



|   |                     |                      |                |
|---|---------------------|----------------------|----------------|
|  | Ventas Estatus E v2 |                      |                |
| Autor   | Versión             | Fecha de elaboración | Última edición |
| Cristhian Carrasco<br>Herrerias   | 1.1                 | 12-02-25             | 18-06-25       |

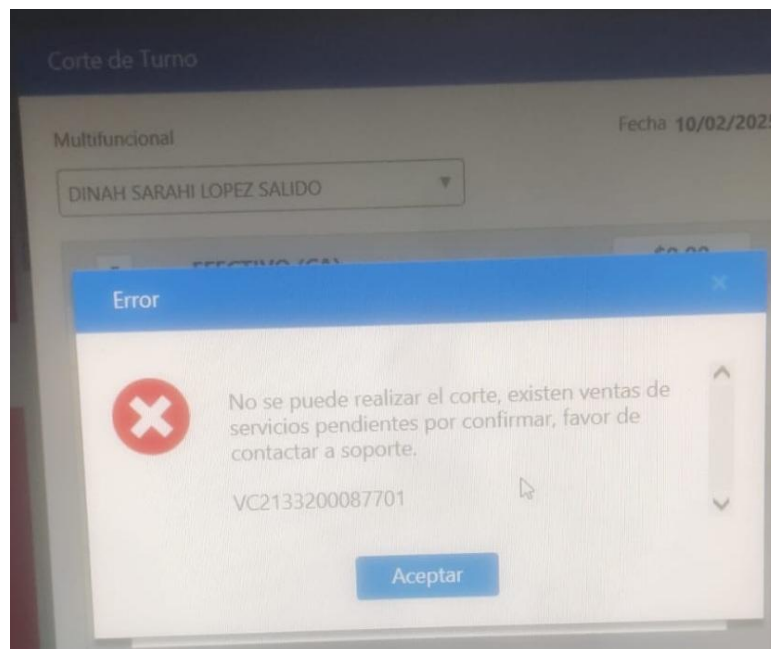
## Control de versiones

| Versión | Fecha    | Autor                              | Descripción                      |
|---------|----------|------------------------------------|----------------------------------|
| 1.0     | 12-02-25 | Hugo<br>Aguilar                    | Elaboración de primer análisis   |
| 1.1     | 18-06-25 | Cristhian<br>Carrasco<br>Herrerias | Causa raíz y posibles soluciones |

|   |                     |                      |                |
|---|---------------------|----------------------|----------------|
|  | Ventas Estatus E v2 |                      |                |
| Autor   | Versión             | Fecha de elaboración | Última edición |
| Cristhian Carrasco<br>Herrerias   | 1.1                 | 12-02-25             | 18-06-25       |


## Descripción general

Se realiza el reporte de una transacción con estatus E, evitando generar el corte de turno de la sucursal:



|                          |                  |
|--------------------------|------------------|
| <b>Sucursal</b>          | 2133             |
| <b>Caja</b>              | 20               |
| <b>Venta</b>             | VC2133200087701  |
| <b>Fecha Transacción</b> | 10/02/2025 15:44 |
| <b>Versión POS</b>       | 10.5.60701.2     |

En dónde se visualiza que la caja tiene instalada la versión (10.5.60701.2) que corrige las transacciones en estatus E

|   |         |                      |                |
|---|---------|----------------------|----------------|
|  |         | Ventas Estatus E v2  |                |
| Autor   | Versión | Fecha de elaboración | Última edición |
| Cristhian Carrasco<br>Herrerias   | 1.1     | 12-02-25             | 18-06-25       |

|                          |   |   |                           |          |
|--------------------------|---|---|---------------------------|----------|
| <input type="checkbox"/> | Soporte N3 - 06 ENEFEVO Referencias 23 dígitos                  |  | N/A 10.5.60701.2          | Liberado |
| <input type="checkbox"/> | Soporte N3 - 07 Transacciones con estatus E                     |  | 10.5.60698.3 10.5.60701.2 | Liberado |
| <input type="checkbox"/> | Soporte N3 - 08 Casos de pieza bonificada pendiente de entregar |  | N/A 10.5.60701.2          | Liberado |
| <input type="checkbox"/> | Soporte N3 - 09 Ventas con cabeceros en estatus diferentes      |  | 10.5.60698.3 10.5.60701.2 | Liberado |
| <input type="checkbox"/> | Soporte N3 - 12 Error de neteo en 3+1 productos                 |  | 10.5.60698.3 10.5.60701.2 | Liberado |


Revisando el histórico de las versiones instaladas en la caja, se confirma que la versión 10.5.60701.2 fue instalada el día 27 de noviembre:

| Remote   |      |        |                           |
|--|------|--------|---------------------------|
| C:\programdata\FAhorro\FARMAX\InstalledVersions\ |      |        |                           |
| Name   | Size | Type   | Date Modified             |
| ..   |      | Folder | 05/24/2023 06:19:29 p. m. |
| 10.5.60702.19                                    |      | Folder | 02/12/2025 08:48:12 a. m. |
| 10.5.60701.2                                     |      | Folder | 11/27/2024 06:42:40 a. m. |
| 10.5.60696.7                                     |      | Folder | 11/04/2024 08:18:26 a. m. |
| 10.5.60695.25                                    |      | Folder | 10/25/2024 02:37:20 a. m. |
| 10.5.60695.24                                    |      | Folder | 10/07/2024 11:58:17 p. m. |
| 10.5.60693.9                                     |      | Folder | 07/04/2024 12:25:11 a. m. |
| 10.5.60692.12                                    |      | Folder | 05/27/2024 12:39:46 a. m. |
| 10.5.60682.6                                     |      | Folder | 03/24/2024 04:43:05 a. m. |
| 10.5.60682.2                                     |      | Folder | 02/29/2024 07:00:48 a. m. |
| 10.5.60681.1                                     |      | Folder | 12/29/2023 01:13:46 a. m. |
| 10.5.60630.2                                     |      | Folder | 11/26/2023 01:21:45 a. m. |
| 10.5.60630.1                                     |      | Folder | 11/13/2023 02:05:36 a. m. |
| 10.5.60630.0                                     |      | Folder | 09/21/2023 06:45:45 a. m. |
| 10.5.60597.4                                     |      | Folder | 09/14/2023 06:59:14 a. m. |
| 10.5.60597.3                                     |      | Folder | 08/31/2023 01:58:42 a. m. |
| 10.5.60597.2                                     |      | Folder | 08/22/2023 01:57:10 a. m. |
| 10.5.60597.0                                     |      | Folder | 08/09/2023 01:11:43 a. m. |
| 10.5.60529.12                                    |      | Folder | 06/19/2023 04:04:15 a. m. |
| 10.5.60529.8                                     |      | Folder | 06/07/2023 01:36:50 a. m. |
| 10.5.60529.4                                     |      | Folder | 05/25/2023 03:46:32 p. m. |
| 10.5.60529.2                                     |      | Folder | 05/24/2023 02:50:49 p. m. |

## Diagnóstico

Tomando en cuenta que la versión 10.5.60701.2 corrige el problema de las transacciones con estatus E, se procede a realizar el análisis de la venta reportada.

