UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

SISTEMA DE REGISTRO DE ÓRDENES PARA UNA EMPRESA DE TELECOMUNICACIONES

T E S I S

QUE PARA OBTENER EL TITULO DE:

INGENIERO EN COMPUTACIÓN

PRESENTA:

MIGUEL ÁNGEL AMÉZQUITA NEGRETE

DIRECTOR DE TESIS

ING. MARICELA CASTAÑEDA PERDOMO

Ciudad Universitaria, Cd. Mx 2016

AGRADECIMIENTOS

A Dios

Por permitirme estar.

A mi MADRE

Que siempre me ha dado ejemplo de amor, constancia y trabajo.
Te agradezco todos los sacrificios que realizaste tan solo para que pudiera salir adelante. Estaré siempre muy orgulloso de ser tu hijo.
Te quiero.

A mi Esposa

Que me tuvo la enorme paciencia y amor para terminar.

A mi Hijo

Mi, siempre, fuente de amor, inspiración y ejemplo.

A mis Maestros

Que me enseñaron que saber es muy importante, pero lo es más el compartir su conocimiento.
Espero algún día corresponder a su buen ejemplo.
A todos ellos y a los que ya no están.

A Mary

Quien siempre me brindó la oportunidad de continuar aun en las condiciones más difíciles.

A los Amigos
Que no se cansaron de decirme que terminara.

Muchas gracias a todos.

Miguel Ángel

1. *ENTORNO DEL SISTEMA*

1. Introducción

Un sistema de control debe permitir las modificaciones o adecuaciones para adaptarse a las cambiantes reglas de negocios de las empresas actuales.

En toda empresa existen procesos y sistemas que se establecen para el control de las distintas actividades, siempre con el fin de poder controlar las variables principales y aplicar las correcciones o modificaciones para su mejor funcionamiento.
Conforme las compañías van creciendo en complejidad las actividades se van dividiendo y especializando y las separaciones en áreas, grupos o sistemas se vuelven más difíciles de controlar.

En una compañía con pocos empleados y una o dos áreas, la información y tareas necesarias se pueden realizar por un mismo empleado y tener un control que permite un funcionamiento adecuado. Sin embargo cuando las actividades requieren de mayor especialización y atención, no solo se requiere el control interno para la realización del trabajo en específico sino la información y el control suficiente para determinar el funcionamiento adecuado de un área con respecto de todos los procesos.

En la actualidad existen muchos sistemas que ofrecen el control de las distintas áreas y procesos de una empresa y que permiten el control de las variables importantes de las mismas. Estos sistemas, dependiendo del presupuesto asignado a este fin, pueden ser tan aislados o modulares como el control de la nómina, el contable, el almacén, o llegar a establecer modos de control específicos para cada área estableciendo parámetros de eficiencia para toda la compañía, siguiendo estándares internacionales.

Los sistemas ERP (Enterprise Resource Planning) son sistemas de gestión de información que automatizan muchas de las prácticas de negocio asociadas con los aspectos operativos o productivos de una empresa. Baste analizar algunas de las bondades ofrecidas por cualquier ERP para darse cuenta que es lo que toda empresa desearía tener:

- Debe poder moldearse a las reglas actuales y futuras del negocio.
- Es modular y contempla todas las partes administrativa, operativa y financiera.
- Debe ser sencillo de implantar y adecuar al negocio.
- Debe respetar las reglas esenciales de tu negocio.
- Debe acercarte a tus clientes y proveedores
- Integra todas las áreas de una empresa llevando la información en línea y evitando la recaptura.
- Facilita y sustenta la toma de decisiones
- Facilita el flujo de la información.
- Proporciona información clara, oportuna y confiable.

Esto es al menos en teoría. En la práctica las cosas no suelen ser tan claras. Muchos de los sistemas medianamente grandes requieren de la convivencia con otros y requieren de alguna forma de inserción o extracción de información y las empresas terminan desarrollando sistemas débiles de trasferencia de información interna y externa. Ejemplo claro de lo anterior, por lo menos en el país, es el pago de impuestos. Los cambios en el pago de impuestos no solo implican un cambio en la fórmula del cálculo de los mismos si no diversas consideraciones en los registros

de información, deducciones, almacenamiento, verificaciones y nuevas reglamentaciones no solo en los sistemas propios si no entre proveedores y vendedores.

Cuando se suman los costos de las nuevas implementaciones resulta claro que las empresas pagan altos costos por ellas. Para mantener en funcionamiento sistemas caros (Por su precio en el mercado) se requiere no solo de la intervención de especialistas en la aplicación implementada, con su elevado costo, si no de la capacitación de empleados propios, también con altos costos y lo que es peor, no se tiene la posibilidad para modificar la aplicación y para adecuarla tantas veces como sea necesario. Ante este panorama el dueño o empresario termina por invertir el mínimo indispensable que complica aún más la situación.

Esta PYME presenta las oportunidades o deficiencias características de las PYMES. Procesos y reglas del negocio en desarrollo, rápido crecimiento, bajo presupuesto, mucha competencia, poca especialización, falta de capacitación, entre otras. En esta situación el sistema se plantea como una alternativa de solución con base en el registro de órdenes de servicio.

El sistema no intenta resolver toda la problemática que presenta la compañía mas bien adecuarse a sus capacidades para la integración de un sistema que le permita llevar un mejor control y disponer de mejor información para la toma de decisiones que pueda modificar con sus propios recursos.

