Crear Comprobante CNV & Asociar Comodato

IT Projects

Table of Contents

1.1. Management Summary 8

1.2. Glossary 8

1.3. Architecture goals and restrictions 8

1.3.1 Aplicabilidad de Desarrollo Seguro de software. 8

1.3.2 Vulnerabilidad conocida en versiones anteriores. 9

1.4. System Requirements View 9

1.4.1. Use Cases 9

1.4.2. Use Case Diagram 9

1.4.3. Components Sequential Flow 9

2. SOLUTION ARCHITECTURE 22

2.1 Vista Macro 22

2.4. Development View 23

**Blueprint Level 3 - Software Architecture**

**ECommerce**

Version 2021 0.1

* [Version Control](#scroll-bookmark-2)
* [Approval History](#scroll-bookmark-3)
* [Software Architecture](#scroll-bookmark-4)
* [1. SOLUTION DESCRIPTION](#scroll-bookmark-5)
  + [1.1. Management Summary](#scroll-bookmark-6)
  + [1.2. Glossary](#scroll-bookmark-7)
  + [1.3. Architecture goals and restrictions](#scroll-bookmark-8)
    - [1.3.1 Aplicabilidad de Desarrollo Seguro de software.](#scroll-bookmark-9)
    - [1.3.2 Vulnerabilidad conocida en versiones anteriores.](#scroll-bookmark-10)
  + [1.4. System Requirements View](#scroll-bookmark-11)
    - [1.4.1. Use Cases](#scroll-bookmark-12)
    - [1.4.2. Use Case Diagram](#scroll-bookmark-13)
    - [1.4.3. Components Sequential Flow](#scroll-bookmark-14)
    - [1.4.4. Presentation and Application delivery requirements](#scroll-bookmark-15)
      * [1.4.4.1. For each interface (i.e. SMS, ussd, web, IVR, Application Interface, others)](#scroll-bookmark-16)
    - [1.4.5. Reporting](#scroll-bookmark-17)
  + [1.5. Monitoring Requirements](#scroll-bookmark-18)
  + [1.6. Quality Attributes](#scroll-bookmark-19)
  + [1.7. Security Attributes](#scroll-bookmark-20)
  + [1.8. Revenue Assurance Requirements](#scroll-bookmark-21)
* [2. SOLUTION ARCHITECTURE](#scroll-bookmark-22)
  + [2.1 Vista Macro](#scroll-bookmark-23)
* [2.2 Process View](#scroll-bookmark-24)
  + [2.3. Non Functional Requirements](#scroll-bookmark-25)
    - [2.3.1. Revenue Assurance Requirements](#scroll-bookmark-26)
  + [2.4. Development View](#scroll-bookmark-27)
  + [2.5. Data View (optional)](#scroll-bookmark-28)
  + [2.6. Integration Requirements](#scroll-bookmark-29)
  + [2.7. Fault Management](#scroll-bookmark-30)
* [3. Physical View](#scroll-bookmark-31)
  + [3.1. Infrastructure base requirements](#scroll-bookmark-32)
  + [3.2. Technology Description](#scroll-bookmark-33)
  + [3.3. Physical Diagram](#scroll-bookmark-34)
    - [3.3.1. Production Environment](#scroll-bookmark-35)
    - [3.3.2. Contingency Environment](#scroll-bookmark-36)
    - [3.3.3. Environment Quality](#scroll-bookmark-37)
    - [3.3.4. Development Environment](#scroll-bookmark-38)
    - [3.3.5. General Network Diagram](#scroll-bookmark-39)
* [4. Country Specific Information](#scroll-bookmark-40)
  + [4.1. Local Regulatory Requirements](#scroll-bookmark-41)
  + [4.2. Language Requirements](#scroll-bookmark-42)
  + [4.3. User Experience](#scroll-bookmark-43)
  + [4.4. Service Assurance Handover Specifics](#scroll-bookmark-44)
* [5. Strategic Quality Specifications](#scroll-bookmark-45)

# Version Control

| **Date** | **Version** | **Description** | **Autor** |
| --- | --- | --- | --- |
| 10/09/2022 | 1.0 | API para crear comprobante CNV y API para asociar comodato a una cuenta. | Hugo Luis Silva Sierra |
| 12/09/2022 | 1.1 | Revisión de Marco Peña. | Marco Peña |
| 28/09/2022 | 1.2 | Se completan las imágenes de las pantallas de asociación de comodato. | Hugo Silva |
| 23/11/2022 | 1.3 | Se colocan las firmas estándar para cada API. | Hugo Silva |
| 25/11/2022 | 1.4 | Se agregó el valor del comprobante a la entrada de la API asociar comodato. | Michael Rodriguez |

# Approval History

| **Version** | **Revised by** | **Role** | **Date** | **Approved/Rejected** | **Observations** |
| --- | --- | --- | --- | --- | --- |
| <x.y> | <name> | <role> | <dd/mm/yyyy> | <Approved/Rejected> | <comments> |
|  |  |  |  |  |  |

# Software Architecture

# 1. SOLUTION DESCRIPTION

## 1.1. Management Summary

El presente BPL3 corresponde al diseño de 2 APIs: una API para crear comprobante CNV, en la que el resultado es el identificador del comprobante CNV creado (además de otros datos) y una segunda API para cambiar la situación del comodato del cliente, en la que el resultado es una confirmación de la operación de asociación del comodato.

## 1.2. Glossary

| **Term** | **Definition** |
| --- | --- |
| **Comprobante CNV** | Documento en el que se reserva el equipo al cliente y lo habilita para que pueda pagarlo. |
| **Comodato** | Equipo mobile del cliente. |

## 1.3. Architecture goals and restrictions

La meta principal del análisis de la arquitectura es mostrar aspectos principales que influyen en la etapa de desarrollo

● Metas

- Tomar en cuenta las buenas prácticas de desarrollo para codificación java, y las buenas prácticas de consultas SQL.

- Se debe asegurar el buen manejo de las conexiones hacia a la base de datos, es decir asegurar que se cierren las conexiones.

- Las pruebas, capacitación, manuales deben ser documentados y versionados.

● Restricciones

- Debe poder se expuesto sobre JFuse.

- El adaptador debe ser construido con tecnología API Rest.

### 1.3.1 Aplicabilidad de Desarrollo Seguro de software.

|  | Si | No |
| --- | --- | --- |
| Aplicación de cara a internet |  | X |
| Aplicación de cara a clientes finales. |  | X |
| Procesará información de identificación personal (PII) |  | X |
| Procesará información de autenticación |  | X |
| procesará otra información de carácter confidencial (Ej. Financiera, Información médica, otras) |  | X |

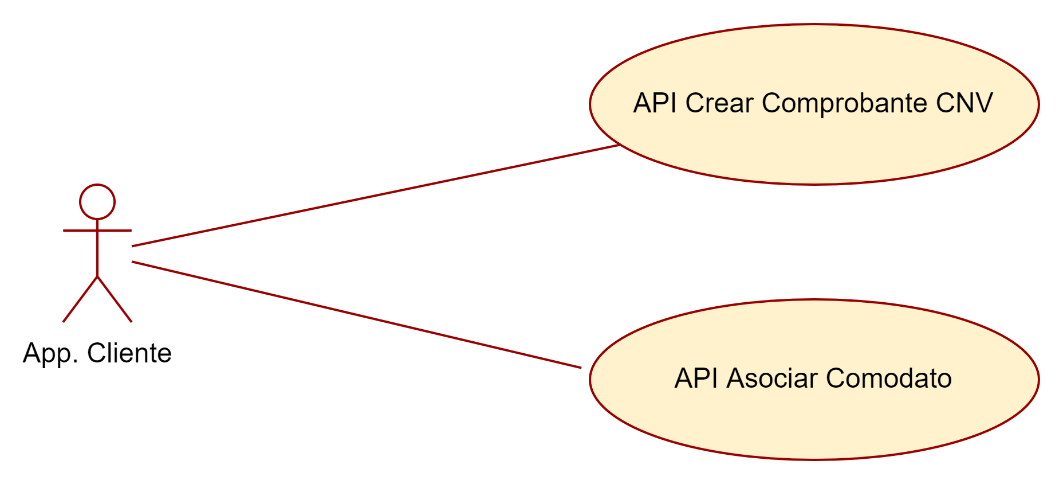
### 1.3.2 Vulnerabilidad conocida en versiones anteriores.

| **Vulnerabilidades conocidas** | **Criticidad** | **Acciones de remodelación** |
| --- | --- | --- |
| No existen vulnerabilidades conocidas |  |  |
|  |  |  |
|  |  |  |

