# **Documentación del Subproyecto: silver\_isp (Infraestructura)**

**Versión documento**: 1.0 — 10/9/2025

## **Visión General y Estratégica**

Este documento proporciona una descripción técnica y funcional completa del módulo silver\_isp. Su propósito es servir como referencia para desarrolladores, administradores de sistemas y gerentes de proyecto involucrados en su implementación, mantenimiento y evolución.

### **Objetivos de Negocio**

* **Centralizar el Inventario de Red:** Crear un registro único y fiable de todos los activos de red.
* **Automatizar el Aprovisionamiento:** Facilitar la configuración y activación de servicios para nuevos clientes.
* **Mejorar la Visibilidad de la Red:** Ofrecer herramientas de visualización y mapeo de la infraestructura.
* **Gestionar el Ciclo de Vida del Cliente:** Controlar el proceso completo desde el contrato hasta la baja del servicio.

### **Roles y Responsabilidades**

A continuación se definen los roles clave para el éxito del proyecto.

* **Jefe de Proyecto (Cliente):** *Carolina Istúriz*
* **Administrador de Red Principal:** *Luis Mora*
* **Responsable de Contratos y Clientes:** *Luis Mora*
* **Líder Técnico (Implementador):** *Sergio Velásquez Zeballos*

## **Alcance funcional**

* **Inventario jerárquico:** isp.zone → isp.node → isp.post → isp.box → isp.splitter.
* **Modelado de OLTs y su estructura:** isp.olt → isp.olt.card → isp.olt.card.port.
* **Gestión de dispositivos de red** (netdev) **y RADIUS** (isp.radius, isp.radius.user, wizards de NAS/CoA).
* **Contratos ligados a puertos/splitters:** isp\_contract se enlaza con puertos/ splitters y genera usuarios RADIUS.
* **Endpoints HTTP para topología:** controller que devuelve estructura JSON (usado por mapeo).

## **Arquitectura general**

* **Usuarios / Frontend**
  + Técnicos NetOps (mapa de red, provisión ONUs, tickets).
  + Administradores/gerencia (reportes, KPIs de saturación).
  + Integraciones externas (WhatsApp → Tickets, CRM → Contratos).
* **Capa de aplicación**: Odoo + módulos silver\_isp (models, wizards, controllers).
  + **Modelos principales**:
    - Activos físicos: isp.zone, isp.node, isp.post, isp.box, isp.splitter.
    - Red activa: isp.olt, isp.olt.card.port, isp.netdev.
    - Servicios: isp.contract, isp.radius.user.
  + **Wizards y controladores**:
    - /network/nodes (mapa de red).
    - Wizards RADIUS (NAS, disconnect, client).
  + **Reglas de negocio**: validación de factibilidad, provisión automática, suspensión por morosidad.
* **Persistencia**: PostgreSQL (estándar Odoo).
* **Integraciones**:
* **RADIUS**: autenticación, autorización y accounting.
* **Equipos de red (OLT, routers, APs)** vía SSH/API.
* **Sistemas de soporte** (WhatsApp/Helpdesk).
* **Despliegue recomendado**: git + proxy inverso (NGINX) + vault .

## **Indicadores KPI**

Los **KPI de infraestructura** que se pueden medir directamente desde el módulo silver\_isp se agrupan en **capacidad, calidad operativa, soporte y seguridad**. Aquí van los más relevantes:

### **KPIs de capacidad y uso**

* + 1. **% de puertos OLT ocupados**
       1. (puertos usados / puertos totales) \* 100
       2. Campo: capacity\_usage\_port\_pon en isp.olt.card.port.
    2. **% de splitters ocupados**
       1. (contratos activos / capacidad del splitter) \* 100.
       2. Campos: contracts\_count, capacity en isp.splitter.
    3. **Disponibilidad de zonas/nodos**
       1. Número de nodos activos por zona (isp.zone → isp.node).
    4. **Inventario de CPEs / ONUs libres vs asignados**
       1. Conteo en isp.onu y contratos que los usan.