Considerando que la primera incidencia reportada, con respecto a transacciones en estatus E, son transacciones de servicios dependientes del servicio de tercero llamado **Tano**.

|   |                     |                      |                |
|---|---------------------|----------------------|----------------|
|  | Ventas Estatus E v2 |                      |                |
| Autor   | Versión             | Fecha de elaboración | Última edición |
| Cristhian Carrasco<br>Herrerias   | 1.1                 | 12-02-25             | 18-06-25       |

Se procede a realizar la siguiente consulta para comparar si la transacción reportada corresponde al tipo de transacciones de Servicio por integrador Tano.

Select

```

        FolioTransaccion
    ,codigoSucursal
    ,numeroCaja
    ,fechaTransaccion
    ,esCobroServicioTercero
    ,status
from trnTransaccionesCab where foliotransaccion = 'V00192100085406'
```

--- La tabla sólo existe en central:

```

select top 1* from venPagoServiciosBitacoraReg (Nolock)
where foliotransaccion = 'V00192100085406'
```

Se procede a realizar consulta sobre una transacción de servicios con tano (V00192100085406) para comparar si la transacción reportada (VC2133200087701) corresponde al tipo de transacciones de servicio por integrador Tano.


La transacción de servicio por integrador Tano, muestra el siguiente resultado:

| Results Messages |                |            |                         |                        |        |  |
|------------------|----------------|------------|-------------------------|------------------------|--------|--|
| FolioTransaccion | codigoSucursal | numeroCaja | fechaTransaccion        | esCobroServicioTercero | status |  |
| V00192100085406  | 19             | 21         | 2024-01-09 14:35:50.490 | 1                      | V      |  |

| idBitacoraServicios | codigoSucursal | numeroCaja | codigoProductoServicio | folioTransaccion | importe | comision | numeroAutorizacion | fechaSolicitud          | fechaRespuesta          | datosSolicitud       | datosRespuesta        |
|---------------------|----------------|------------|------------------------|------------------|---------|----------|--------------------|-------------------------|-------------------------|----------------------|-----------------------|
| 15191960            | 19             | 21         | 1689228615             | V00192100085406  | 49.9999 | 0.0000   | 722531             | 2024-01-09 14:35:48.997 | 2024-01-09 14:35:50.427 | <ArrayOfTanoServi... | <ArrayOfTanoServic... |

En dónde se confirma que la bandera esCobroServicioTercero está activa cuando se trata de una transacción de servicios Tano.



|   |                     |                      |                |
|---|---------------------|----------------------|----------------|
|  | Ventas Estatus E v2 |                      |                |
| Autor   | Versión             | Fecha de elaboración | Última edición |
| Cristhian Carrasco<br>Herrerias   | 1.1                 | 12-02-25             | 18-06-25       |

## Traza de transacción en Farmax


1.- Comienza la transacción, ingresando el sku: 7502276850813

```

10/02/2025 03:44:04 p. m. | Log: TransactionsLog.txt.log | Evento: SaleItemsEvent.

<POSLog xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  MajorVersion="6"
  MinorVersion="0"
  FixVersion="0" xmlns="http://www.nrf-arts.org/IXRetail/namespace/">
  <Transaction
    MajorVersion="6"
    MinorVersion="0"
    FixVersion="0">
    <BusinessUnit>
      <UnitID
        Name="F A TIJUANA, JUAREZ">2133</UnitID>
      </BusinessUnit>
      <WorkstationID
        TypeCode="POS"
        SerialNumber="3812d879-6680-41cf-8626-cbea7e444c3e">OS: Microsoft Windows NT 6.2.9200.0 MachineName: FA2133C20 POSVersion : 1.0.0.0
        IpList:10.95.6.137
        MacAddressList: E0-73-E7-2D-2F-97</WorkstationID>
      <OperatorID
        OperatorName="DINAH SARAHI LOPEZ SALIDO">167733</OperatorID>
      <RetailTransaction>
        <LineItem>
          <Sale>
            <POSIdentity>
              <POSItemID>7502276850813</POSItemID>
            </POSIdentity>
            <Description>G AHORRO TRAMADOL 100MG/10ML SOL 0813</Description>
            <UnitListPrice>781.5600</UnitListPrice>
            <RegularSalesUnitPrice>0</RegularSalesUnitPrice>
            <ActualSalesUnitPrice>42.0000</ActualSalesUnitPrice>
            <Quantity>1</Quantity>
          </Sale>
        </LineItem>
      </RetailTransaction>
      <TillID>20</TillID>
    </Transaction>
  </POSLog>

```


|   |                     |                      |                |
|---|---------------------|----------------------|----------------|
|  | Ventas Estatus E v2 |                      |                |
| Autor   | Versión             | Fecha de elaboración | Última edición |
| Cristhian Carrasco<br>Herrerias   | 1.1                 | 12-02-25             | 18-06-25       |