La empresa se dedica principalmente a la instalación de enlaces privados y otros servicios de telecomunicaciones.

El registro de la información relativa a los enlaces presenta, sin duda, una oportunidad para el establecimiento de controles, que permitirán disminuir los tiempos requeridos y contar con la información suficiente para que todos los procesos involucrados.

Los Enlaces de Comunicaciones
Los enlaces de comunicaciones son conexiones físicas, inalámbricas o lógicas que se establecen entre dos o más equipos con el fin de intercambiar información.
Los enlaces pueden realizarse entre sucursales o ser dedicados a internet además de establecer redes privadas o servicios en la red (internet).

Entre los servicio que presta están

- Enlaces privados dedicados inalámbricos.
- Enlaces privados dedicados inalámbricos a internet.
- Establecimiento de redes privadas en instalaciones propias
- Establecimiento de redes privadas sobre la red
- Establecimiento de servicios de telefonía privada
- Servicios de administración remota de servidores.
- Servicios de seguridad informática.

La principal función de la empresa es la instalación física de los enlaces necesarios para la implementación de servicios. Los clientes también pueden ser otras empresas que ofrezcan otros servicios. Es por esto que la asociación entre empresas ha sido necesaria para dar al cliente una solución completa a los problemas de comunicación que requiere resolver.

Existen empresas asociadas dedicadas, por ejemplo, a la seguridad informática que requieren de la instalación de enlaces dedicados específicamente para esta función y que no cuentan con la infraestructura física para dar el servicio. Estas empresas necesitan solicitar los costos y los

tiempos en que podrían realizarse las instalaciones que necesita. De la efectividad de estas peticiones depende la buena relación de negocios que se establezca entre las compañías

La instalación de los enlaces o servicios requiere la realización de diversas actividades para lograr la instalación, conexión y funcionamiento o viabilidad, por lo que es necesaria la intervención de distintas áreas o personal que pueda cumplir con los requerimientos.

Para establecer un enlace se requiere de mucha información, procesos, equipos, personal entre ellos:
Del enlace

- Su localización geográfica a nivel GPS.
- Torres de comunicaciones o postes
- Sistemas de energía ininterrumpida
- Antenas emisoras y receptoras y equipo para fijarlas
- Cables y conexiones dependiendo de la tecnología
- Servidores, switches, enrutadores. Además de sus configuraciones
- Sistemas de respaldo

De administración:

- Datos fiscales del cliente
- Direcciones de oficinas.
- Permisos locales o federales y contactos
- Cotizaciones previas
- Contratos, autorizaciones
- Evaluación de propuestas
- Costos y disponibilidad de equipos
- Manejo de Comisiones

Se requiere además de áreas especializadas de operaciones que se encarguen del funcionamiento de las redes establecidas, el equipo y los servicios, para lo cual se requiere de equipos y aplicaciones de monitoreo y mantenimiento especializados y de nivel critico

Para controlar todas estas operaciones esta empresa por ejemplo, se ha dividido en áreas, departamentos y funciones especiales, así podemos encontrar:

- Instalaciones
- Ventas
- Ingeniería
- Operaciones
- Compras
- Administrativa
- Contabilidad
- Personal
- Dirección y Gerencia
- Publicidad

Y ha adquirido diversas aplicaciones especializadas para el manejo de cada departamento, área o función entre ellas:

- Nómina
- Manejo de comisiones para vendedores
- Monitoreo de equipo y enlaces.
- Help Desk
- Contabilidad
- Almacén
- Office
- Correo
- Páginas web
- Servidores especializados

Todas ellas en constante modificación y adecuación.

La empresa requiere que el sistema de registro de la información relativa a los enlaces pueda modificarse y adecuarse a las capacidades, necesidades cambiantes de su negocio y que pueda integrarse a los demás sistemas.

A raíz de los primeros análisis, la necesidad de consultoría es evidente y necesaria pero el costo de ella, a vista de la baja capacitación, resulta incosteable. Los beneficios de una consultoría generalizada de los procesos de la empresa resultarían en enormes beneficios económicos y de supervivencia.

2. Objetivo

Construir una aplicación que permita a la empresa llevar el registro de las órdenes de instalación de los servicios de telecomunicaciones que oferta, mediante el manejo de órdenes de servicio en los diferentes departamentos y áreas de la empresa

Objetivos particulares

La aplicación se integrara a los demás sistemas que maneja la empresa, bajo el control del personal que designe para ello.

La aplicación dará una solución básica y modificable para las áreas involucradas en la instalación de los enlaces y el seguimiento de las actividades requeridas.

La información que se recabe deberá servir para conocer básicamente:

- Cuantas solicitudes se han realizado y en que estatus se encuentran.

Es decir si una solicitud de instalación ha sido aceptada, si ya se ha realizado el estudio de viabilidad, Si se encuentra en estudio o instalando

- A quien o a que empresa pertenecen o quien las pidió.
- Que se pidió y cuando

Se pretende que esta información permitirá

- Reducir el tiempo de respuesta y aumentar la eficiencia de las diferentes áreas.
- Permitirá conocer la magnitud del negocio propio y que se tiene con las empresas aliadas
- La planeación del crecimiento y soporte
- La planeación de gasto
- Colabore en controlar la operación

La seguridad permitirá a las empresas afiliadas ver su información y no las del resto del grupo o de la empresa, así como también los administradores de las órdenes no podrán acceder a toda la información de las órdenes de compra o su autorización.