### **KPIs de provisión y operación**

* + 1. **Tasa de éxito en provisión automática**
       1. (provisiones exitosas / total de intentos) \* 100.
       2. Fuente: logs de isp.netdev y workers NetOps.
    2. **Tiempo medio de provisión (MTTP)**
       1. Minutos entre la creación de contrato y el alta efectiva en RADIUS/OLT.
    3. **Errores de capacidad (overbooking evitado)**
       1. Intentos fallidos de asignar contrato a puerto/splitter saturado.
  1. **KPIs de soporte**
     1. **Incidencias por nodo/zona**
        1. Tickets vinculados a isp.node o isp.zone.
     2. **Tiempo medio de resolución de incidencias (MTTR)**
        1. Desde creación de ticket hasta cierre.
     3. **Porcentaje de suspensiones automáticas vs manuales**
        1. Campo is\_cutoff\_port en puertos + registros CoA.

### **KPIs de seguridad / cumplimiento**

* + 1. **Credenciales migradas a Vault**
       1. % dispositivos (isp.netdev) sin password en claro.
    2. **Trazabilidad de desconexiones RADIUS (CoA)**
       1. % de operaciones con log correcto en exec\_log.

Estos KPIs se pueden montar en **dashboards Odoo** o alimentar a **Grafana**.  
 Los más estratégicos para gerencia son:

* % puertos OLT libres,
* % splitters ocupados,
* MTTR (soporte),
* Tasa de éxito en provisión automática.

## **Modelos**

Añado aquí el nombre del modelo y su propósito principal.

### **Modelos estructurales:**

* **isp.asset**: modelo base para todos los objetos ubicables en un mapa (localizaciones gps, direcciones, fechas, marca ).
* **isp.netdev**: dispositivo genérico (ip, port, username, password, type, marca, modelo, serial). Es la base para routers, OLT, radius.
* **isp.zone**: área geográfica y lógica relacionada a un grupo de nodos con la misma salida a internet.

### **Modelos geográficos:**

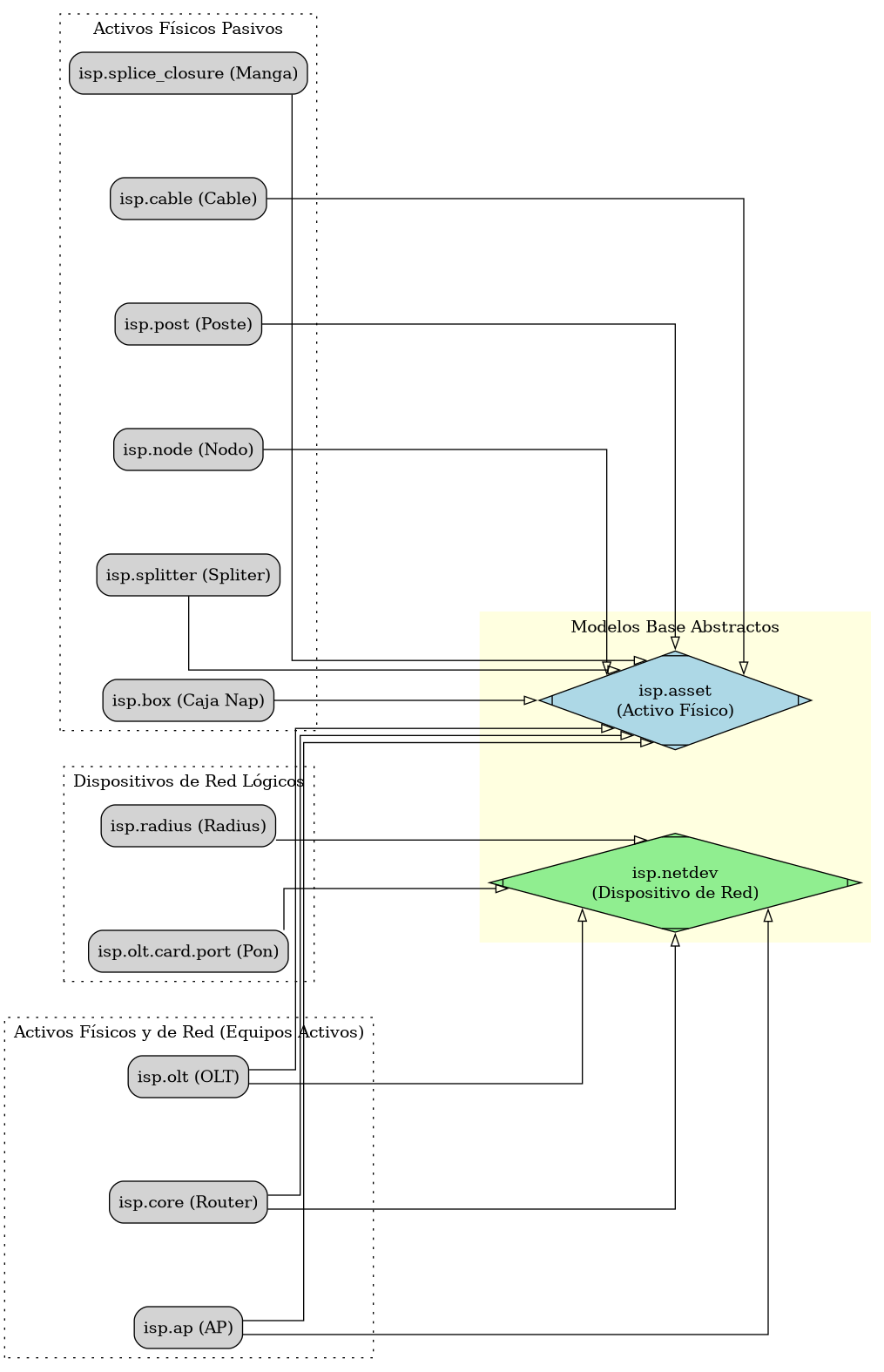
* **isp.node**: Punto de presencia principal para un grupo de equipos.
* **isp.post**: Postes. Principalmente para planificación de instalaciones.
* **isp.box**: Cajas Nap. Punto más cercano a la ONU.
* **isp.splice\_closure:** Mangas. Contenedores de splitters y OLT.
* **isp.splitter:** Jerarquía de divisores de cables.
* **isp.cable:** Indicadores gráficos de troncales y fibras..