## 2.- Se procede a las formas de pago:

```

10/02/2025 03:44:05 p. m. | Log: TransactionsLog.txt.log | Evento: TotalizedEvent.

<POSLog xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  MajorVersion="6"
  MinorVersion="0"
  FixVersion="0" xmlns="http://www.nrf-arts.org/IXRetail/namespace/">
  <Transaction
    MajorVersion="6"
    MinorVersion="0"
    FixVersion="0">
    <BusinessUnit>
      <UnitID
        Name="F A TIJUANA, JUAREZ">2133</UnitID>
      </BusinessUnit>
      <WorkstationID
        TypeCode="POS"
        SerialNumber="3812d879-6680-41cf-8626-cbea7e444c3e">OS: Microsoft Windows NT 6.2.9200.0 MachineName: FA2133C20 POSVersion : 1.0.0.0
        IpList:10.95.6.137
        MacAddressList: E0-73-E7-2D-2F-97</WorkstationID>
      <OperatorID
        OperatorName="DINAH SARAHI LOPEZ SALIDO">167733</OperatorID>
      <RetailTransaction>
        <LineItem>
          <Sale>
            <POSIdentity>
              <POSItemID>7502276850813</POSItemID>
            </POSIdentity>
            <Description>G AHORRO TRAMADOL 100MG/10ML SOL 0813</Description>
            <UnitListPrice>781.5600</UnitListPrice>
            <RegularSalesUnitPrice>42.0000</RegularSalesUnitPrice>
            <ActualSalesUnitPrice>42.0000</ActualSalesUnitPrice>
            <Quantity>1</Quantity>
          </Sale>
        </LineItem>
        <Total
          TotalType="TransactionTaxAmount">0.0000</Total>
        <Total
          TotalType="TransactionTotalSavings">0</Total>
        <Total
          TotalType="TransactionSubtotal">42.0000</Total>
        <Total
          TotalType="TransactionNetAmount">42.0000</Total>
        </RetailTransaction>
        <TillID>20</TillID>
        <SerialNumber />
      </Transaction>
    </POSLog>
  
```

|   |                     |                      |                |
|---|---------------------|----------------------|----------------|
|  | Ventas Estatus E v2 |                      |                |
| Autor   | Versión             | Fecha de elaboración | Última edición |
| Cristhian Carrasco<br>Herrerias   | 1.1                 | 12-02-25             | 18-06-25       |

### 3.- Se finaliza la transacción con la forma de pago **VENTA CREDITO (MI)**


```

10/02/2025 03:44:08 p. m. |Log: TransactionsLog.txt.log|Evento: SaleEvent.

<POSLog xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  MajorVersion="6"
  MinorVersion="0"
  FixVersion="0" xmlns="http://www.nrf-arts.org/IXRetail/namespace/">
  <Transaction
    MajorVersion="6"
    MinorVersion="0"
    FixVersion="0">
    <BusinessUnit>
      <UnitID
        Name="F A TIJUANA, JUAREZ">2133</UnitID>
      </BusinessUnit>
      <WorkstationID
        TypeCode="POS"
        SerialNumber="3812d879-6680-41cf-8626-cbea7e444c3e">OS: Microsoft Windows NT 6.2.9200.0 MachineName: FA2133C20 POSVersion : 1.0.0.0
        IpList:10.95.6.137
        MacAddressList: E0-73-E7-2D-2F-97</WorkstationID>
      <OperatorID
        OperatorName="DINAH SARAHI LOPEZ SALIDO">167733</OperatorID>
      <RetailTransaction>
        <LineItem>
          <Sale>
            <ItemType="Product">
              <POSIdentity>
                <POSItemID>7502276850813</POSItemID>
              </POSIdentity>
              <Description>G AHORRO TRAMADOL 100MG/10ML SOL 0813</Description>
              <UnitListPrice>781.5600</UnitListPrice>
              <RegularSalesUnitPrice>42.0000</RegularSalesUnitPrice>
              <ActualSalesUnitPrice>42.0000</ActualSalesUnitPrice>
              <Quantity>1</Quantity>
            </Sale>
          </LineItem>
          <LineItem>
            <Tender>
              <TenderType="VENTA CREDITO (MI)">
                <Amount>42.0000</Amount>
              </Tender>
            </LineItem>
            <LineItem>
              <Donation>
                <Amount>0</Amount>
              </Donation>
            </LineItem>
            <LineItem>
              <Discount>
                <SequenceNumber>0</SequenceNumber>
                <Amount>0</Amount>
              </Discount>
            </LineItem>
            <LineItem>
              <Tax>
                <Amount>0.0000</Amount>
              </Tax>
            </LineItem>
            <LineItem>
              <LoyaltyReward>
                <PointsAwarded>0</PointsAwarded>
              </LoyaltyReward>
            </LineItem>
            <LineItem>
              <Tender>
                <TypeCode="Cashback">
                  <Cashback>0</Cashback>
                </Tender>
              </LineItem>
            </RetailTransaction>
            <TillID>20</TillID>
            <SerialNumber>VC2133200087701</SerialNumber>
          </Transaction>
        </POSLog>

```



|   |         |                      |                |
|---|---------|----------------------|----------------|
|  |         | Ventas Estatus E v2  |                |
| Autor   | Versión | Fecha de elaboración | Última edición |
| Cristhian Carrasco<br>Herrerias   | 1.1     | 12-02-25             | 18-06-25       |

Es decir, el resumen de la traza de la transacción en farmax es el siguiente:

- Comienza la transacción con el artículo 7502276850813 por la cantidad de 1 pieza con precio de \$42
- Se avanza a las formas de pago con el artículo 7502276850813 por la cantidad de 1 pieza con precio de \$42
- Se finaliza la transacción con la forma de pago **VENTA CREDITO (MI)** por la cantidad de \$42. Generando el folio de transacción: **VC2133200087701**.