## 1.4. System Requirements View

### 1.4.1. Use Cases

### 1.4.2. Use Case Diagram



### 1.4.3. Components Sequential Flow

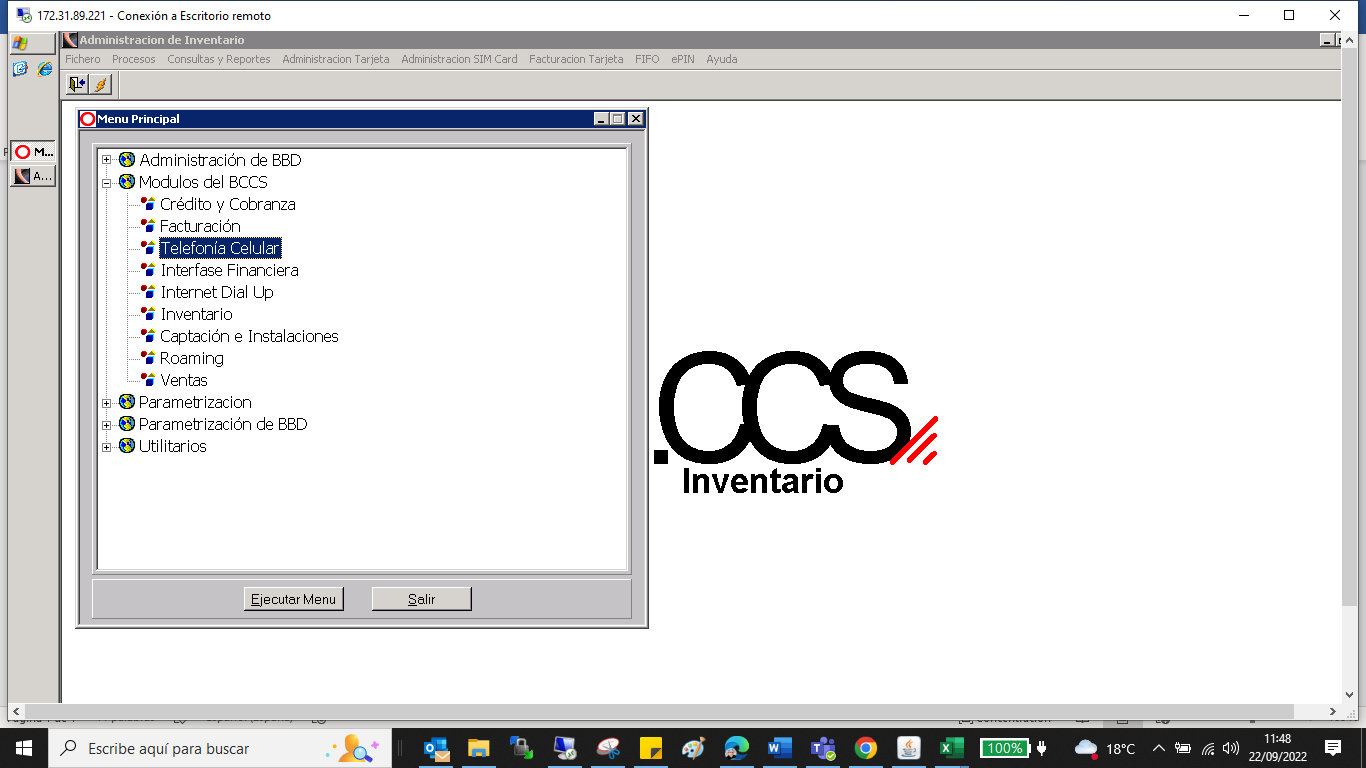
| **Use Case Name** | **Functionality Description** | **Use Case Confluence Route** |
| --- | --- | --- |
| **API Crear Comprobane CNV.** | Este caso de uso describe el funcionamiento de una API, la cual permite crear un comprobante CNV, en base a los datos del cliente. El resultado describe los datos relevantes del comprobante creado.  Verbo: POST  URL: /v1/tigo/bo/generate-invoice-cnv  Body:  {  "requestedStartDate": "2022-11-16T13:22:32.870Z",  "isMandatory": **true**,  "channel": [  {  "name": "BCCS"  }  ],  "characteristic": [  {  "name": "codigoBodega",  "valueType": "string",  "value": "154"  },  {  "name": "numeroDocumento",  "valueType": "string",  "value": "6254665"  },  {  "name": "tipoDocumento",  "valueType": "string",  "value": "CI"  },  {  "name": "cliente",  "valueType": "string",  "value": "1500416"  },  {  "name": "msisdn",  "valueType": "string",  "value": "75002636"  },  {  "name": "numeroContrato",  "valueType": "string",  "value": "9492242"  },  {  "name": "usuario",  "valueType": "string",  "value": "ECOMMERCE"  },  {  "name": "tipoVenta",  "valueType": "string",  "value": "POSTPAGO"  },  {  "name": "motivoRegistracion",  "valueType": "string",  "value": "COR"  },  {  "name": "subTipoCuenta",  "valueType": "string",  "value": "GSM"  },  {  "name": "identificadorIndividual",  "valueType": "string",  "value": "373725698215873"  },  {  "name": "codigoDescuento",  "valueType": "string",  "value": "7537"  },  {  "name": "precioOriginal",  "valueType": "decimal",  "value": "170"  },  {  "name": "montoDescuento",  "valueType": "decimal",  "value": "169.28"  },  {  "name": "codigoMoneda",  "valueType": "string",  "value": "DOL"  }  ],  "relatedEntity": [  {  "id": "71100690256\_355513316959181",  "href": "http://{host}/api/v1/devices/?stockId={stockId}&sku={sku}",  "name": "API getDevice info",  "role": "device",  "@type": "RelatedEntity"  }  ],  "@type": "TaskFlow",  "@baseType": "TaskFlow"  }  Descripción de los campos de entrada:   |  |  |  | | --- | --- | --- | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  |   Response Caso satisfactorio  Response:  {  "completionDate": "2022-11-16T13:22:32.870Z",  "completionMethod": "automatic",  "isMandatory": **true**,  "state": "completed",  "channel": [  {  "name": "BCCS"  }  ],  "characteristic": [  {  "name": "invoiceType",  "valueType": "string",  "value": "ondemand"  },  {  "name": "invoiceId",  "valueType": "string",  "value": "CNV-CC-8905"  },  {  "name": "billingPeriod",  "valueType": "string",  "value": "ondemand"  },  {  "name": "billingDate",  "valueType": "decimal",  "value": "2022-07-28"  },  {  "name": "invoiceAmount",  "valueType": "string",  "value": "162.66"  },  {  "name": "dueAmount",  "valueType": "string",  "value": "162.66"  },  {  "name": "currencyId",  "valueType": "string",  "value": "BOL"  },  {  "name": "hasPayment",  "valueType": "boolean",  "value": "false"  }  ],  "relatedEntity": [  {  "id": "CNV-CC-8905",  "href": "https://{host}/cxf/api/v1/billing/InvoiceHeader/{crmSystem}/{idType}/{id}/invoices?invoiceId=CNV-CC-8905",  "name": "API get Invoice header",  "role": "invoice",  "@type": "RelatedEntity"  }  ],  "@type": "TaskFlow",  "@baseType": "TaskFlow"  }  Parámetros de salida:   |  |  |  | | --- | --- | --- | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  |   Response de Caso fallido.  {  "errorCode": 400,  "errorType": "BadRequest",  "code": "400",  "errorType": "Error en mensaje de entrada",  } |  |
|  | Respuestas de la API:   |  |  |  | | --- | --- | --- | | **http status** | **Estado** | **Descripción** | | 200 | OK | Ok / Error | | 403 | Forbidden | Error de accesos | | 500 | Internal Server Error | Error interno | | 400 | Bad Request | Campos input del Servicio REST obligatorios en blanco o nulo | | 404 | Not Found | Si la línea no existe | | 401 | Unauthorized | del Servicio REST Credenciales Invalidas | |  |