Los reportes presentarán gráficos y datos duros transferibles para su utilización.

La aplicación no pretende ser el sitio web de las empresas ni ser una aplicación masiva, pues se pretende que solo se tenga un acceso o responsable en las áreas involucradas.

La aplicación no se planea como una aplicación de uso critico porque ni la adquisición de la información ni su explotación es determinante en el mismo momento. Es decir cuando se registra la información en la aplicación no se realiza en el momento en que esta se produce si no como resultado de actividades previas
Aunque la aplicación no es crítica, la información si es de mucha importancia y se requerirá de un nivel de respaldos que permitan su recuperación y seguridad.

3. Reglas del negocio

La empresa se dedica a la instalación de enlaces privados ya se entre empresas o hacia internet además de establecer redes privadas o servicios al través de enlaces en internet.

Venta de servicios

La empresa está organizada por sucursales o plazas. Cada plaza cuenta con su propio personal para realizar las operaciones básicas del negocio y cuando se requiere equipos especializados, desplazan equipo y personal.

Las principales sucursales son México, Guadalajara, Querétaro en lo que se refiere a instalaciones propias. Monterrey y Zacatecas para monitoreo de instalaciones externas además se cuenta con instalaciones especiales que se han realizado en poblaciones en el interior de la república.

Cada plaza es una unidad independiente que llevan su propia organización administrativa cuando son grandes o de un solo miembro en el caso de las más pequeñas. En algunos casos solo se renta el nombre de la empresa

La venta de servicios es realizada por un vendedor que se pone en contacto con las empresas mediante dos mecanismos

- Por publicidad
  Donde el cliente llama a las oficinas en la plaza.
- Por invitación
  Donde el cliente es visitado para ofrecer los servicios que presta la empresa.

Estos vendedores pueden ser propios o aliados, es decir que sean contratados por la empresa o que pertenezcan a una empresa asociada que vende otros servicios y que asocia su solución a uno de los productos de la empresa.

Los productos y los precios

La empresa ofrece productos de dos categorías,

- Los propios
- Los externos
- La combinación de ambos

Los propios son los que dependen de la infraestructura y personal de la compañía para proveerlos y mantenerlos.

Los externos los servicios en los que participa como parte de una solución.

La combinación de ambos se refiere a productos donde la compañía se sirve de productos externos para generar una solución propia.

Clasificación de productos	
Los propios	Enlaces dedicados, Monitoreo de servidores.
Los externos	Seguridad administrada, Voip
Los combinados	Enlaces de última milla administrados

Figura 1.3.1 Clasificación de productos

Estructura de precios

La estructura de precios es compleja debido a la cantidad de productos y las variantes de los mismos

Acotando la problemática que presenta esta estructura a los productos a los que se refiera la aplicación (Enlaces) se puede establecer lo siguiente

- No existe un único precio por producto
- El precio depende del cliente y contrato
- Existen precios distintos por fecha o antigüedad
- Las promociones y descuentos dependen del cliente y ocasión.
- Los precios dependen también del lugar físico donde se encuentren
- Los precios dependen también de la infraestructura necesaria.
- Los precios son fijados por el área de ventas y la dirección
- Los precios pueden dividirse

Este es uno de los puntos sobresalientes en el análisis de negocio para la ingeniería y reingeniería de procesos que necesita la empresa.

El análisis de información de precios resulta difícil de analizar y presentar pues por ejemplo un catálogo de productos con precio resulta muy poco útil.

Para analizar adecuadamente los precios tendría que considerarse:

- La información de la fecha de contrato y su duración
- Las fechas de la o las promociones otorgadas,
- Los descuentos especiales y a que aplican.
- El lugar donde se encuentra.
- Si forma parte de acuerdos o no.
- Si tiene planes especiales o créditos

El análisis de precios podría llevarse a cabo como si se tratase de facturaciones de servicios analizando los tipos de promociones y descuentos y zonas, sin embargo en el primer análisis de la información realizado se puede observar la carencia de una forma estandarizada para la otorgación de descuentos, promociones y/o, facilidades y la falta del registro apropiado de dicha información. Los análisis quedan acotados a que producto se vende más, cuanto se factura por periodo, cliente, instalaciones, etc.

Asociaciones

Existen al menos siete empresas asociadas formalmente (nombres ficticios por seguridad) además de la empresa entre ellas:

- **Edetel** dedicada a telefonía sobre ip
- **Sadmex** dedicada a servicios de administración remota.

Estas empresas son llamadas internamente canales de venta

El precio y la comisión de los enlaces y/o servicio dependen de los acuerdos que se establecen entre compañías y de la importancia o valor del proyecto, por lo que un mismo enlace, en términos técnicos, puede tener una gran variación de precio por pertenecer a un proyecto o canal en particular como se estableció en el inciso anterior.

Los acuerdos entre las empresas quedan registrados en contratos que garanticen bajo diversos criterios ya sea la disponibilidad de los servicios, la atención, la calidad y la eficiencia entre otros.
Hasta la fecha las discusiones sobre montos facturaciones y relación entre empresas se realiza en juntas programadas mensualmente, en el mejor de los casos o cuando se presentan problemas graves, donde cada quien lleva el soporte escrito de los productos y servicios otorgados y facturados.