Por lo cual, se valida la forma de pago de la transacción con la siguiente consulta:

```
select
    c.folioTransaccion
    ,c.fechaTransaccion
    ,b.descripcion
    ,estatusFormaPago
    ,a.importePago
    ,a.syncCreadoFecha
    ,a.syncActualizadoFecha
from genMovimientosCajasdet (nolock) a
inner join genFormasDePagoCat (nolock) b on a.codigoFormaPago = b.codigoFormaPago
inner join trnTransaccionesCab (nolock) c on c.codigoMovimientoCaja =
a.codigoMovimientoCaja
where c.foliotransaccion = 'VC2133200087701'
```


Obteniendo el siguiente resultado:

En dónde se observa que la forma de pago se encuentra en estatus A desde la fecha de creación de la transacción.

| folioTransaccion | fechaTransaccion        | descripcion        | estatusFormaPago | importePago | syncCreadoFecha         | syncActualizadoFecha    |
|------------------|-------------------------|--------------------|------------------|-------------|-------------------------|-------------------------|
| VC2133200087701  | 2025-02-10 15:44:06.550 | VENTA CREDITO (MI) | A                | 42.0000     | 2025-02-10 15:44:06.587 | 2025-02-10 15:44:06.947 |

En el archivo log de la caja, se visualiza el siguiente flujo de la venta:

```
[2025-02-10 15:44:06:075],FARMAX_TILL_SERVER,SALE,INFO,,Venta :: Metodo:AddSale FolioVenta:VC2133200087701 Estado:N
[2025-02-10 15:44:06:079],FARMAX_TILL_SERVER,SALE,INFO,,Donativo :: Metodo:UpdateSaleHeader FolioVenta:VC2133200087701 Estado:N Monto: IdBeneficiencia: FolioDonativo:
[2025-02-10 15:44:06:079],FARMAX_TILL_SERVER,SALE,INFO,,Venta :: Metodo:UpdateSaleHeader FolioVenta:VC2133200087701 Estado:N
[2025-02-10 15:44:06:084],FARMAX_TILL_SERVER,SALE,INFO,,Venta :: Metodo:UpdateSaleAfterWriteSaleDetails FolioVenta:VC2133200087701 Estado:N
[2025-02-10 15:44:06:088],FARMAX_TILL_SERVER,SALE,INFO,,Donativo :: Metodo:AddOrUpdateSale FolioVenta:VC2133200087701 Estado:N Monto: IdBeneficiencia: FolioDonativo:
[2025-02-10 15:44:06:088],FARMAX_TILL_SERVER,SALE,INFO,,Venta :: Metodo:AddOrUpdateSale FolioVenta:VC2133200087701 Estado:N
[2025-02-10 15:44:06:092],FARMAX_TILL_SERVER,SALE,INFO,,Donativo :: Metodo:UpdateSaleHeader FolioVenta:VC2133200087701 Estado:N Monto: IdBeneficiencia: FolioDonativo:
[2025-02-10 15:44:06:092],FARMAX_TILL_SERVER,SALE,INFO,,Venta :: Metodo:UpdateSaleHeader FolioVenta:VC2133200087701 Estado:N
[2025-02-10 15:44:06:094],FARMAX_TILL_SERVER,SALE,INFO,,Venta :: Metodo:UpdateSaleAfterWriteSaleDetails FolioVenta:VC2133200087701 Estado:N
[2025-02-10 15:44:06:591],FARMAX_TILL_SERVER,SALE,INFO,,Donativo :: Metodo:UpdateSaleHeader FolioVenta:VC2133200087701 Estado:N Monto: IdBeneficiencia: FolioDonativo:
[2025-02-10 15:44:06:591],FARMAX_TILL_SERVER,SALE,INFO,,Venta :: Metodo:UpdateSaleHeader FolioVenta:VC2133200087701 Estado:N
[2025-02-10 15:44:06:595],FARMAX_TILL_SERVER,SALE,INFO,,Donativo :: Metodo:UpdateSaleHeader FolioVenta:VC2133200087701 Estado:N Monto: IdBeneficiencia: FolioDonativo:
[2025-02-10 15:44:06:596],FARMAX_TILL_SERVER,SALE,INFO,,Venta :: Metodo:UpdateSaleHeader FolioVenta:VC2133200087701 Estado:N
[2025-02-10 15:44:06:598],FARMAX_TILL_SERVER,SALE,INFO,,Venta :: Metodo:UpdateSaleAfterWriteSaleDetails FolioVenta:VC2133200087701 Estado:N
[2025-02-10 15:44:06:916],FARMAX_TILL_SERVER,SALE,INFO,,Donativo :: Metodo:AddOrUpdateSale FolioVenta:VC2133200087701 Estado:N Monto: IdBeneficiencia: FolioDonativo:
[2025-02-10 15:44:06:916],FARMAX_TILL_SERVER,SALE,INFO,,Venta :: Metodo:AddOrUpdateSale FolioVenta:VC2133200087701 Estado:N
[2025-02-10 15:44:06:989],FARMAX_TILL_SERVER,SALE,INFO,,Donativo :: Metodo:UpdateSaleHeader FolioVenta:VC2133200087701 Estado:E Monto: IdBeneficiencia: FolioDonativo:
[2025-02-10 15:44:06:989],FARMAX_TILL_SERVER,SALE,INFO,,Venta :: Metodo:UpdateSaleHeader FolioVenta:VC2133200087701 Estado:E
[2025-02-10 15:44:06:992],FARMAX_TILL_SERVER,SALE,INFO,,Venta :: Metodo:UpdateSaleAfterWriteSaleDetails FolioVenta:VC2133200087701 Estado:E
[2025-02-10 22:04:02:964],FARMAX_TILL_SERVER,SALE,INFO,,Venta :: Metodo:GetTransactionsByStatus FolioVenta:VC2133200087701 Estado:E
[2025-02-10 22:04:38:688],FARMAX_TILL_SERVER,SALE,INFO,,Venta :: Metodo:GetTransactionsByStatus FolioVenta:VC2133200087701 Estado:E
[2025-02-10 22:59:00:642],FARMAX_TILL_SERVER,SALE,INFO,,Venta :: Metodo:GetTransactionsByStatus FolioVenta:VC2133200087701 Estado:E
[2025-02-10 23:04:22:225],FARMAX_TILL_SERVER,SALE,INFO,,Venta :: Metodo:GetTransactionsByStatus FolioVenta:VC2133200087701 Estado:E
```

|   |         |                      |                |
|---|---------|----------------------|----------------|
|  |         | Ventas Estatus E v2  |                |
| Autor   | Versión | Fecha de elaboración | Última edición |
| Cristhian Carrasco<br>Herrerias   | 1.1     | 12-02-25             | 18-06-25       |

En dónde se confirma que la transacción se queda con estados E, desde las **15:44:06:992**  
En el método: **UpdateSaleAfterWriteSaleDetails**

Por lo cual la hora es congruente con la fecha de transacción de cabecero de la transacción:

| FolioTransaccion | codigoSucursal | numeroCaja | fechaTransaccion        | esCobroServicioTercero | syncCreadoFecha         |
|------------------|----------------|------------|-------------------------|------------------------|-------------------------|
| VC2133200087701  | 2133           | 20         | 2025-02-10 15:44:06.550 | 0                      | 2025-02-10 15:44:06.070 |

## Causa del problema

- Una de las causas de que las ventas a crédito se actualicen con estatus E y estas no sean un servicio de tercero es por la validación que se encuentra el método **CompletePaymentSale** en **SaleManager.cs**

En la siguiente imagen de código (línea 1457) se muestran dos condicionales: Si es un cobro por servicios de tercero o si el tipo de venta es crédito y es un pago pendiente.