### **Modelos de red activa**

* **isp.olt**: OLT con flags (is\_pppoe/is\_ipoe, secret, node\_id, core\_id).
* **isp.olt.card** y **isp.olt.card.port**: tarjetas y puertos; estos últimos contienen capacity\_usage\_port\_pon, s\_vlan, c\_vlan, campos de provisioning.
* **isp.ap**: punto de acceso inalámbrico y su relación con isp.core y isp.node.
* **isp.core**: Routers de acceso a internet.
* **isp.radius**: Manejo para NAS y desconexión (CoA).

### **Diagrama de Componentes de Alto Nivel**

El sistema interactúa con varios componentes. El siguiente diagrama ilustra la arquitectura general:



#### **Herencia de Clases y Modelos Base**

El módulo utiliza la herencia de Odoo para reutilizar lógica y campos, promoviendo un diseño DRY (Don't Repeat Yourself). Dos modelos base son fundamentales:

* isp.asset: Un modelo **abstracto** que proporciona campos y lógica para cualquier elemento considerado un "activo físico". Incluye campos para ubicación (dirección, coordenadas GPS), datos de instalación, marca y modelo. Cualquier modelo que represente un objeto físico hereda de él.
* isp.netdev: Un modelo **abstracto** para "dispositivos de red". Proporciona campos para la gestión de red, como dirección IP, puerto, credenciales de acceso, tipo de conexión (SSH, Telnet) y versión de software.
* mail.thread y mail.activity.mixin: La mayoría de los modelos heredan de estos mixins de Odoo para obtener el chatter, la gestión de seguidores y la planificación de actividades, lo cual es crucial para la trazabilidad.