Operaciones y Monitoreo

El monitoreo de la infraestructura de comunicaciones es de vital importancia para el mantenimiento de las operaciones en la red. Es por ello que muchas de las actividades quedan relegadas a esta actividad. La instalación de nuevos enlaces es una ellas. Para instalar un nuevo enlace es necesario el visto bueno del área dedicada al soporte y monitoreo de la red y los equipos y puede determinar su instalación o no en los tiempos previsto por dificultades o mantenimientos

Existen operaciones de monitoreo y control que se llevan a cabo para otras empresas y por ello existen estándares de servicio acordados según sea el caso. Estas condiciones influyen en la eficiencia y el tiempo total de instalación de un enlace ya que, por ejemplo, si existe un nuevo enlace que compromete instalaciones o equipo que soportan este tipo de servicio la instalación puede tardar el triple del tiempo estimado. Figura 1.3.5

Proveedores

Los proveedores son parte fundamental de la operación diaria y de la planeación de infraestructura.
Debido a lo critico de las operaciones es necesario la redundancia o en su caso la alternativa de soluciones y productos de los proveedores.
Se tienen proveedores por ejemplo para la conexión a internet como TELMEX, MAXCOM, PROTEL, BESTEL, AXTEL entre otros. Estos proveedores requieren de condiciones especiales técnicas para proveer sus productos.

T-Carriers & E-Carriers

En telecomunicaciones, la portadora-T (Inglés: T-Carrier) es la designación de un sistema genérico de telecomunicaciones para los sistemas digitales multiplexados originalmente desarrollados por los Laboratorios Bell y utilizado en Estados Unidos y Japón.

La unidad básica del sistema de portadoras-T es el DS0 que tiene una velocidad de transmisión de 64 kbit/s y es normalmente usado para un circuito de voz.

El sistema de Portadoras–E (Inglés: E-Carrier), o sistema europeo de portadoras es incompatible con las Portadoras-T y se utiliza en el todo el mundo excepto en Japón y los Estados Unidos.
Figura 1.3.2 y figura 1.3.3