Código:

```


1456 // Si es un servicio de tercero se pone en estatus "E" = EnProceso
1457 if (saleTransaction.Sale.esCobroServicioTercero || (saleTransaction.Sale.tipoVenta == SaleType.Credit && transactionStatusBd == SaleStatus.PaymentsPending))
1458 {
1459     // Se pone como finalizada pero pendiente de confirmar.
1460     saleTransaction.Sale.status = SaleStatus.PendingToConfirm;
1461 }
1462 else
1463 {
1464     // Realiza la actualización de status de 'Pendiente de Pago' a Vigente y se marca con estaActivo = true
1465     saleTransaction.Sale.status = SaleStatus.Active;
1466 }

```

En la siguiente imagen se muestra como el tipo de venta reportada (VC2133200087701) el valor de si es cobro de un servicio a tercero es falso (0), el tipo de venta es Crédito (R) y en su momento cuando se estaba procesando el estatus de esta fue Pendiente de Pago (R). Esto quiere decir que paso la validación y su estatus cambio a En Proceso de Confirmación (E)

Registro en BD de la venta VC2133200087701:

| esCobroServicioTercero | tipoVenta | idTransaccion                        | codigoMovimientoCaja                 | folioTransaccion | f |
|------------------------|-----------|--------------------------------------|--------------------------------------|------------------|---|
| 0                      | R         | 5b96bee9-08e8-ef11-a143-e073e72d2f97 | 5c96bee9-08e8-ef11-a143-e073e72d2f97 | VC2133200087701  | r |

|   |                     |                      |                |
|---|---------------------|----------------------|----------------|
|  | Ventas Estatus E v2 |                      |                |
| Autor   | Versión             | Fecha de elaboración | Última edición |
| Cristhian Carrasco<br>Herrerias   | 1.1                 | 12-02-25             | 18-06-25       |

- Dentro del método que valida si existen ventas con estatus E se agregó un proceso el cual antes de validar, actualiza las ventas en estatus E a V, solo que estos dos procesos tienen diferentes filtros lo cual es incongruente ya que el **FIX** no se aplicaría en su totalidad.

En la siguiente imagen se muestra en la línea 977 el método que actualiza las ventas con estatus E (**PendingMovementsToConfirm**) y en la línea 987 el método que obtiene las ventas con estatus E (**GetTransactions**).

#### ValidatePendingSaleService en ViewModelCutBase.cs

```

975 using (var transactionClient = new TransactionServiceClient())
976 {
977     var response = transactionClient.PendingMovementsToConfirm();
978     if (response != null && response.SoldTrnsactions != null && response.SoldTrnsactions.Any())
979     {
980         foreach (var transaction in response.SoldTrnsactions)
981         {
982             NotificationManager.ShowMessage(Messages.TitleWarning, string.Format(Messages.PendingSalesServicePayments, transaction), MessageIcon.Warning);
983         }
984     }
985
986     var result = transactionClient.GetTransactions(filter).Where(s => s.TillId.Equals(tillNumber)).ToList();
987     return result.Select(r => r.TransactionNumber).ToList();
988 }
989 
```

El método que actualiza (**PendingMovementsToConfirm**), filtra el usuario actual y actualiza solo esas ventas con estatus E.


#### PendingMovementsToConfirm en TransactionServiceClient.cs (Store)

```

/// <summary>
/// Procesa las transacciones pendientes de confirmar
/// </summary>
/// <returns>Movimientos aplicados</returns>
2 referencias | -
public PendingToConfirmSalesResponse PendingMovementsToConfirm()
{
    PendingToConfirmSalesRequest request = new PendingToConfirmSalesRequest();
    if (AppContext.Instance.UserInfo == null)
    {
        return null;
    }

    request.UserId = AppContext.Instance.UserInfo.Id;
    return this.UseService((ITransactionService transactionService) => transactionService.ProcessPendingToConfirmSales(request));
}

```

|   |         |                      |                |  |
|---|---------|----------------------|----------------|--|
|  |         | Ventas Estatus E v2  |                |  |
| Autor   | Versión | Fecha de elaboración | Última edición |  |
| Cristhian Carrasco<br>Herrerias   | 1.1     | 12-02-25             | 18-06-25       |  |

El método que valida (**GetTransactions**), tiene diferentes filtros como: sucursal, caja actual y el estatus E, como se muestra en las siguientes imágenes.

### ValidatePendingSaleService en ViewModelCutBase.cs

```

965         SaleFilterModel filter = new SaleFilterModel();
966         filter.FilterType = FilterType.ByStatus.ToString();
967         filter.Purse = string.Empty;
968         filter.BranchOfficeLocator = PosAppContext.Instance.BusinessUnit.BusinessUnitId;
969         filter.IsTrainingMode = PosAppContext.Instance.IsTrainingMode;
970         filter.IsPendingSalePayment = false;
971         filter.Status = SaleStatus.PendingToConfirm;
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987         var result = transactionClient.GetTransactions(filter).Where(s => s.TillId.Equals(tillNumber)).ToList();

```

- El método que actualiza las ventas en estatus E a V filtrada por el usuario actual contiene una validación la cual genera una excepción por esa razón no actualiza las ventas.