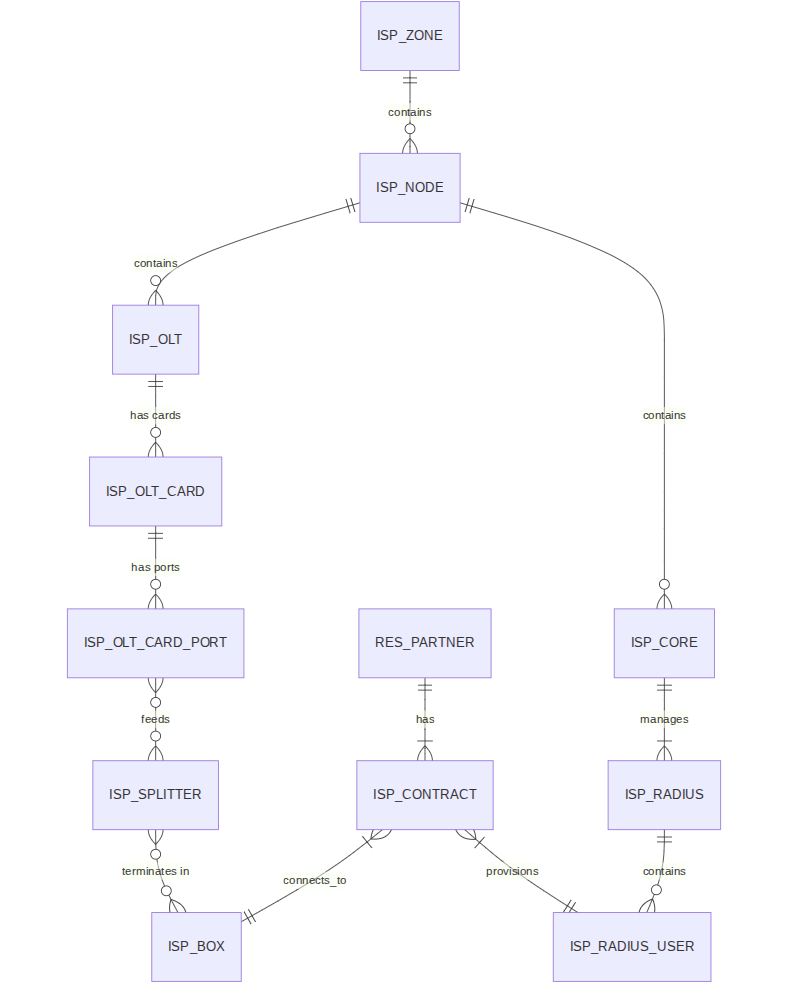
La siguiente tabla muestra la estructura de herencia de los modelos más importantes:

| **Modelo Hijo** | **Hereda de...** | **Propósito de la Herencia** |
| --- | --- | --- |
| isp.node, isp.post, isp.cable, isp.splice\_closure | isp.asset, mail.thread | Son activos físicos con ubicación y trazabilidad. |
| isp.core, isp.olt, isp.ap | isp.asset, isp.netdev, mail.thread | Son activos físicos y a la vez dispositivos de red configurables. |
| isp.olt.card.port | isp.netdev, mail.thread | Es un componente lógico de un dispositivo de red, por lo que es configurable. |
| isp.radius | isp.netdev, mail.thread | Se trata como un dispositivo de red configurable (el servidor RADIUS). |

## **Endpoints & Wizards**

* **Controller /network/nodes**: devuelve la topología JSON (nodo→postes→cajas→splitters), útil para el mapa. Revisar controllers/node\_network.py.
* **Wizards importantes**:
  + isp\_netdev\_radius\_client\_wizard.py — agregar cliente RADIUS (gestiona netdev\_id, secret, servicios).
  + isp\_radius\_disconnect\_wizard.py — desconexión/CoA por usuario.
  + isp.radius.nas.wizard y isp.netdev.radius.client.wizard (MikroTik focused).