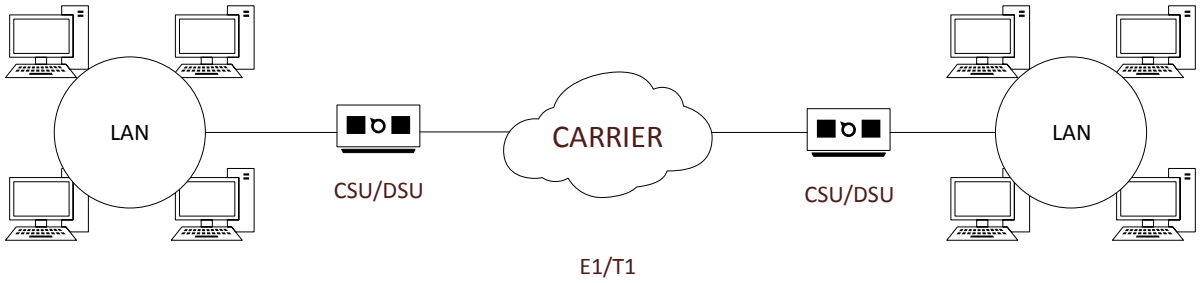


Figura 1.3.2 Carrier

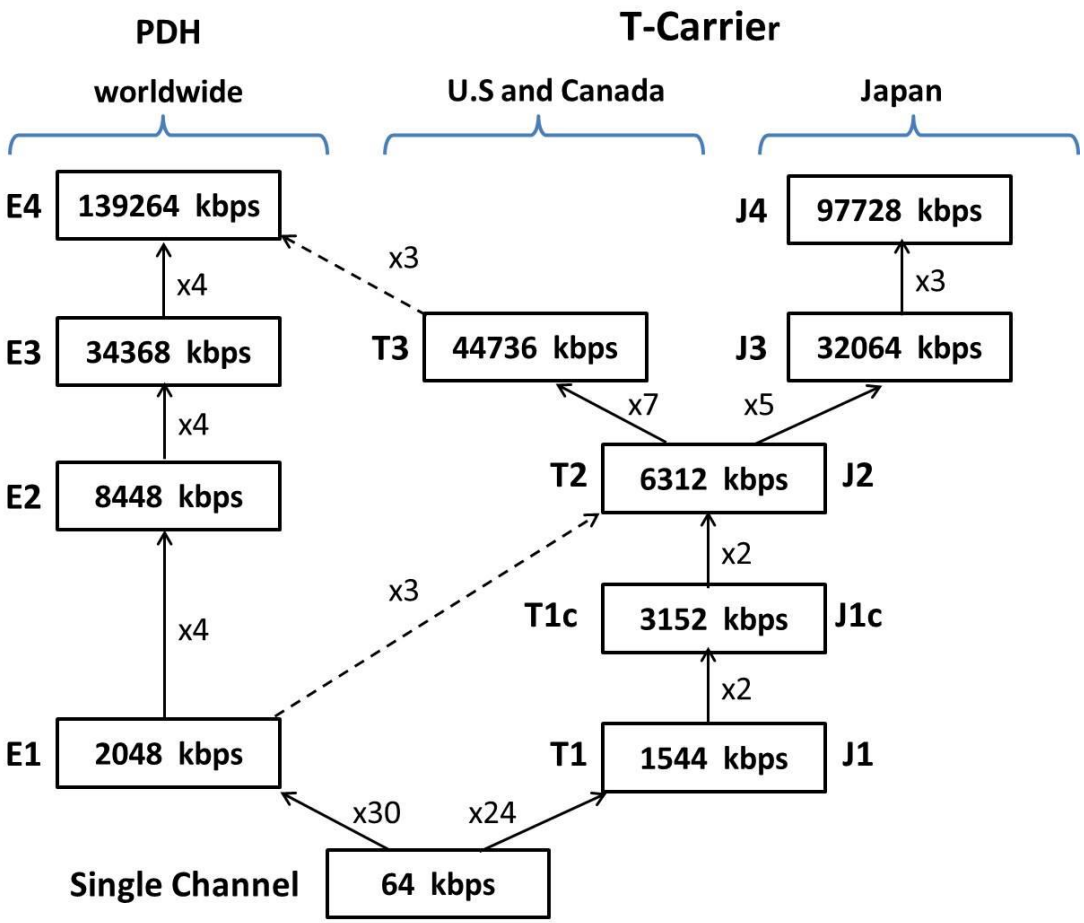


Figura 1.3.3 Nomenclatura de Enlaces

En el sistema europeo, se tiene hasta cinco jerarquías, como se puede observar en la siguiente tabla.

Jerarquía	Velocidad	Canales	Trama
E1	2048 Kbit/s	30	256 bits = 125.00 us
E2	8448 Kbit/s	120	848 bits = 100.38 us
E3	34368 Kbit/s	480	1536 bits = 44,7 us
E4	139264 Kbit/s	1920	2904 bits = 20.85 us
E5	564992 Kbit/s	7680	2688 bits = 4.7 us

Figura 1.3.4 Tipos de Enlaces

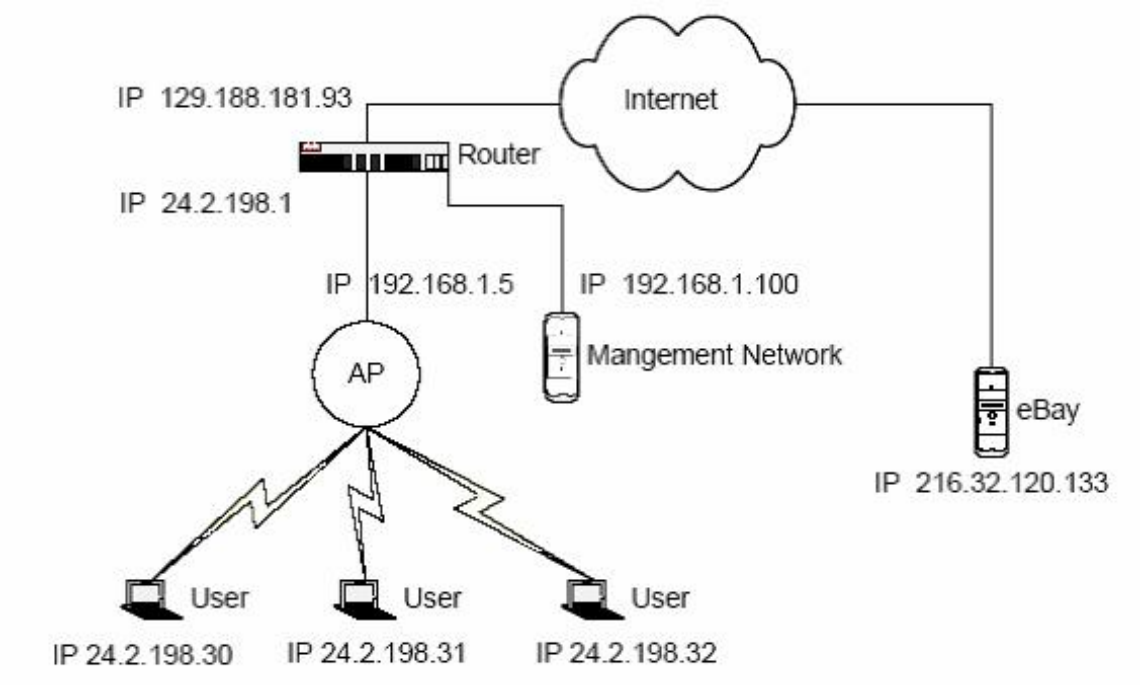


Figura 1.3.5 Red privada IP

Se cuenta además con proveedores de equipos de comunicaciones que implica distintos tiempos de entrega, reparaciones y sustitución según sea el caso. El costo de estos equipos de telecomunicaciones es elevado y su compra, renta o sustitución requiere de planeación del gasto. Como empresa de tecnología la modernización constante de la infraestructura y tecnologías es lo que permite continuar en la competencia de servicios y en la disminución de costos. Ejemplo figura 1.3.6