En la siguiente imagen se muestra como antes de actualizar la venta a estatus V o C debe de pasar por una validación (línea 6900)


ITransactionService→ProcessPendingToConfirmSales→ProcessPendingToConfirmTransactions→ProcessSale

### ProcessSale en SaleManager.cs

```

6896     /// <summary>
6897     /// Metodo que procesa las ventas pendientes
6898     /// </summary>
6899     /// <param name="serviceClient">Cliente para el servicio/</param>
6900     /// <param name="transaction">Objeto de la venta/</param>
6901     /// <param name="result">Lista para agregar las transacciones procesadas/</param>
6902     /// </param>
6903     /// </summary>
6904     public void ProcessSale(Ahorro POS.BusinessLogic.Sale.ServicePayments.ServicePaymentsClient serviceClient, tmTransaccionesCab transaction, PendingToConfirmSalesResponse result)
6905     {
6906         try
6907         {
6908             var responseService = serviceClient.GetServiceAuthorization(transaction.numeroCaja, transaction.codigoSucursal, transaction.folioTransaccion, transaction.tmTransaccionesDet.FirstOrDefault().codigoProducto);
6909             if (responseService != null && !string.IsNullOrEmpty(responseService.Folio))
6910             {
6911                 result.SoldTransactions.Add(transaction.folioTransaccion);
6912                 this.UpdateStatus(transaction.idTransaccion, SaleStatus.Active);
6913             }
6914             else
6915             {
6916                 result.CanceledTransactions.Add(transaction.folioTransaccion);
6917                 this.UpdateStatus(transaction.idTransaccion, SaleStatus.Cancel);
6918             }
6919             if (transaction.codigoMovimientoCaja.HasValue) // TODO: HNA: Monitorear este cambio
6920             {
6921                 var manager = new TenderManager();
6922                 manager.CancelTenderTypesList(new List<Guid> { transaction.codigoMovimientoCaja.Value });
6923             }
6924         }
6925         catch (Exception)
6926         {
6927             return;
6928         }
6929     }
6930 }

```

|   |                     |                      |                |
|---|---------------------|----------------------|----------------|
|  | Ventas Estatus E v2 |                      |                |
| Autor   | Versión             | Fecha de elaboración | Última edición |
| Cristhian Carrasco<br>Herrerias   | 1.1                 | 12-02-25             | 18-06-25       |

El método **GetServiceAuthorization** se ejecuta en central como se muestra en las siguientes imágenes


### GetServiceAuthorization en ServicePaymentsClient.cs (Central)

```

20 public class ServicePaymentsClient : RemoteProxyCentral<IServicePayments>
{
    /// <summary>
    /// Metodo con el cual se obtiene los parametros del servicio
    /// </summary>
    /// <param name="tillId">Id de la caja</param>
    /// <param name="businnesUnit">id de la sucursal</param>
    /// <param name="transactionId">identificador de transaccion</param>
    /// <param name="productCode">Codigo de producto.</param>
    /// <returns>
    /// parametros encontrados
    /// </returns>
    1 referencia | -
    public ServicePaymentResponse GetServiceAuthorization(int tillId, int businnesUnit, string transactionId, int productCode)
    {
        var request = new ServicePaymentRequest
        {
            BusinessUnitId = businnesUnit,
            TillId = tillId,
            ProductCode = productCode,
            TypeServicePayments = new ServicePaymentsHeaderType
            {
                BusinessUnit = businnesUnit,
                FolioTransaction = transactionId
            }
        };

        return this.MapTransactionResponse(this.UseService((IServicePayments s) => s.GetServiceAuthorization(request)));
    }
}

```

|   |                     |                      |                |
|---|---------------------|----------------------|----------------|
|  | Ventas Estatus E v2 |                      |                |
| Autor   | Versión             | Fecha de elaboración | Última edición |
| Cristhian Carrasco<br>Herrerias   | 1.1                 | 12-02-25             | 18-06-25       |

Cuando se ejecuta el método **GetServiceAutorization** en central obtiene el proveedor de servicio en base a un código de producto del detalle de la transacción como se muestra en la línea 153 de la siguiente imagen.

Posterior en la línea 156 crea una instancia en base al nombre del proveedor de servicios.

Podemos ver que el método tiene un encabezado (línea 147) el cual su función es guardar las excepciones en el LOG **FARMAX\_WSC\_CORE**.

```

142  /// <summary>
143  /// Consulta la respuesta de autorización de un servicio.
144  /// </summary>
145  /// <param name="parameters">Datos de la transacción a consultar.</param>
146  /// <returns>Regresa respuesta del servicio</returns>
147  [ExceptionHandlerService]
148  2 referencias | -
149  public ServicePaymentResponse GetServiceAutorization(ServicePaymentRequest parameters)
150  {
151      var managerServicePayments = new ServicePaymentsManager();
152      var transactionLog = managerServicePayments.GetDataLogTransaction(parameters.TypeServicePayments.FolioTransaction);
153      var provider = managerServicePayments.GetServiceProvider(parameters.ProductCode);
154      if (transactionLog != null)
155      {
156          var process = CreatorFactory.CreateInstance(provider.Name);
157          return process.GetTransactionResponse(parameters);
158      }
159      return ServiceResponseFactory.GetErrorResponseNoAuthFound(provider.ID, provider.Name);
160  }
161
162
163

```


El método que obtiene del proveedor de servicios desde la base de datos realiza una relación de tablas (**genProductosServicio** con **genProveedorServicioCat**) la cual es filtrada por el código de producto como se muestra en la siguiente imagen.

### GetServiceProvider en ServicePaymentsDao.cs

```

/// <summary>
/// Consulta el proveedor apartir del codigo de servicio.
/// </summary>
/// <param name="productCode">Codigo del producto.</param>
/// <returns>Proveedor que proporciona el servicio</returns>
1 referencia | -
public genProveedorServicioCat GetServiceProvider(int productCode)
{
    return this.DataContext.genProductosServicio.Include("genProveedorServicioCat").Single(s => s.codigoProducto == productCode).genProveedorServicioCat;
}

```

|   |         |                      |                |
|---|---------|----------------------|----------------|
|  |         | Ventas Estatus E v2  |                |
| Autor   | Versión | Fecha de elaboración | Última edición |
| Cristhian Carrasco<br>Herrerias   | 1.1     | 12-02-25             | 18-06-25       |