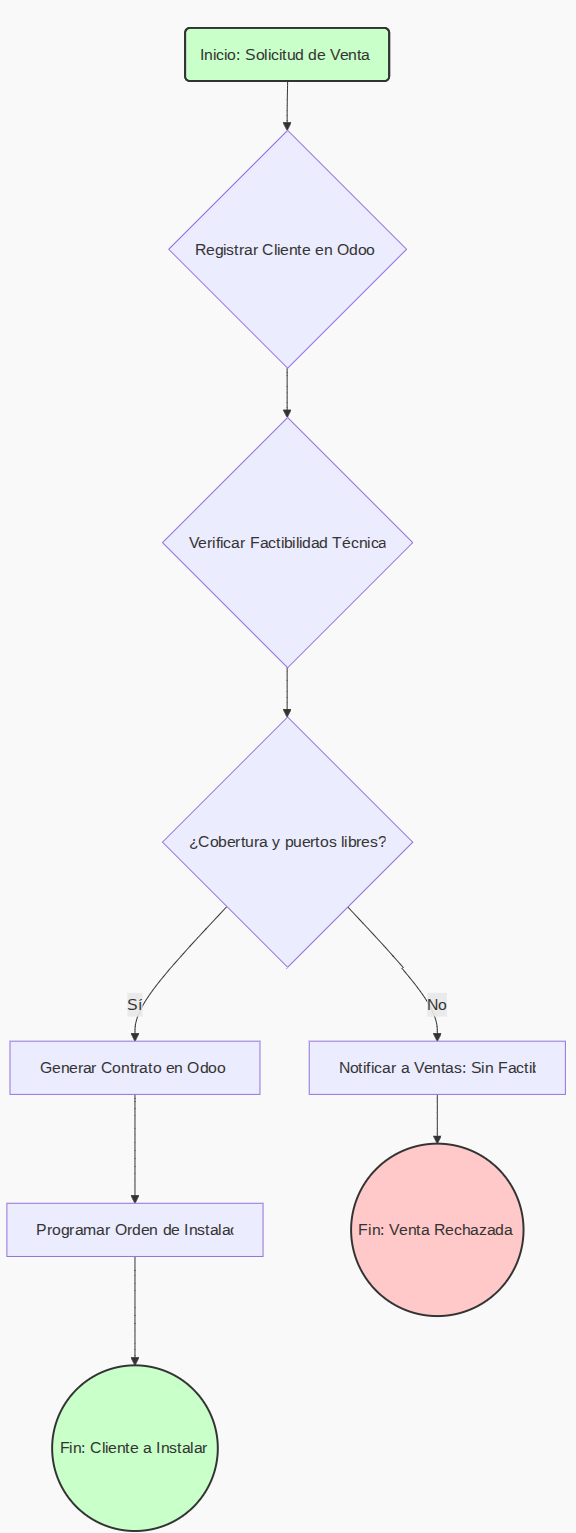
## **Diagrama Entidad-Relación**

El siguiente diagrama muestra las relaciones fundamentales entre las entidades principales del módulo. 