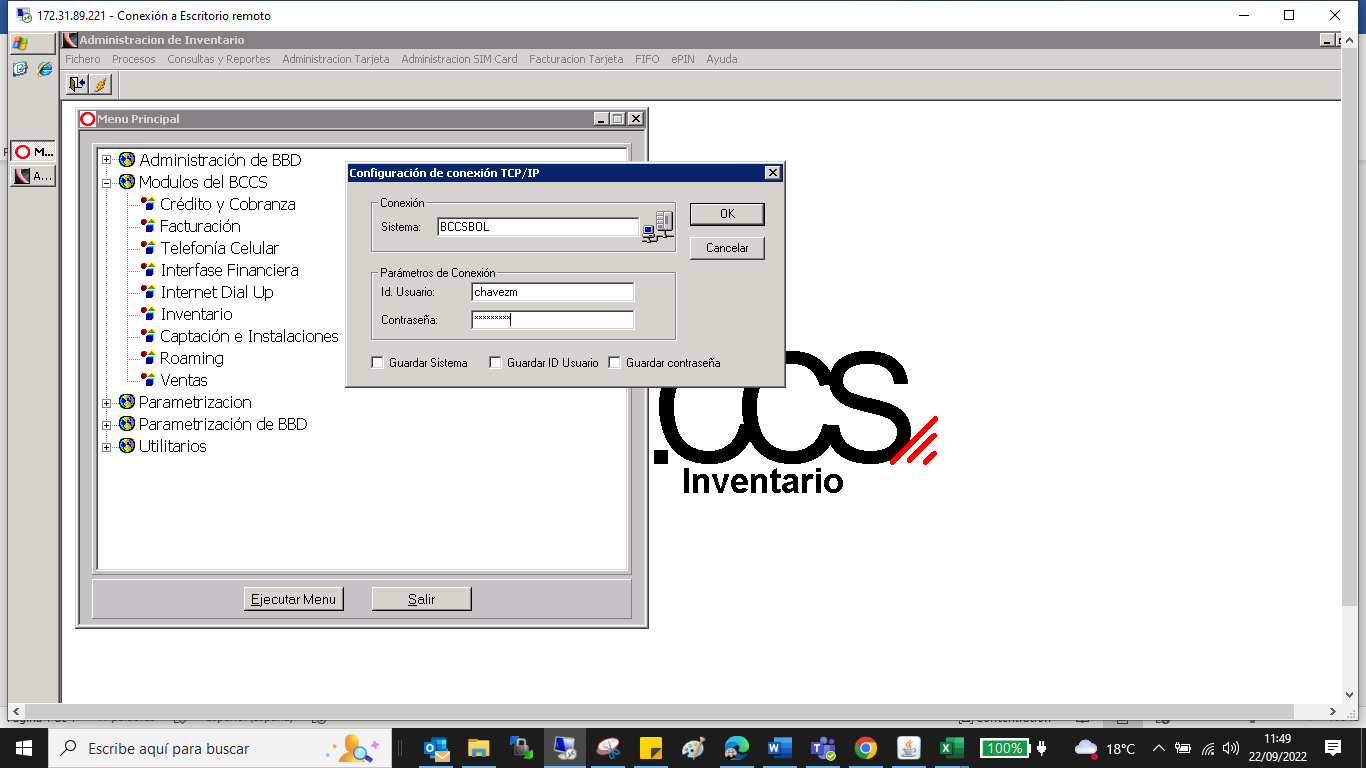
| **Use Case Name** | **Functionality Description** | **Use Case Confluence Route** |
| --- | --- | --- |
| **API Asociar comodato a cuenta** | Este caso de uso es una API Rest en la que se asocia el IMEI de un dispositivo mobile a la cuenta de un cliente en BCCS.  Verbo: POST  URL: /v1/tigo/bo/comodato  Body  {  "requestedStartDate": "2022-11-16T13:22:32.870Z",  "isMandatory": **true**,  "channel": [  {  "name": "BCCS"  }  ],  "characteristic": [  {  "name": "numeroDocumento",  "valueType": "string",  "value": "6254665"  },  {  "name": "tipoDocumento",  "valueType": "string",  "value": "CI"  },  {  "name": "clienteId",  "valueType": "string",  "value": "1500416"  },  {  "name": "contrato",  "valueType": "string",  "value": "9492242"  },  {  "name": "phoneNumber",  "valueType": "string",  "value": "75002636"  },  {  "name": "usuario",  "valueType": "string",  "value": "ECOMMERCE"  },  {  "name": "identificadorIndividual",  "valueType": "string",  "value": "373725698215873"  },  {  "name": "comprobanteCNV",  "valueType": "string",  "value": "CNV-CC-8905"  }  ],  "relatedEntity": [  {  "id": "71100690256\_355513316959181",  "href": "http://{host}/api/v1/devices/?stockId={stockId}&sku={sku}",  "name": "API getDevice info",  "role": "device",  "@type": "RelatedEntity"  }  ],  "@type": "TaskFlow",  "@baseType": "TaskFlow"  }  Descripción de los campos de entrada:   |  |  |  | | --- | --- | --- | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  |   **Response ejemplo de proceso satisfactorio (Http Status Code – 200):**  {      "codigo" : "OK",      "mensaje" : "Proceso satisfactorio"  }  **Response ejemplo de proceso NO satisfactorio (Http Status Code – 200):**  {      "codigo" : "ERROR",      "mensaje" : "Mensaje detallado del motivo del proceso no satisfactorio."  }  Properties:  Ejemplo de parámetro de salida:  {  "requestedStartDate": "2022-11-16T13:22:32.870Z",  "isMandatory": **true**,  "channel": [  {  "name": "BCCS"  }  ],  "characteristic": [  {  "name": "numeroDocumento",  "valueType": "string",  "value": "6254665"  },  {  "name": "tipoDocumento",  "valueType": "string",  "value": "CI"  },  {  "name": "clienteId",  "valueType": "string",  "value": "1500416"  },  {  "name": "contrato",  "valueType": "string",  "value": "9492242"  },  {  "name": "phoneNumber",  "valueType": "string",  "value": "75002636"  },  {  "name": "identificadorIndividual",  "valueType": "string",  "value": "373725698215873"  }  ],  "relatedEntity": [  {  "id": "71100690256\_355513316959181",  "href": "http://{host}/api/v1/devices/?stockId={stockId}&sku={sku}",  "name": "API getDevice info",  "role": "device",  "@type": "RelatedEntity"  }  ],  "@type": "TaskFlow",  "@baseType": "TaskFlow"  } |  |
|  | Respuestas de la API:   |  |  |  | | --- | --- | --- | | **http status** | **Estado** | **Descripción** | | 200 | OK | Ok / Error | | 404 | Not Found | Ningún registro encontrado. | | 403 | Forbidden | Error de accesos | | 500 | Internal Server Error | Error interno | | 400 | Bad Request | Campos input del Servicio REST obligatorios en blanco o nulo | | 401 | Unauthorized | del Servicio REST Credenciales Invalidas | |  |
| **Requerimiento no funcional** | Se requiere que los siguientes puntos sean configurados en el archivo de configuración del servicio.   * Parámetros por defecto para los inputs y response de cada método. * Scripts de base de datos. * Conexión de bases de datos. * Parametrización de timeout de conexiones. * Documentación de las APIs (Manual Implementación, Operaciones) * Logs (Ver RNF Logs). |  |

| **Clasificación** | **Descripción** |
| --- | --- |
| RNF: Logs | Para todos los procesos se deben dejar los logs en el siguiente formato.  [TIPO] [FECHA HORA] [USUARIO/CUENTA/CANAL] [ACCION] [TIEMPO]  TIPO: INFO/DEBUG/ERROR…. FECHA HORA: mostrar la fecha y hora en formato dd/MM/yyyy HH:mm:ss API*:* Para los procesos sms y ussd se debe imprimir la cuenta que está realizando el flujo, para la web y el ws se tiene que imprimir el usuario y la cuenta, y para el ws debe imprimir el canal. ej: ws. INFO 23/04/2015 10:00:00 **pepito 77390320 sms** iniciando flujo.  ACCION: mensaje de la actividad que esté realizando. TIEMPO: Para las conexiónes a interfaces se tiene que mostrar el tiempo en ms que tarda en responder al momento de invocar el ws, esto para cualquier método a utilizar del ws.  Para los casos de errores, las excepciones deben elevarse hasta la clase principal e imprimir en el log el stacktrace de la excepción. |
| RNF: Implementación | El proceso requerido debe estar implementado considerando los siguientes puntos:   * Tomar en cuenta las buenas prácticas de desarrollo para codificación java. * Se debe asegurar el buen manejo de las conexiones hacia servicios externos. Se debe asegurar el correcto cierre de las conexiones. * Las pruebas, capacitación, manuales deben ser documentados y versionados. * La solución deberá ser desplegada en un contenedor Docker orquestado por Kubernetes |
| RNF: Configuración | Se requiere que los siguientes puntos sean configurados en el archivo de configuración del servicio.   * Parámetros por defecto para los inputs y response de cada método. * Scripts de base de datos. * Conexión de bases de datos. * Parametrización de timeout de conexiones. * Logs (Ver RNF Logs). |
| RNF: Autenticación (Authorization Basic) | El servicio debe poder solicitar credenciales de autenticación para el uso de este.  El Autorization debe ser de tipo *Basic Auth.* |
| RNF: Documentación | Se deberá proveer con la documentación correspondiente.  Documentación de las APIs (Manual Implementación, Operaciones) |