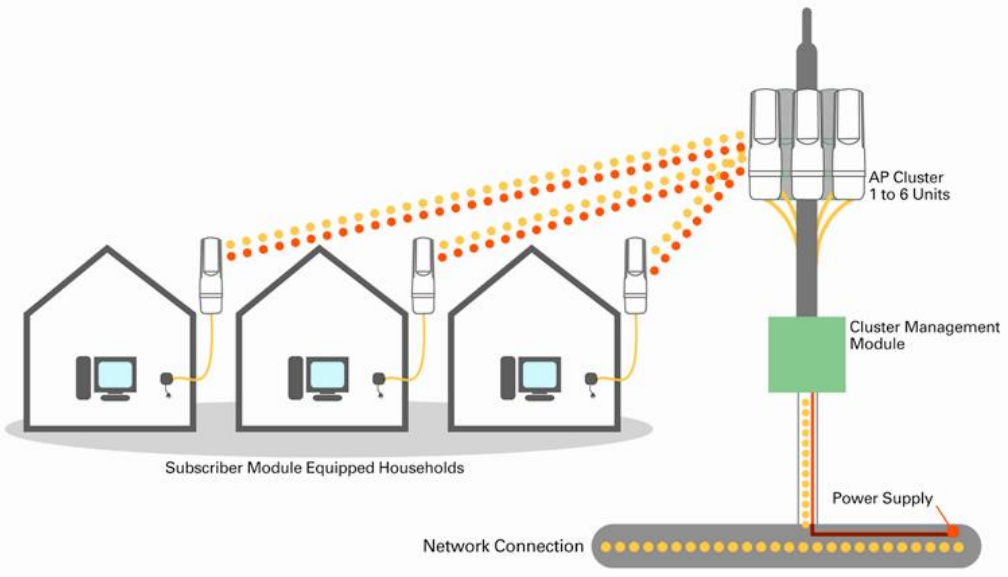


Figura 1.3.6 Red punto Multipunto CANOPY

Tipos de antenas

Hay varios tipos de antenas. Los más relevantes para aplicaciones en bandas libres son:

- Antenas Dipolo
- Antenas Dipolo multi-elemento
- Antenas Yagi
- Antenas Panel Plano (Flat Panel)
- Antenas parabólicas (plato parabólico)

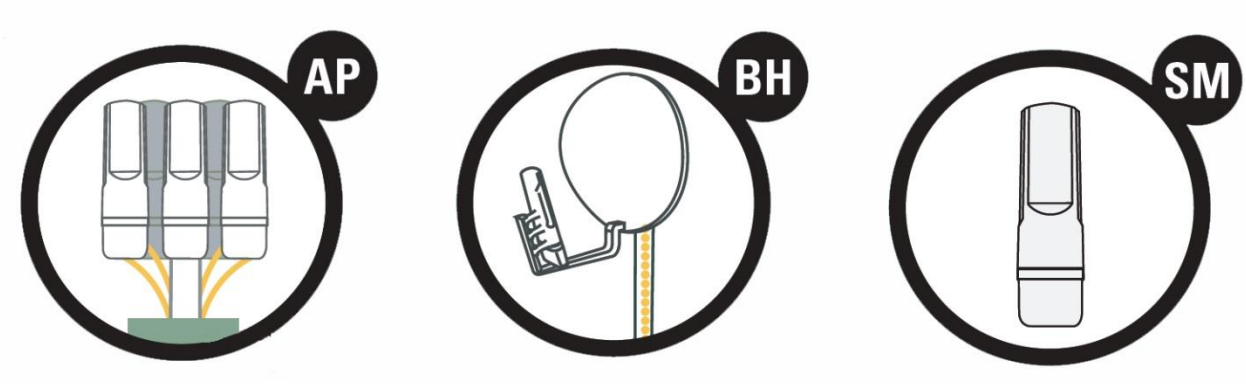


Figura 1.3.7 Solución CANOPY

Antenas CANOPY Motorola

- Módulo AP Canopy
- Módulo SM Canopy
- Módulo Backhaul Canopy

Figura 1.3.7

Relación entre los sistemas

La relación y comunicación entre sistemas se ha ido reduciendo. Pese a lo deseable que sería la comunicación o transferencia de información ente los distintos sistemas esta se ha ido reduciendo debido, entre muchos factores, al uso o desarrollo de sistemas especializados, el crecimiento sin la planeación adecuada y otras carencias.
Ejemplos de ellos son:
Especialización en configuración de equipos. Figura 1.3.8
Programa para establecer enlaces. Figura 1.3.9 ,1.3.10, 1.3.11