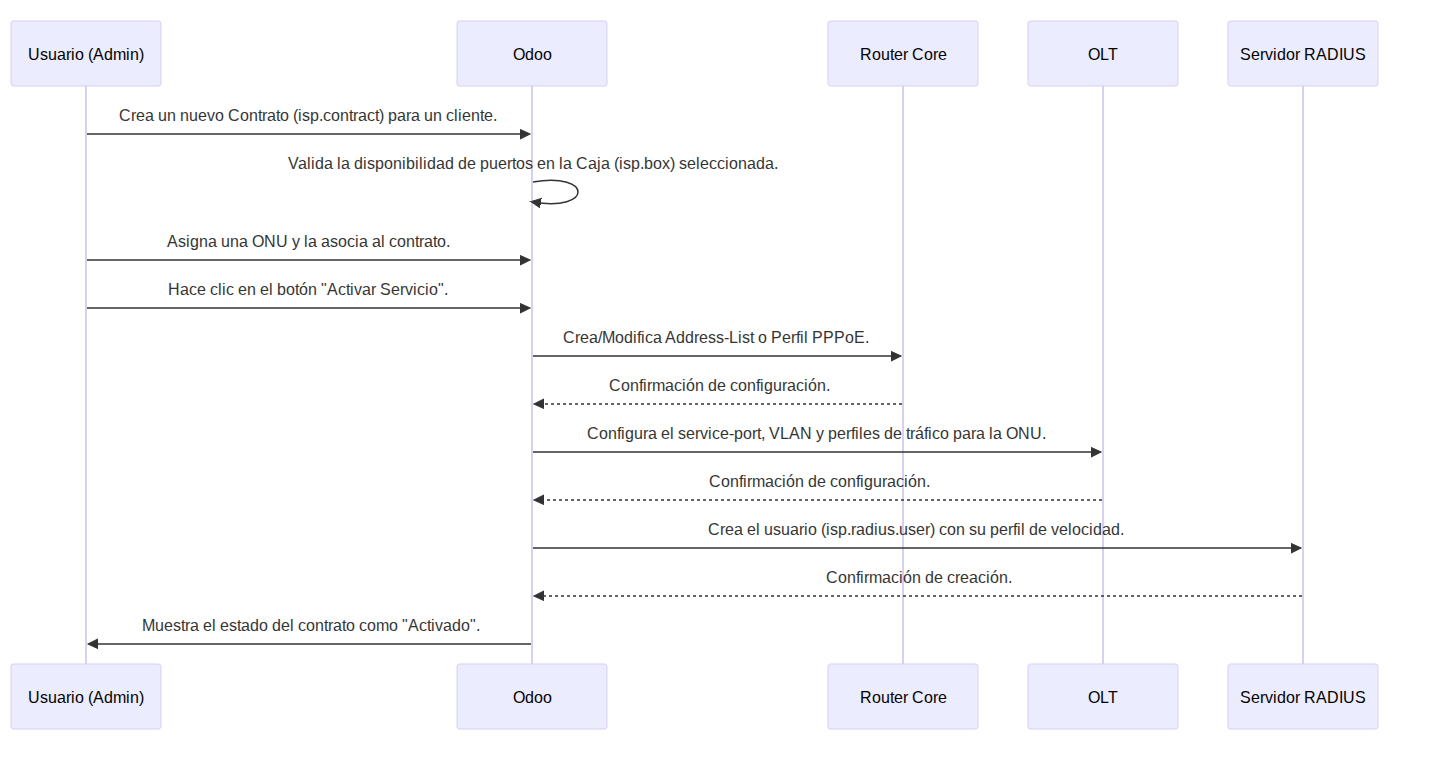
## **Flujos BPM críticos**

### **Alta cliente/venta → Validación de factibilidad**

Comprobar isp.olt.card.port.capacity\_usage\_port\_pon y isp.splitter.contracts\_count. Si OK, reservar puerto (marcar used\_by\_contract) y crear isp.contract → crear isp.radius.user.