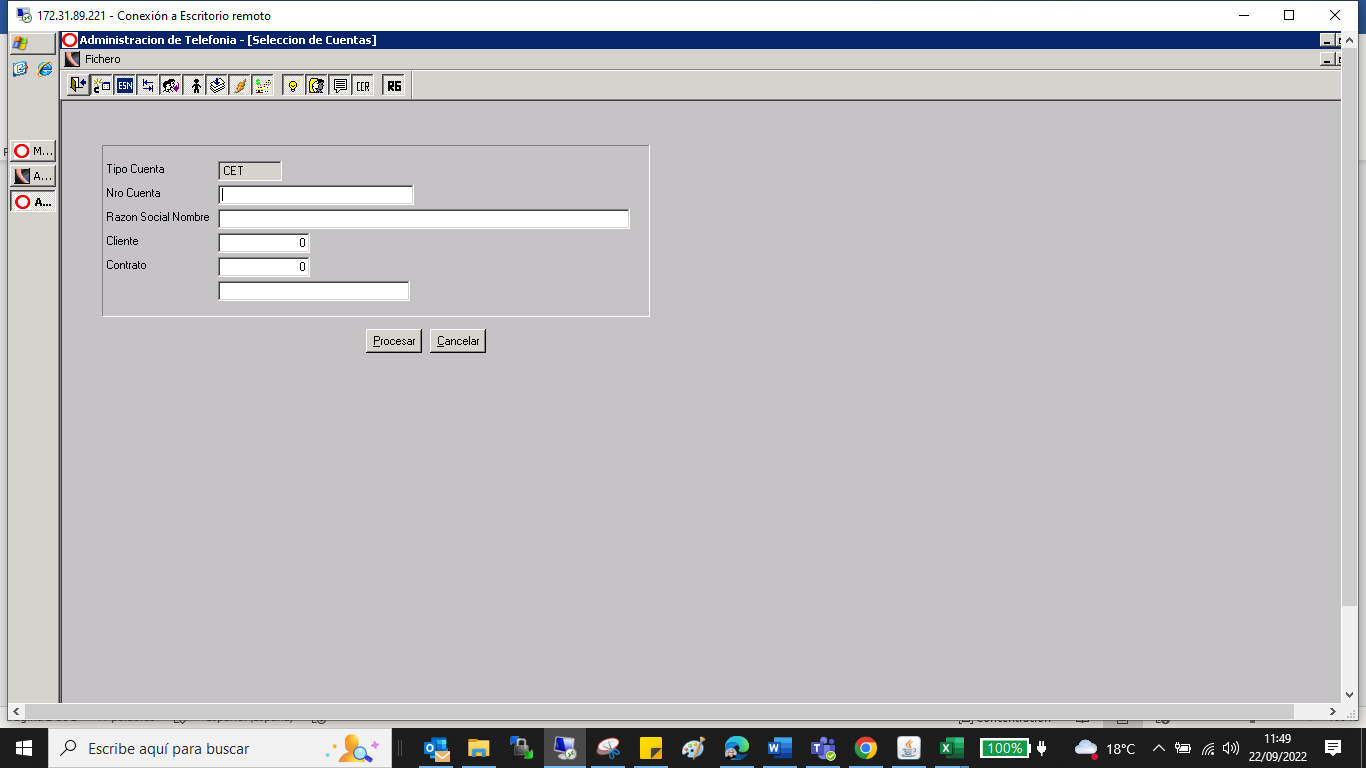
**Anexos.**

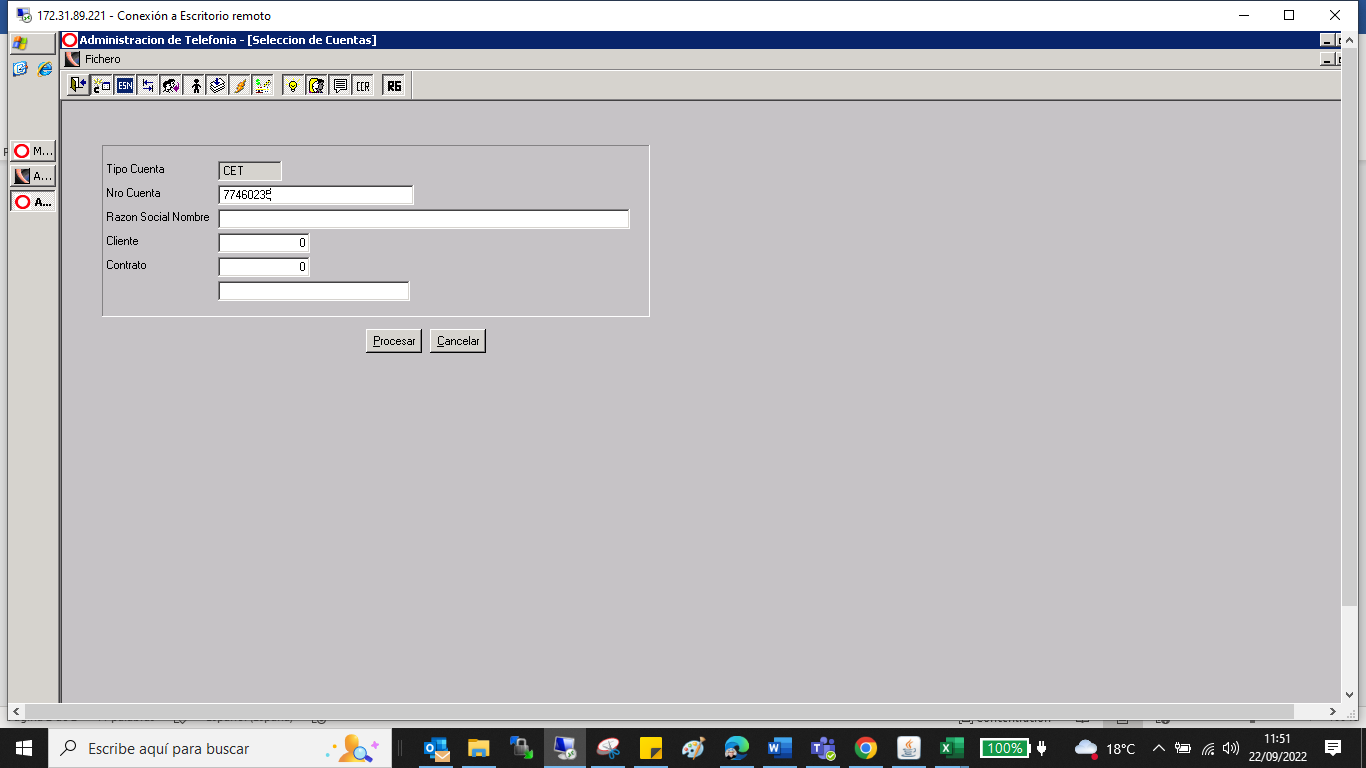
**Anexo – Cambio de Situación del Equipo En Propio y Cambiar a COR**