Tomando en cuenta el ejemplo de la venta **VC2133200087701** el código de producto del detalle es **134877834**, el cual no se encuentra en **genProductosServicio** y genera la excepción, por lo cual no actualiza el estatus de E a V. Esta excepción (Sequence contains no elements) se guarda en el log **FARMAX\_WSC\_CORE** en el servidor **Central**, como se muestra en la siguiente imagen.

```
[2025-02-10 23:18:07.203] FARMAX_WSC_CORE.ERROR [BACKGROUNDEXECUTIONTASK] EXCEPTION: Method: GetServiceAuthorization - Exception: Sequence contains no elements Sequence contains no elements at System.Linq.Enumerable.Single[TSource](IEnumerable<TSource> source) at System.Data.Entity.Core.Objects.ELinq.ObjectQueryProvider.<GetElements>_3[TResult](IEnumerable<TSource> sequence) at System.Data.Entity.Core.Objects.ELinq.ObjectQueryProvider.ExecuteSingle[TResult](IEnumerable<TSource> sequence, Expression query) at System.Data.Entity.Core.Objects.ELinq.ObjectQueryProvider.System.Linq.IQueryProvider.Execute[TResult](Expression expression) at System.Data.Entity.Internal.Linq.ObjectQueryProvider.Execute[TResult](Expression expression) at System.Linq.Queryable.Single[TSource](IEnumerable<TSource> source, Expression expression) at Faborro.POS.Data.ServicesPayments.DAO.ServicePaymentsDAO.GetServicePayment(Int32 productId) in C:\Builds-0r\projects\qW47b2D4\Farmacias\delahorro\coe\Farmax\pos-or-pos\Farmax\Src\POS\Data.ServicesPayments\DAO\ServicePaymentsDAO.cs:line 372 at Faborro.POS.BusinessLogic.ServicesPayments.ServicePaymentManager.GetServicePayment(Int32 productId) in C:\Builds-0r\projects\qW47b2D4\Farmacias\delahorro\coe\Farmax\pos-or-pos\Farmax\Src\POS\BusinessLogic.ServicesPayments\ServicePaymentManager.cs:line 585 at Faborro.POS.Services.ServicesPayments.ServicesPayments.GetServiceAuthorization(ServicePaymentRequest parameters) in C:\Builds-0r\projects\qW47b2D4\Farmacias\delahorro\coe\Farmax\pos-or-pos\Farmax\Src\POS\Services.ServicesPayments\ServicePayments.cs:line 158
```

Corroborando lo que se muestra en la imagen anterior podemos visualizar los logs del **POS** y **TILL** los siguientes procesos:

- Proceso de Cierre (Archivo **20250210\_FARMAX\_POS\_CORE.XLOG**).
- Proceso que actualiza las ventas de estatus E a V (Archivo **20250210\_FARMAX\_TILL\_SERVER\_SERVICECORE.XLOG**) el cual genera la excepción ya mencionada pero ahora desde el **TILL**.
- Proceso que obtiene las ventas con estatus E (Archivo **20250210\_FARMAX\_TILL\_SERVER\_SALE.XLOG**) el cual contiene la venta **VC2133200087701** ya que nunca se actualizo.

| Artefacto | Archivo                                      | Fecha                   | Información   |
|-----------|--|-------------------------|---|
| POS       | 20250210_FARMAX_POS_CORE.XLOG                | 2025-02-10 22:59:00:013 | FARMAX_POS_CORE.ERROR [AUTOMATICPROCESSES] Proceso de cierre disparado a las: 10/02/2025 10:59:00 p. m.   |
| TILL      | 20250210_FARMAX_TILL_SERVER_SERVICECORE.XLOG | 2025-02-10 22:59:00:630 | FARMAX_TILL_SERVER_SERVICECORE.ERROR [GETSERVICEAUTHORIZATION] RemoteProxyCentral.ServicePayments Excepción en el servicio. Excepción:Sequence contains no elements StackTrace: |
| TILL      | 20250210_FARMAX_TILL_SERVER_SALE.XLOG        | 2025-02-10 22:59:00:662 | FARMAX_TILL_SERVER_SALE.INFO Venta - Método: GetTransactionsByStatus Falta venta VC2133200087701 Estado: E  |


*Nota: El horario del log **Central** con el de **Till** y **Pos** están desfasados.*

En la siguiente imagen podemos visualizar los procesos anteriores en código:

- Proceso de Cierre (Ejecuta el método **ValidatePendingSaleService**).
- Proceso que actualiza las ventas de estatus E a V (Ejecuta el método **PendingMovementsToConfirm**).
- Proceso que obtiene las ventas con estatus E (Ejecuta el método **GetTransactions**).

```
955 // Summary
956 // Validar si existen ventas de servicio con estatus E (En espera de cambio de estatus a V)
957 // Summary
958 // Param name="cashRegisterId"=Número de la caja.</param>
959 // Returns
960 // True en caso de que no existan ventas pendientes / Falso en caso de que existan ventas pendientes y no se puedan borrar en automatico
961 // Returns
962 // Returns
963
964 protected List<string> ValidatePendingSaleService(int? cashRegisterId = null)
965 {
966     bool isValid = true;
967     SaleFilterModel filter = new SaleFilterModel();
968     filter.FilterType = FilterType.ByStatus.ToString();
969     filter.Parse = String.Empty;
970     filter.BranchOfficeLocator = PosAppContext.Instance.BusinessUnit.BusinessUnitId;
971     filter.IsTrainingMode = PosAppContext.Instance.IsTrainingMode;
972     filter.IsPendingSalePayment = false;
973     filter.Status = SaleStatus.PendingToConfirm;
974
975     int tillNumber = cashRegisterId ?? PosAppContext.Instance.CashRegisterId;
976
977     using (var transactionClient = new TransactionServiceClient())
978     {
979         var response = transactionClient.PendingMovementsToConfirm();
980
981         if (response != null && response.SoldTrnsactions != null && response.SoldTrnsactions.Any())
982         {
983             foreach (var transaction in response.SoldTrnsactions)
984             {
985                 NotificationManager.ShowMessage(Messages.TitleWarning, string.Format(Messages.PendingSalesServicePayments, transaction), MessageBoxIcon.Warning);
986             }
987
988             var result = transactionClient.GetTransactions(filter).Where(s => s.TillId.Equals(tillNumber)).ToList();
989             return result.Select(r => r.TransactionNumber).ToList();
990         }
991     }
992 }
```



|   |                     |                      |                |
|---|---------------------|----------------------|----------------|
|  | Ventas Estatus E v2 |                      |                |
| Autor   | Versión             | Fecha de elaboración | Última edición |
| Cristhian Carrasco<br>Herrerias   | 1.1                 | 12-02-25             | 18-06-25       |