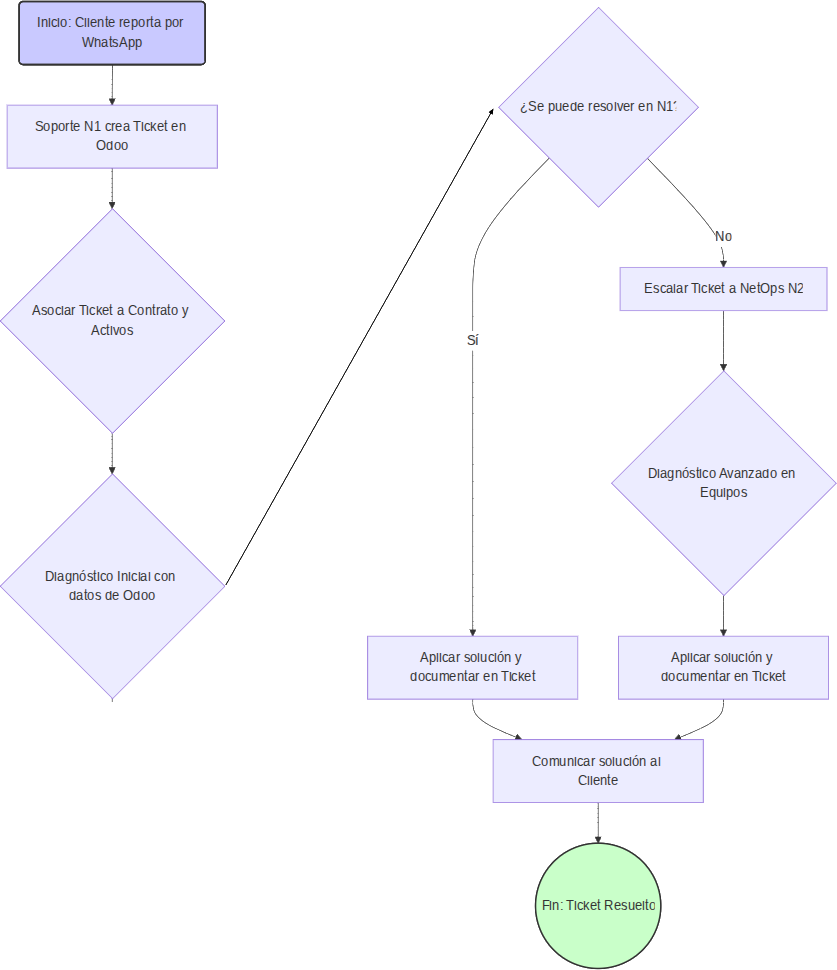


### **Instalación & provisión** Este es el proceso más crítico que el módulo busca automatizar. El siguiente diagrama de secuencia ilustra los pasos desde la creación del contrato hasta la activación del servicio.