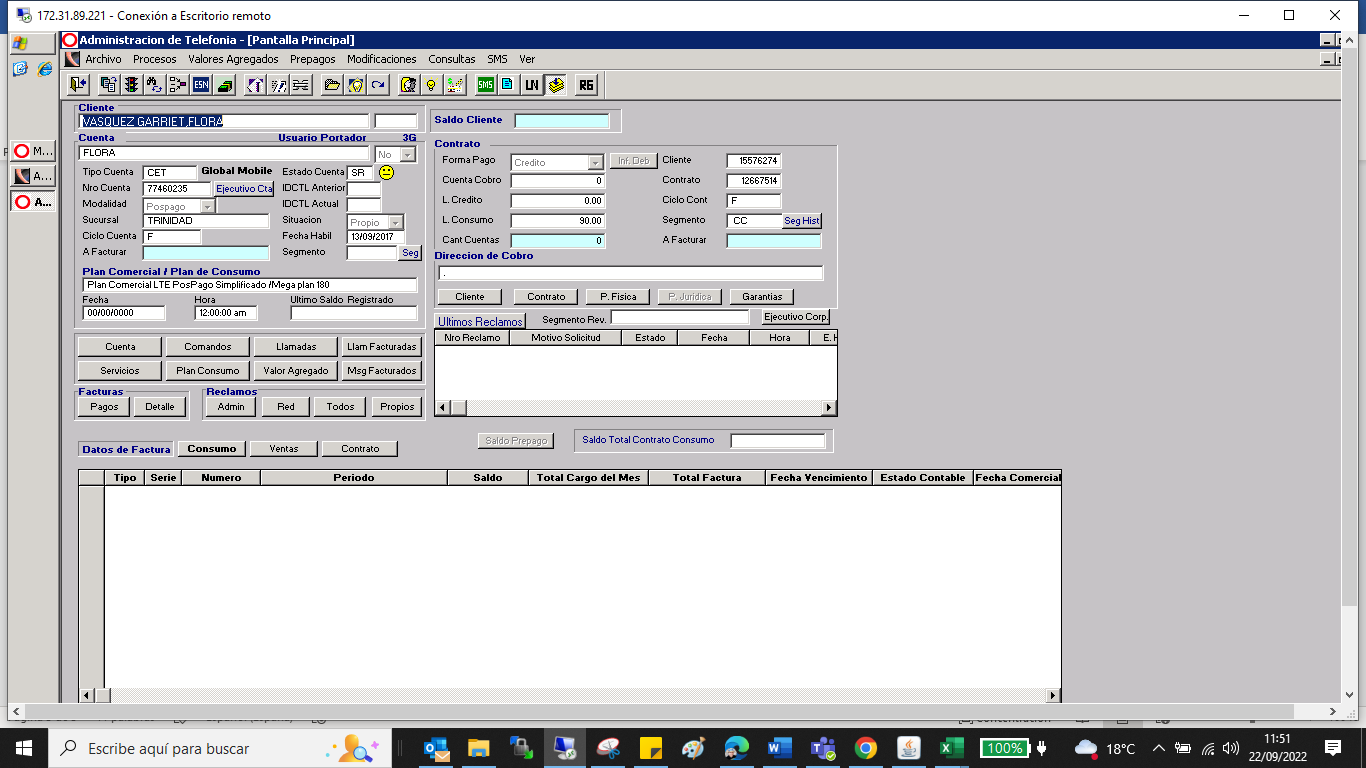


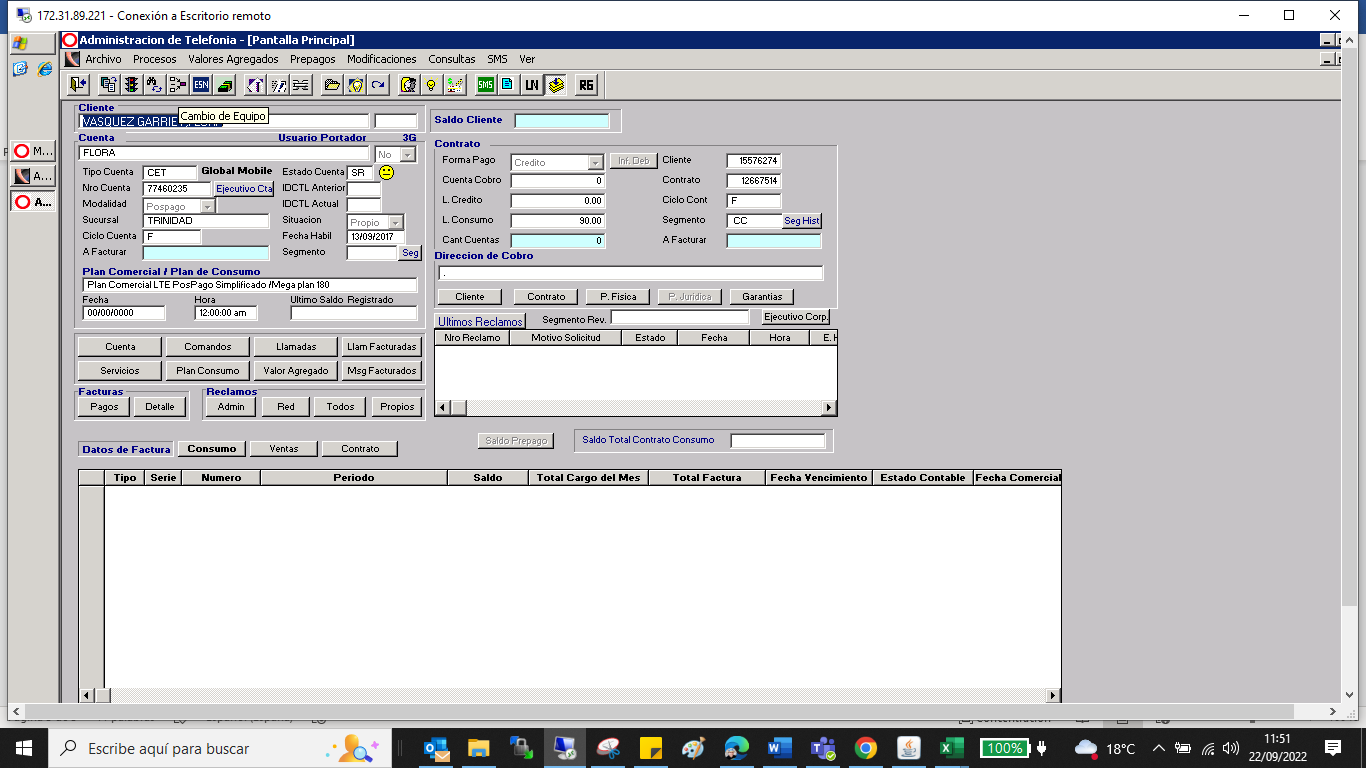


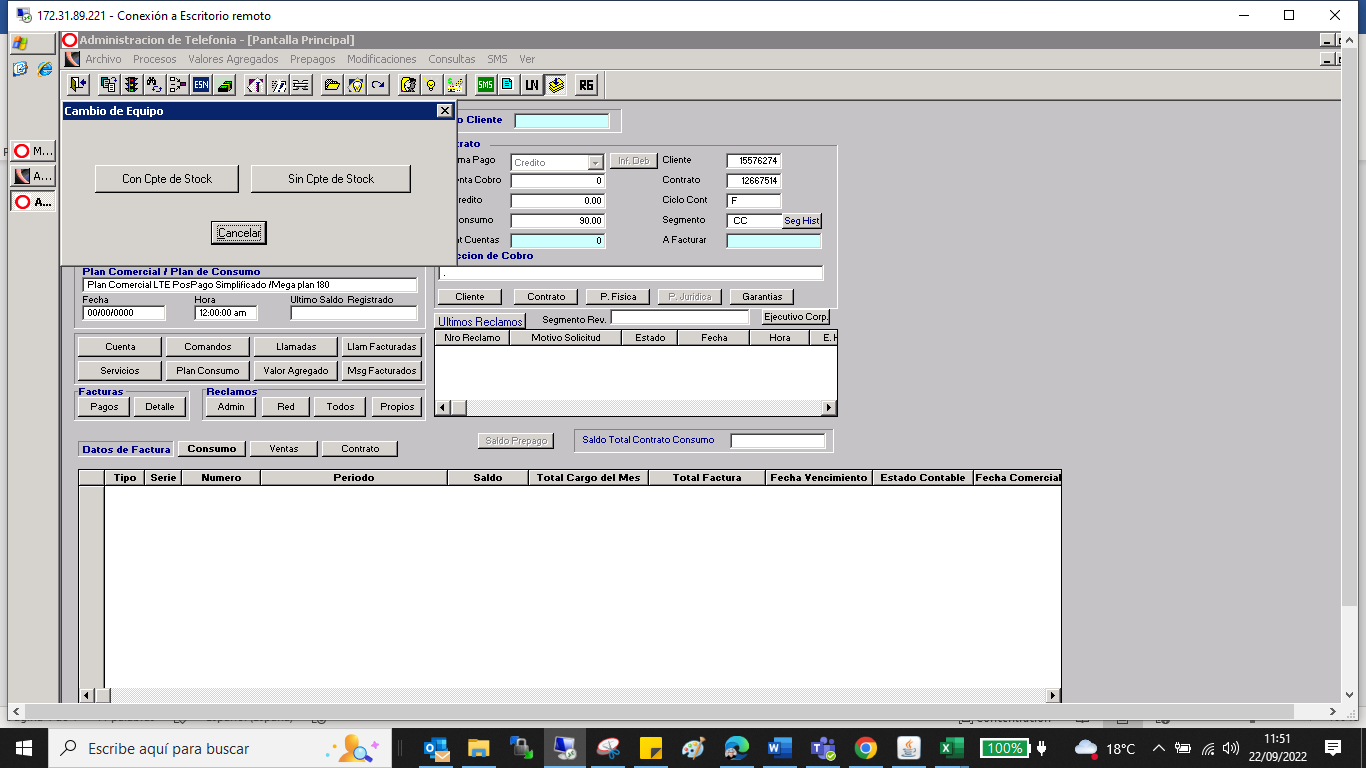