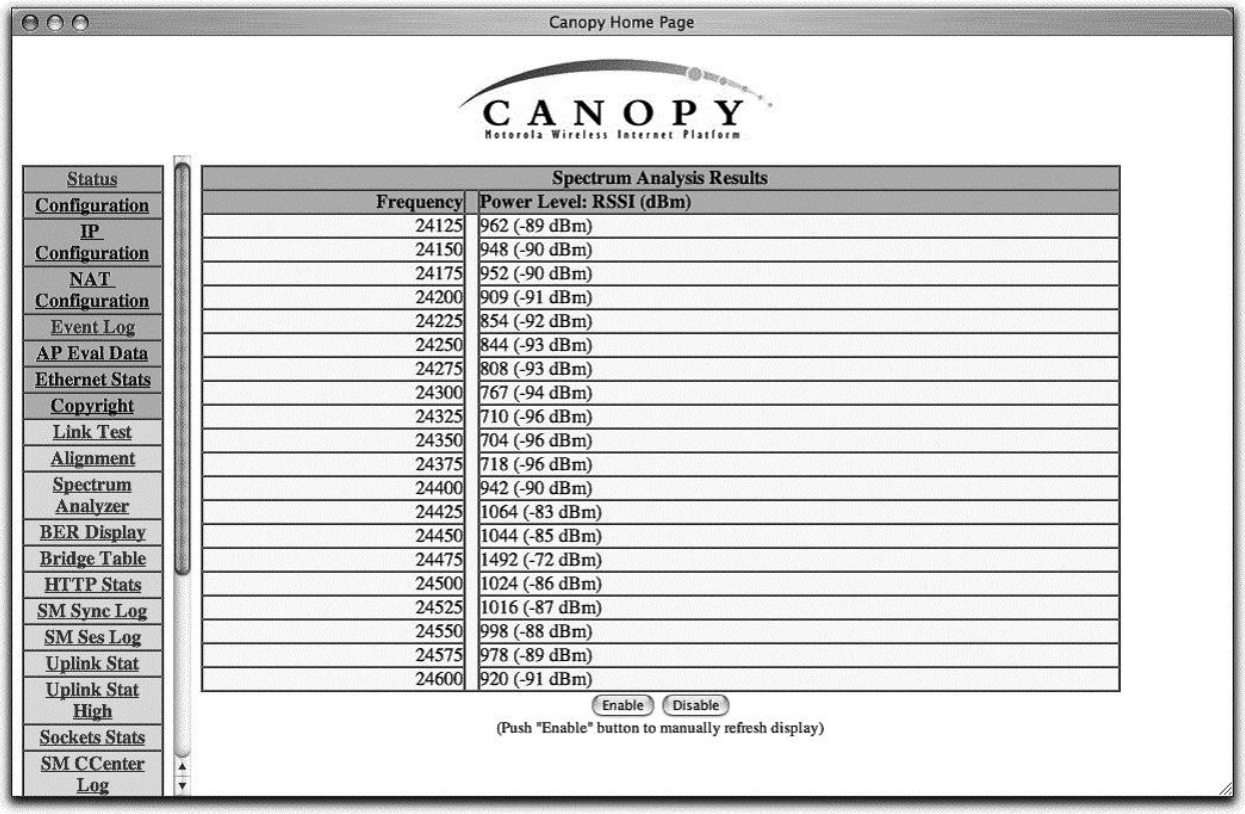


Figura 1.3.8 Programa para configuración

Figura 1.3.9 Software RadioMobile planeación de enlace

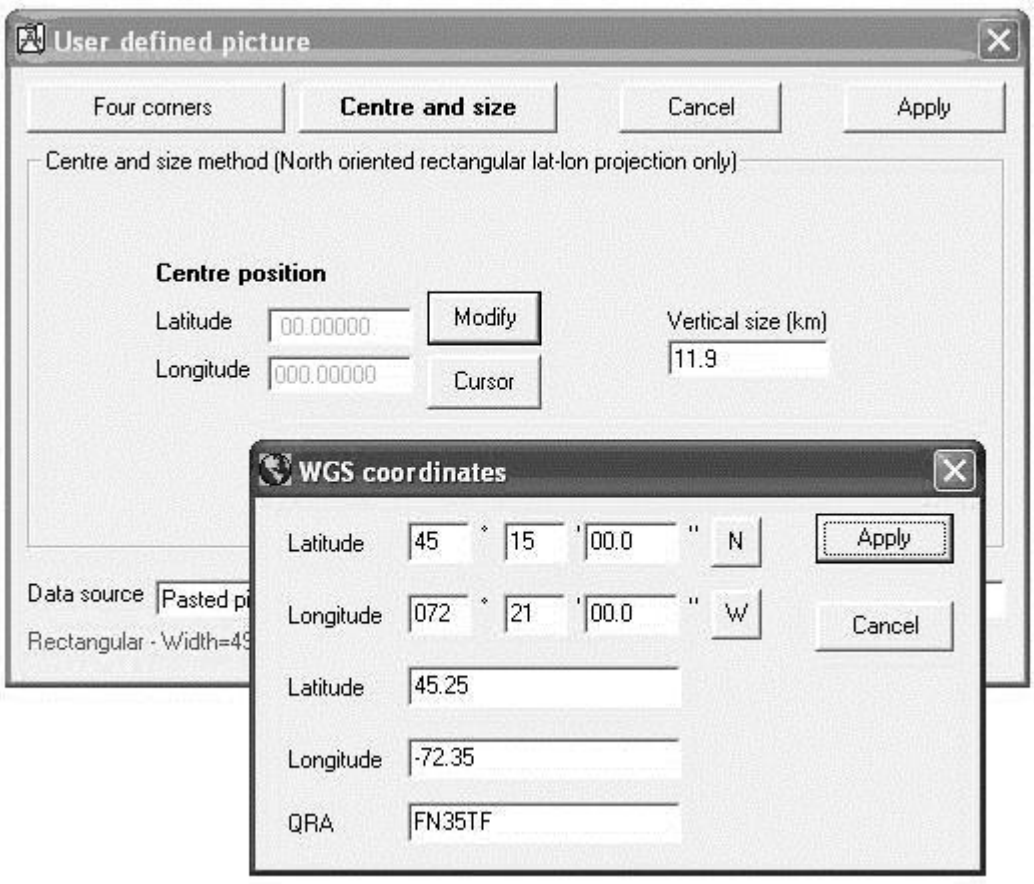


Figura 1.3.10 Software RadioMobile localización

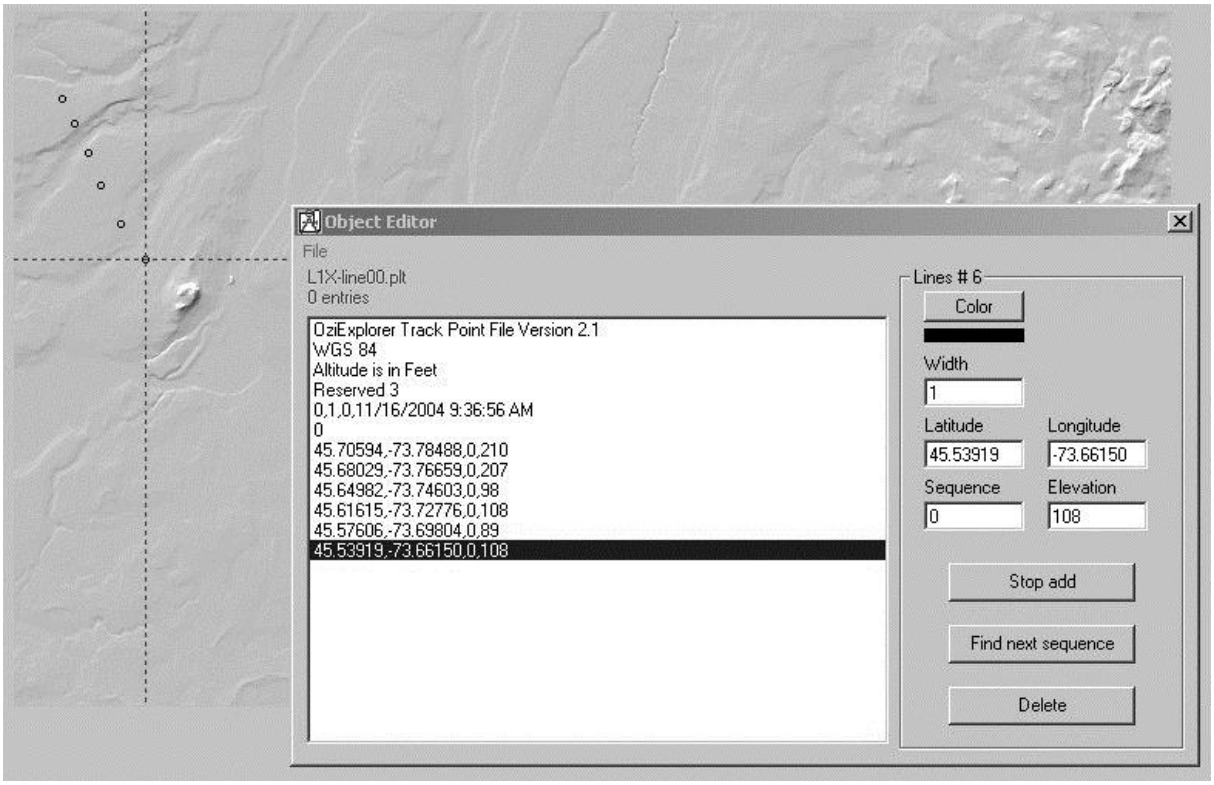


Figura 1.3.11 Localización de Enlaces

Sistema Comisiones
El sistema de comisiones tiene que ser verificado manualmente con el de facturación o directamente con los expedientes de los clientes.
La facturación es afectada por descuentos relativos a las fallas del servicio contratado que tiene que ser determinadas por el área de operaciones y monitoreo o de instalaciones y otorgado por el área comercial y o de gerencia.
El Help Desk
La comunicación de una ventana de mantenimiento se realiza fuera del sistema de help-desk en forma manual.

Los sistemas y procesos de la empresa se encuentran en distintos grados de desarrollo entendiéndose este como el grado de evolución de los procesos e integración de sus sistemas. Así podemos encontrar tanto procesos manuales para los expedientes de los clientes como altamente especializados para el monitoreo de fallas en equipos de comunicación que realizan en forma automática, envíos de alarmas, cambios de configuración para sistemas de respaldo y asignación de encargados de reparación de fallas.

La información en estas circunstancias tiene un distinto grado de adquisición y confianza según el sistema o proceso que la produce o la maneja,

La seguridad