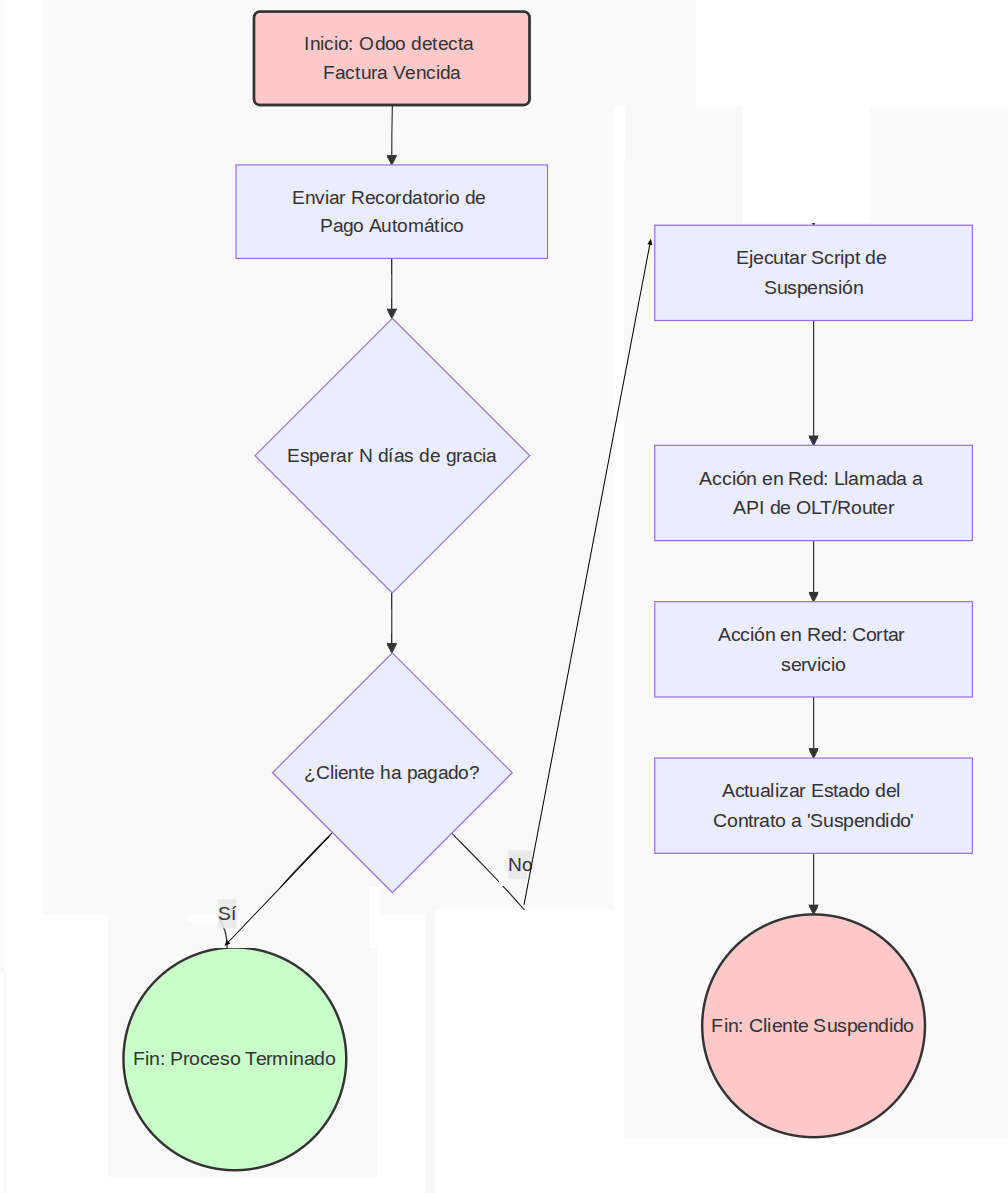
* 1. **Incidencia**

Mensaje entrante crea/actualiza helpdesk.ticket (integración externa). Ticket con categoría técnica genera task NetOps que ejecuta playbook/diagnóstico y actualiza isp.olt.card.port/isp.cpe según resultado.



* 1. **Suspensión por morosidad**

Factura vencida → proceso de cobranza dispara wizard/worker que ejecuta CoA/disconnect en isp.radius o marca is\_cutoff\_port. Registrar auditoría.



## **Integraciones externas**

* **RADIUS (AAA)**: pyrad es la librería sugerida en los archivos; los modelos y wizards facilitan la gestión de NAS y CoA. Se debe asegurar: conexión segura, reintentos, idempotencia en CoA (no crear cuentas duplicadas), y audit logging.
* **Equipos de red (SSH/API)**: isp.netdev almacena credenciales y campos api\_port, ip, username y password. Recomendación: mover credenciales a Vault y usar playbooks (Ansible) o workers en colas.  
   db
* **Visualización/Mapa**: endpoint /network/nodes alimenta frontend (web\_map). Ver web\_map dependency en el manifest.  
   silverisp

## **Calidad, pruebas y deuda técnica**

### **Recomendaciones de pruebas (priorizadas)**

* **Unitarias**: validaciones de modelos (e.g., unicidad serial\_number en isp.cpe, constraints en isp.olt\_card\_port).
* **Integración**: tests contra RADIUS (mock/pyrad), y simulaciones de comandos OLT (mocks SSH/API).
* **E2E/UAT**: escenario completo: venta → reserva puerto → asignación CPE → provisión RADIUS → primer login.
* **Migración**: validar contajes, sumas, y reconciliaciones (scripts ETL).  
   silverisp

## **Migración de datos**

* **Tablas críticas** para migrar: isp.asset, isp.zone, isp.node, isp.post, isp.box, isp.splitter, isp.olt, isp.olt.card, isp.olt.card.port, isp.contract, isp.radius.user, isp.onu/CPEs.
* **Reglas**: preservar ids externos cuando sea necesario (external\_id), reconciliar conteos de contratos por splitter/port, verificar capacity\_usage\_\* contra asignaciones reales.
* **Dress rehearsal**: hacer un parallel run con doble registro (ver tu política de cutover).

## **Seguridad y cumplimiento**

* **Secrets**: mover todos los password y nas\_secret fuera del código a un vault. isp.netdev hoy guarda password (no es aceptable en prod).  
   db
* **RBAC**: validar security/ir.model.access.csv — roles Usuario ISP y Manager ISP ya detectados (ajustar granularidad si se necesita).  
   silverisp
* **Auditoría**: activar mail.thread y mail.activity.mixin (ya usado), pero añadir logs escritos en exec\_log y registros de CoA/desconexión.

## **Operaciones y despliegue**

* **Provisionamiento**: worker/cola para tareas NetOps; retries exponenciales; dead-letter queue para fallos manuales.  
   silverisp
* **Backup/DR**: snapshots diarios; pruebas trimestrales de restore.
* **Despliegue**: Docker + docker-compose/ansible; ssync.sh existe pero recomiendo Ansible para robustez.  
   silverisp