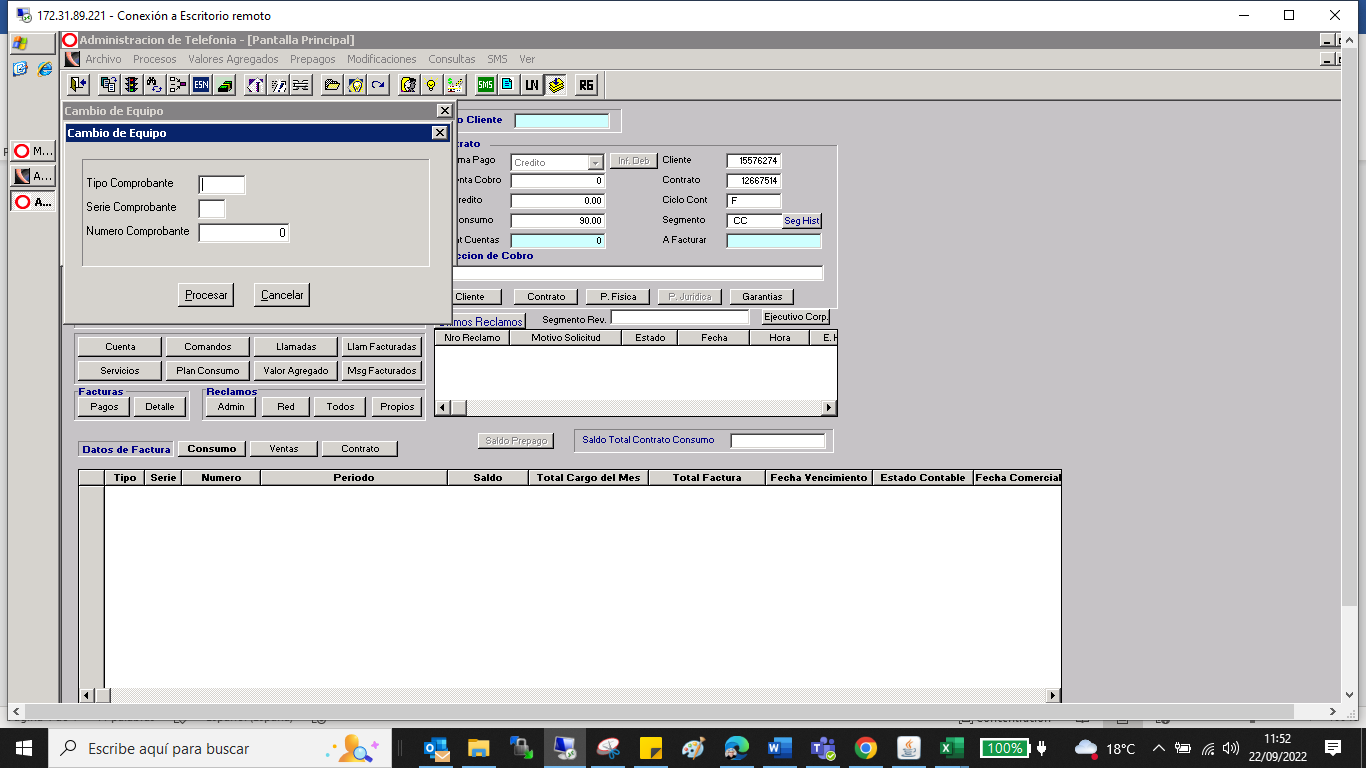


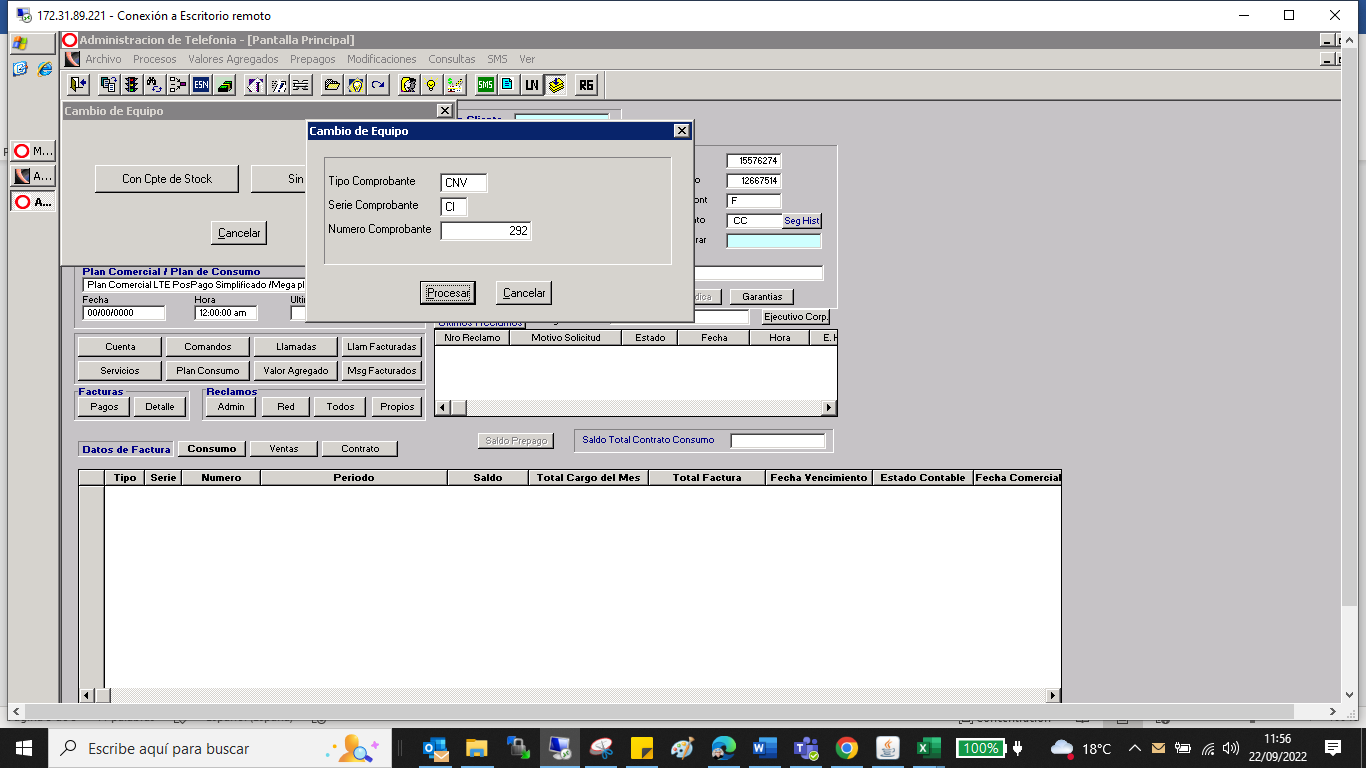


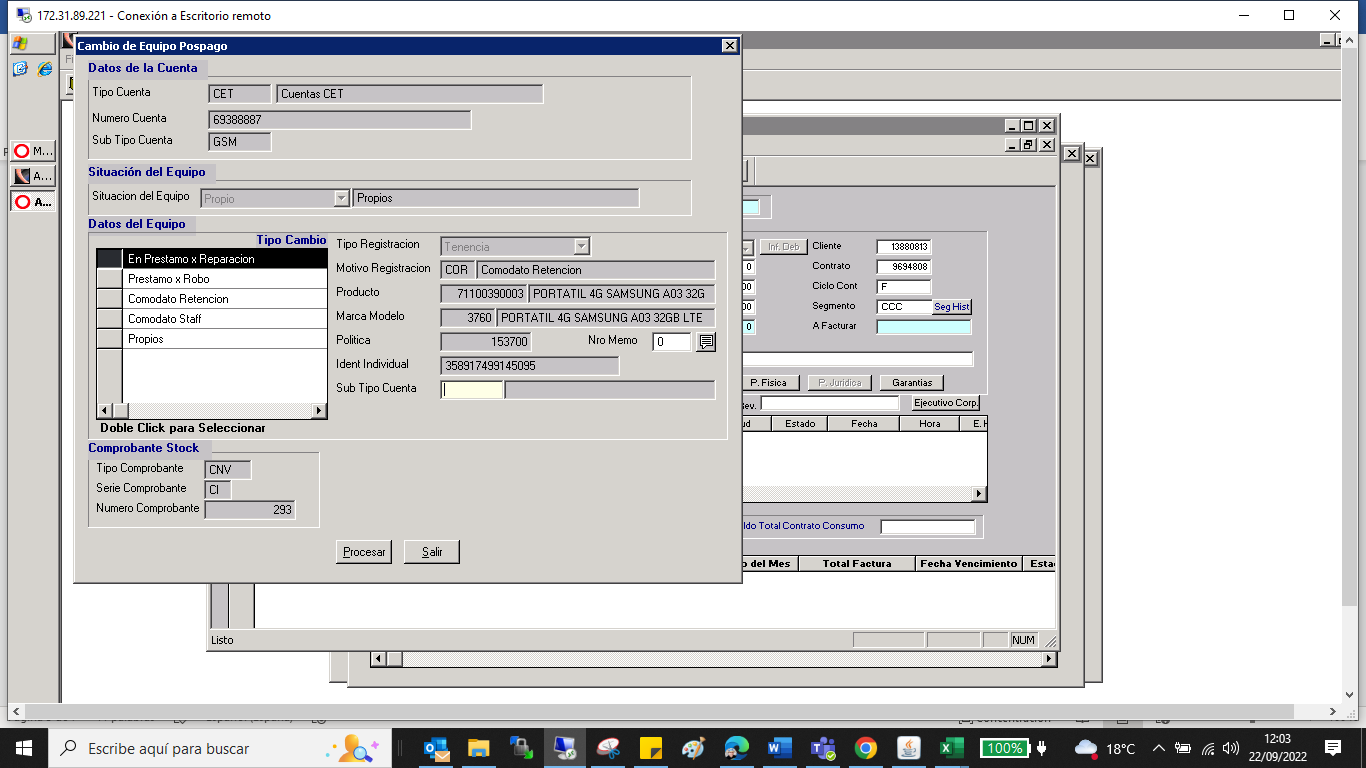


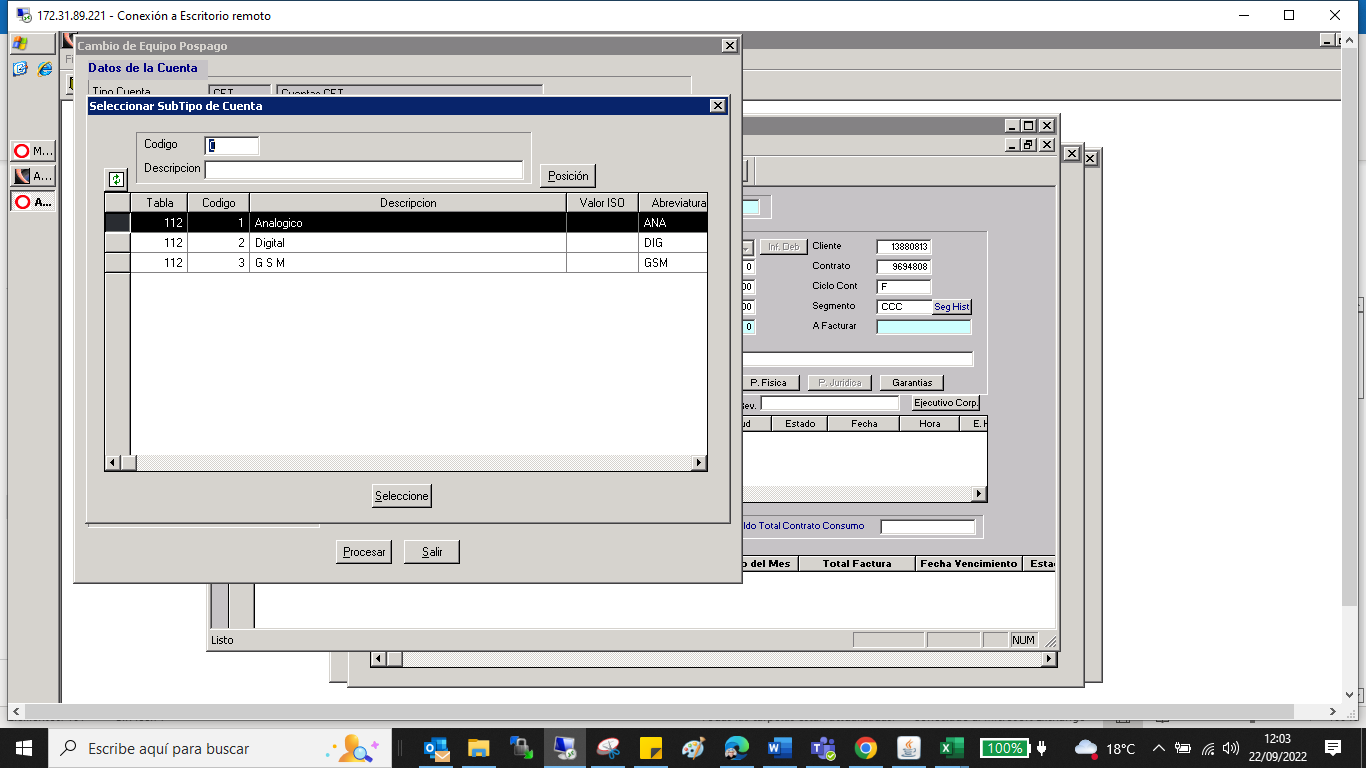