## Posible solución

1. Corroborar si realmente el tipo de venta a crédito y que no sea un servicio de terceros se deba actualizar a estatus E (**CompletePaymentSale** en **SaleManager.cs**), esto implicaría quitar el OR lógico condicional de la línea 1457 o en esta misma line agregar una validación si los detalles de venta pertenecen a un proveedor de servicio y así se evitarían los puntos 2 y 3.

```

1456 // Si es un servicio de tercero se pone en estatus "E" = EnProceso
1457 if (saleTransaction.Sale.esCobroServicioTercero || (saleTransaction.Sale.tipoVenta == SaleType.Credit && transactionStatusBd == SaleStatus.PaymentsPending))
1458 {
1459     // Se pone como finalizada pero pendiente de confirmar;
1460     saleTransaction.Sale.status = SaleStatus.PendingToConfirm;
1461 }
1462 else
1463 {
1464     // Realiza la actualización de status de 'Pendiente de Pago' a Vigente y se marca con estaActivo = true
1465     saleTransaction.Sale.status = SaleStatus.Active;
1466 }

```


2. Homologar los filtros de **PendingMovementsToConfirm** con los de **GetTransactions**, esto implicaría cambiar los contratos de **ITransactionService**→**ProcessPendingToConfirmSales** y los parámetros de sus métodos consecutivos.

```

975 using (var transactionClient = new TransactionServiceClient())
976 {
977     var response = transactionClient.PendingMovementsToConfirm();
978     if (response != null && response.SoldTrnsactions != null && response.SoldTrnsactions.Any())
979     {
980         foreach (var transaction in response.SoldTrnsactions)
981         {
982             NotificationManager.ShowMessage(Messages.TitleWarning, string.Format(Messages.PendingSalesServicePayments, transaction), MessageIcon.Warning);
983         }
984     }
985 }
986
987 var result = transactionClient.GetTransactions(filter).Where(s => s.TillId.Equals(tillNumber)).ToList();
988 return result.Select(r => r.TransactionNumber).ToList();
989 }

```



|   |                     |                      |                |
|---|---------------------|----------------------|----------------|
|  | Ventas Estatus E v2 |                      |                |
| Autor   | Versión             | Fecha de elaboración | Última edición |
| Cristhian Carrasco<br>Herrerias   | 1.1                 | 12-02-25             | 18-06-25       |

### 3. En este punto hay dos mejoras

- Filtrar correctamente al obtener el código de servicio del detalle de la venta ya que solo obtiene lo el primer elemento encontrado (línea 6900), esto implicaría modificar validaciones dentro del método **ProcessSale**.

```

6899  /// <summary>
6900  /// Metodo que procesa las ventas pendientes
6901  /// </summary>
6902  /// <param name="serviceClient">Cliente para el servicio/</param>
6903  /// <param name="transaction">Objeto de la venta/</param>
6904  /// <param name="result">Lista para agregar las transacciones procesadas/</param>
6905  /// </param>
6906  public void ProcessSale(Ahorro POS BusinessLogic Sale.ServicePayments.ServicePaymentsClient serviceClient, trnTransaccionesCab transaction, PendingToConfirmSalesResponse result)
6907  {
6908  {
6909      var responseService = serviceClient.GetServiceAuthorization(transaction.numeroCaja, transaction.codigoSucursal, transaction.folioTransaccion, transaction.trnTransaccionesDet.FirstOrDefault().codigoProducto);
6910      if (responseService != null && !string.IsNullOrEmpty(responseService.Folio))
6911      {
6912          result.SoldTransactions.Add(transaction.folioTransaccion);
6913          this.UpdateStatus(transaction.idTransaccion, SaleStatus.Active);
6914      }
6915      else
6916      {
6917          result.CanceledTransactions.Add(transaction.folioTransaccion);
6918          this.UpdateStatus(transaction.idTransaccion, SaleStatus.Cancel);
6919          if (transaction.codigoMovimientoCaja.HasValue) // TODO: HMA: Monitorear este cambio
6920          {
6921              var manager = new TenderManager();
6922              manager.CancelTenderTypesList(new List<Guid> { transaction.codigoMovimientoCaja.Value });
6923          }
6924      }
6925  }
6926  catch (Exception)
6927  {
6928      return;
6929  }
6930  }
6931  }

```

Obtención de código de producto línea 6900


```
transaction.trnTransaccionesDet.FirstOrDefault().codigoProducto);
```

- Modificar la consulta **GetServiceProvider** que se encuentra en **ServicePaymentsDao.cs** la cual genera una excepción por obtener la tabla **genProveedorServicioCat** relacionada a **genProductosServicio** ya que al querer obtenerla filtrando por el código del producto por el método **Single** sin validar. Esta modificación implicaría agregar logs más específicos.

```

6932  /// <summary>
6933  /// Consulta el proveedor apartir del codigo de servicio.
6934  /// </summary>
6935  /// <param name="productCode">Codigo del producto.</param>
6936  /// <returns>Proveedor que proporciona el servicio/</returns>
6937  /// </param>
6938  public genProveedorServicioCat GetServiceProvider(int productCode)
6939  {
6940      return this.DataContext.genProductosServicio.Include("genProveedorServicioCat").Single(s => s.codigoProducto == productCode).genProveedorServicioCat;
6941  }

```

|   |                     |                      |                |
|---|---------------------|----------------------|----------------|
|  | Ventas Estatus E v2 |                      |                |
| Autor   | Versión             | Fecha de elaboración | Última edición |
| Cristhian Carrasco<br>Herrerias   | 1.1                 | 12-02-25             | 18-06-25       |

## Conclusión

El primer causa raíz con respecto a transacciones que se quedan en estatus E, corresponde al tipo de transacciones de servicios de terceros gestionado por el integrador **Tano**.

Mientras que la reciente transacción reportada con estatus E, corresponde al tipo de transacción **“VENTA CREDITO (MI)”**.

Por lo cual no tiene relación con el fix liberado en la versión productiva: 10.5.60701.2

En base a la documentación de la primera versión de este documento y análisis de la incidencia de [estatus E del integrador Tano](#) se encontraron las causas raíz y posibles soluciones de la versión 1.1 de este documento.