
212         CustomerLoyalty = new RTSCustomerAdd
213         {
214             ARTSHeader = new ARTSCommonHeaderType()
215             {
216                 SpecialOrderNumber = purse.OrderNumberWallet,
217             },
218             POSLog = new[]
219             {
220                 new TransactionDomainSpecific
221                 {
222                     CustomerOrderTransaction = new[]
223                     {
224                         new CustomerOrderTransactionBase
225                         {
226                             SpecialOrderNumber = purse.OrderNumberWallet,
227                         }
228                     }
229                 }
230             }
231         }
232     }
233 }
234
235

```


❏ SaleViewModel ❏ TenderManager ❏ Win.Tender(...)LoyaltyProgramServiceClient ❏ TenderServiceClient ❏ PaymentMethodModel ❏ TenderProviderFactory ❏ EwalletTender

❏ TransactionPayment ❏ ILoyaltyProgramService ❏ SadPanelOrdenesViewModel ❏ TransactionServiceClient.partial ❏ EWalletModel ❏ EWalletService ❏ Win.EWallet(...)LoyaltyProgramServiceClient

```
    }
28
29 > /// Valida la existencia de un monedero, (startSession) ...
    (2 users: Juan Murillo)
36 public LoyaltyProgramResponse ValidateLoyaltyCard(LoyaltyProgramRequest cardRequest)
37 {
38     return this.UseService((ILoyaltyProgramService loyaltyProgramService) => loyaltyProgramService.StartSession(cardRequest).Result);
39 }
```

```

94      /// </returns>
95      [ExceptionHandlerService]
96      public async Task<LoyaltyProgramResponse> StartSession(LoyaltyProgramRequest cardRequest)
97      {
98          >
141         if (isSessionStart)
142         {
143             valid = await manager.StartSession(accountInfo, referenceCart: customerOrderTransaction.ShoppingBasketID, customerOrderTransaction.SpecialOrderNumber, referenceNumber: customerOrderTransaction.Reason);
144         }
145     }
146 }

```

```
C# OrigisManager  C# LoyaltyProgramManager x  C# LoyaltyProgramService  C# TenderService  C# TenderManager  C# TenderDao  C# LoyaltySession  C# LoyaltyAccount  C# LoyaltyBenefits

C# FactoryInstance  C# NovartisManager  C# Reference  C# ILoyaltyProgram  C# LoyaltyProgramDao

45 // <param name="referenceNumber">numero de referencia relacionado</param>
46 // <returns>Regresa la información de la tarjeta validada en el objeto <see cref="LoyaltyAccount"/></returns>
47 // (31 usages) @ Juan Murillo +1
48 public async Task<ResultStruct<LoyaltySession>> StartSession(LoyaltyAccount accountInfo, string referenceCard = null, string orderNumber = null, string referenceNumber = null)
49 {
50     var loyalty = FactoryInstance.CreateInstance<IService>(accountInfo.LoyaltyProgramId + this.suffix);
    return await loyalty.StartSession(accountInfo, referenceCard, orderNumber, referenceNumber);
```

```

    public override async Task<ResultStruct>(LoyaltySession) StartSession(LoyaltyAccount accountInfo, string referenceCart = null, string orderId = null, string referenceNumber = null)
    {
        try
        {
            bool isSendOriginActivated = DeterminaEnvioOriginaVenta(accountInfo);

            using (var serviceClient = new Monedero_del_AhorroSoapClient(MDA_ENDPOINT_NAME, checkConnection: false, enableOriginal: true))
            {
                result = await serviceClient.setInicioTransAsync(stajes.accountInfo.Card, sIduse: accountInfo.Branch, accountInfo.Tillid, accountInfo.Payrollid, sOrigen: isSendOriginActivated ? accountInfo.Origin : OriginaSale,
                    sPedidoNumero: string.IsNullOrEmpty(orderNumber) ? "0" : orderNumber, sReferenciaNumero: string.IsNullOrEmpty(referenceNumber) ? "0" : referenceNumber, sCarteraNumero: string.IsNullOrEmpty(referenceCart) ? "0" :
            }
        }
    }
}

```

Solución

Se propone cambiar en el método

FAhorro.POS.Win.Sale.ViewModels.SaleViewModel.LoadDeliveryServiceDetail la siguiente sección de código:

```
// Si no hay numero de carrito buscamos si en la forma de pago de monedero se cuenta con alguno
var fmMonedero = String.IsNullOrEmpty(deliveryService.CurrentEWallet?.ReferenceCart) ?
    deliveryService.FundingMethodsDetail?.Where(r => !string.IsNullOrEmpty(r.Monedero) &&
!string.IsNullOrEmpty(r.ReferenciaTarjeta)).FirstOrDefault()
    : null;
if (fmMonedero != null && deliveryService.CurrentEWallet != null)
{
    deliveryService.CurrentEWallet.ReferenceCart = fmMonedero.ReferenciaTarjeta;
    deliveryService.CurrentEWallet.ReferenceNumberWallet = fmMonedero.PagoFolioAutorizacion;
    deliveryService.CurrentEWallet.OrderNumberWallet = deliveryService.OrdenFolio;
}
```

Por esta otra:

```
var currentWallet = deliveryService.CurrentEWallet;
if (currentWallet != null) {
    // Si no hay numero de carrito buscamos si en la forma de pago de monedero se cuenta con alguno
    if (string.IsNullOrEmpty(currentWallet.ReferenceCart)) {
        var fmMonedero =
            deliveryService.FundingMethodsDetail?.Where(r => !string.IsNullOrEmpty(r.Monedero) &&
!string.IsNullOrEmpty(r.ReferenciaTarjeta))
            .FirstOrDefault();
        if (fmMonedero != null && !string.IsNullOrEmpty(fmMonedero.ReferenciaTarjeta) &&
!string.IsNullOrEmpty(fmMonedero.PagoFolioAutorizacion)) {
            currentWallet.ReferenceCart = fmMonedero.ReferenciaTarjeta;
            currentWallet.ReferenceNumberWallet = fmMonedero.PagoFolioAutorizacion;
        }
    }
    currentWallet.OrderNumberWallet = deliveryService.OrdenFolio;
}
```

que básicamente significa que coloques *OrderNumberWallet* solo condicionado a que *deliveryService.CurrentEWallet* no sea nulo y los datos *ReferenceCart* y *ReferenceNumberWallet* de *deliveryService.CurrentEWallet* se traten de llenar con los datos de la forma de pago si encuentra alguna que los tenga, pero solo si *ReferenceCart* de *deliveryService.CurrentEWallet* no trae dato

Se anexa comparativa del cambio propuesta vs la versión actual