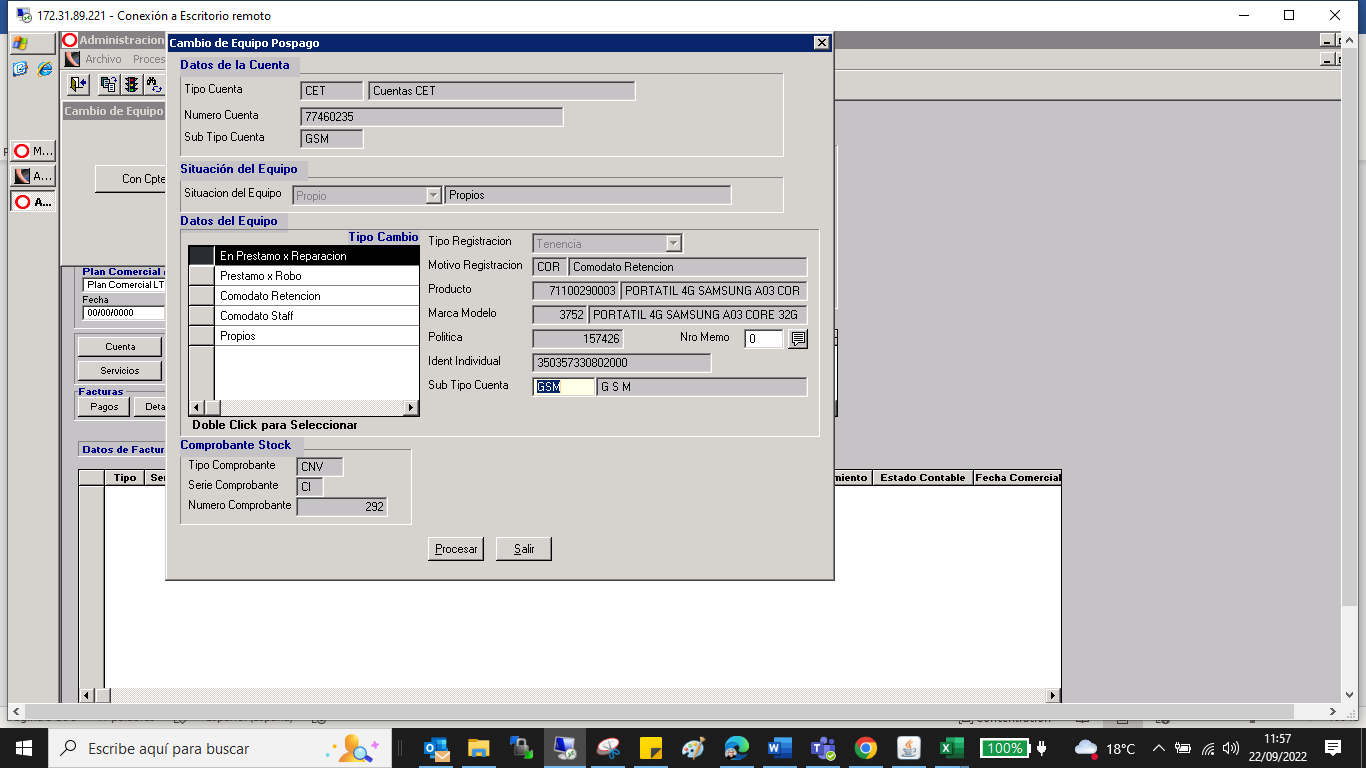


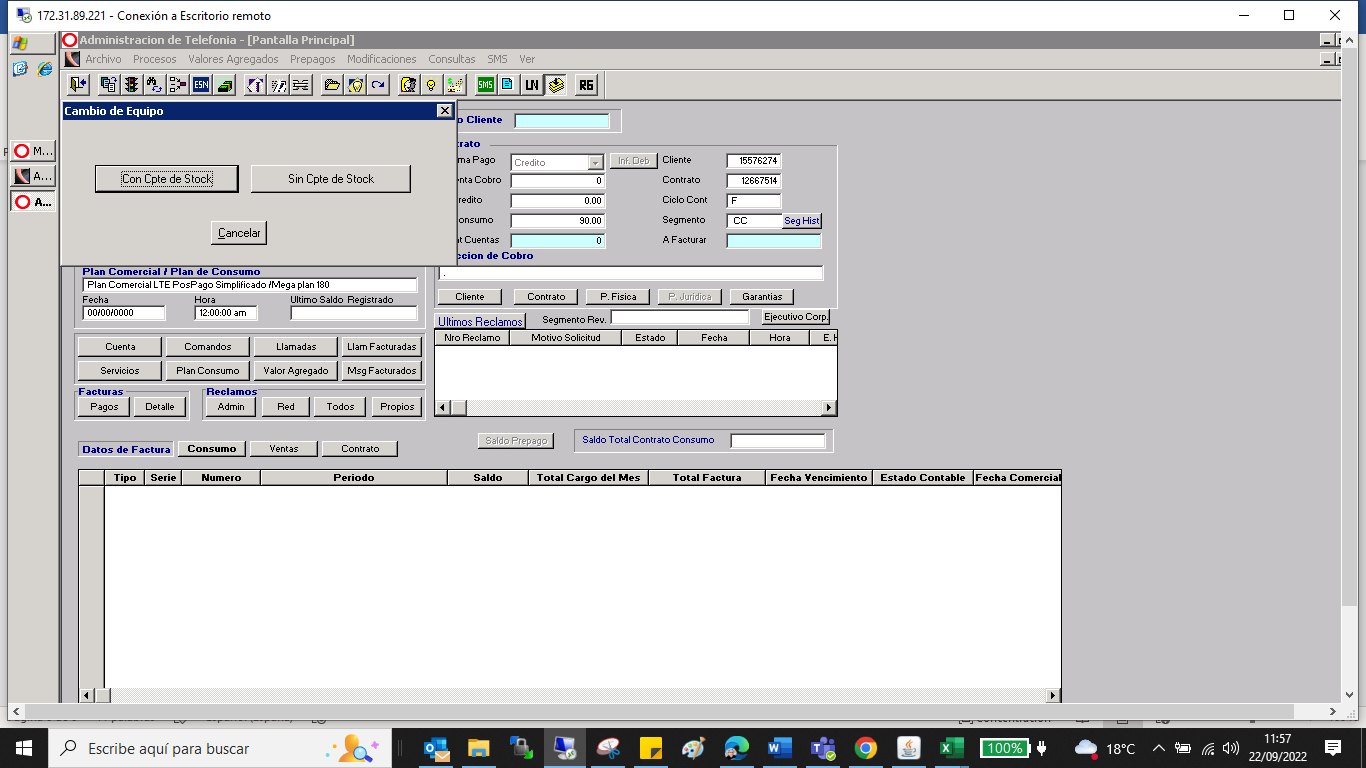


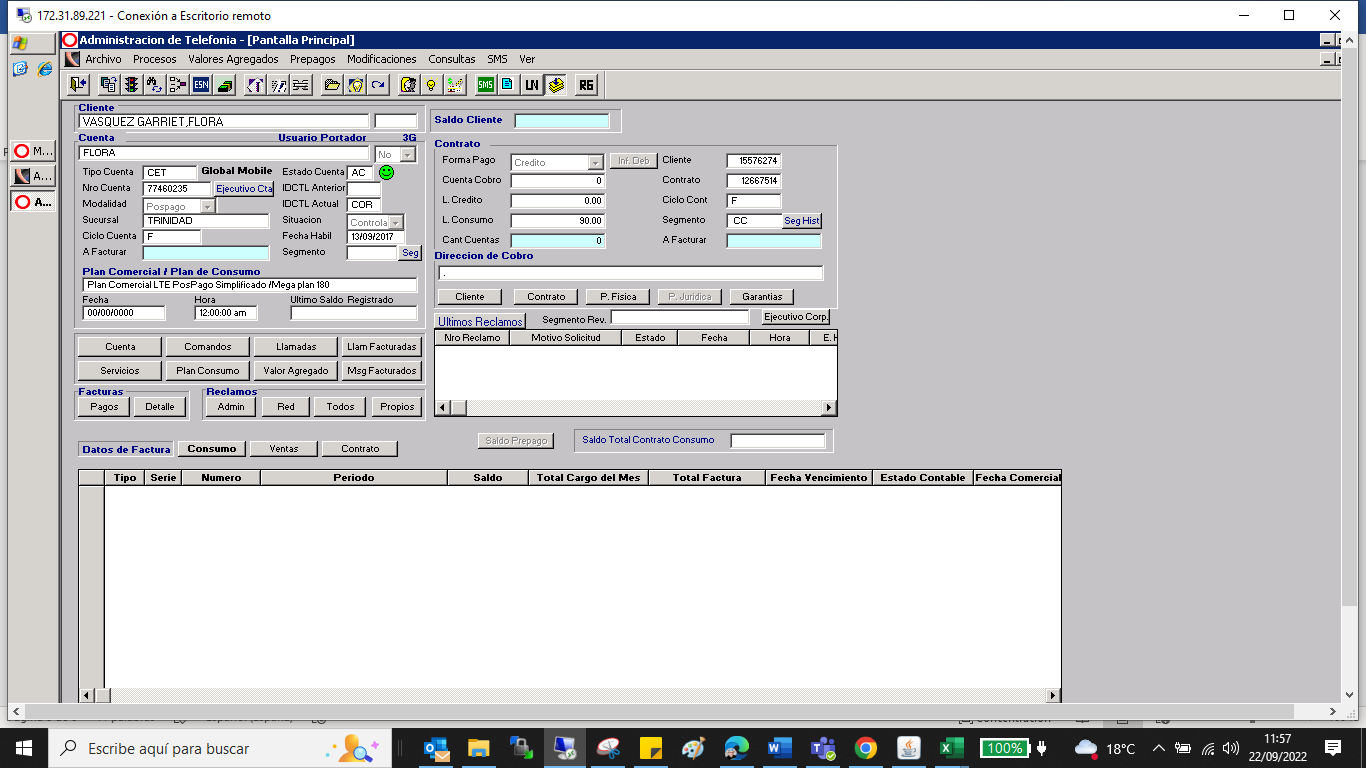






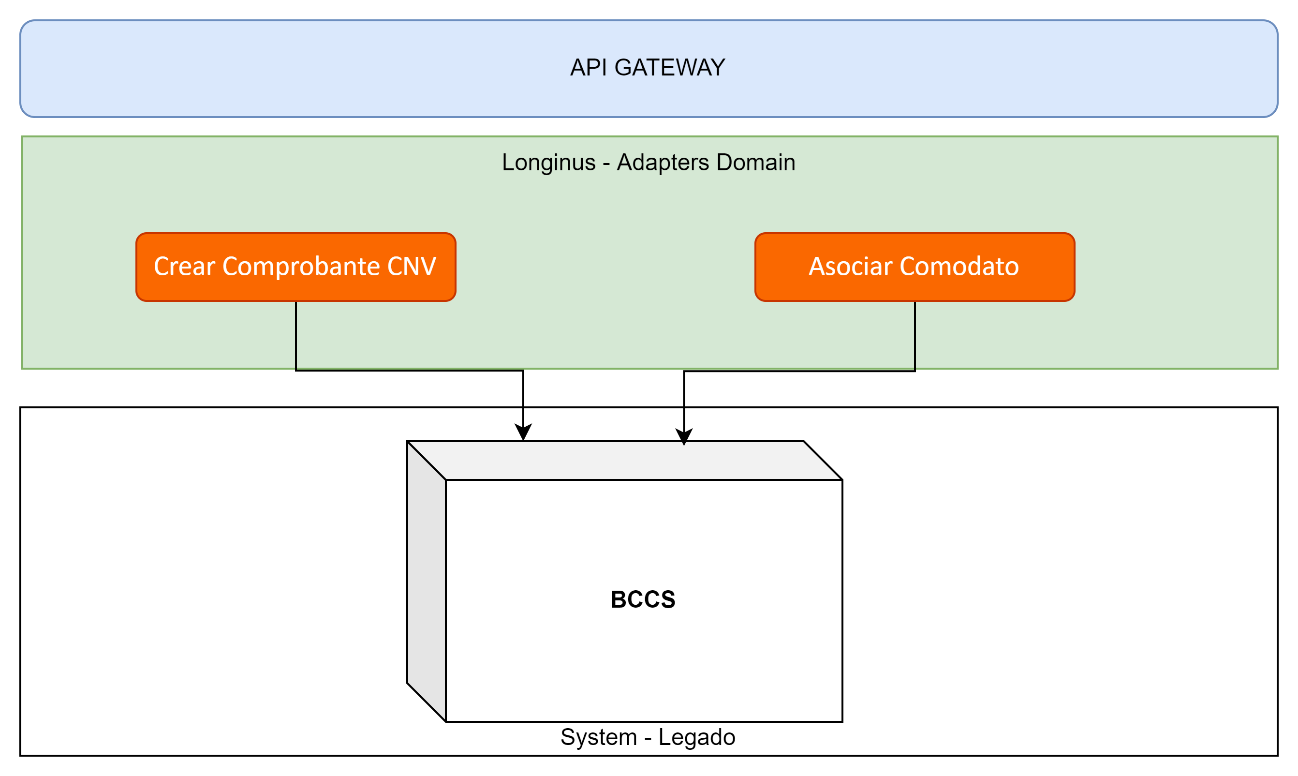






### 2. SOLUTION ARCHITECTURE

## 2.1 Vista Macro



## 2.4. Development View