La seguridad al igual de la información existe en diferentes grados de evolución según el entorno en que se maneje. Así encontramos la seguridad en:

- Los sistemas críticos de red
- Los sistemas de los clientes
- La información financiera
- La información del negocio

Los sistemas críticos de red

Se refiere a la protección de las identidades y claves de acceso a los equipos o redes que permiten la operación de la red. Servidores, enrutadores, antenas, switches, computadoras, y programas.

Los sistemas de los clientes

Se refiere a la protección de las identidades y claves de acceso a los equipos o redes que pertenecen a los clientes. Servidores de correo, claves de administración, equipos, firewalls.

La información financiera

Se refiere a la protección de cantidades y magnitudes y cifras que son, por política de la empresa, del conocimiento exclusivo del área de finanzas, préstamos, deudas, facturación de clientes y global, precios, costos, etc.

La información del negocio

Se refiere a la protección de cantidades y magnitudes y cifras que son, por política de la empresa, del conocimiento exclusivo del área de dirección. Contratos, inversiones, asociaciones, precios, productos, infraestructura.

Procesos y manuales de procedimientos

Existen muchos procesos de información dentro de la empresa. Muchos de ellos en desarrollo y otros más establecidos. Entre ellos podemos encontrar

Procesos

- Contratación del servicio
- Help-desk externo / interno
- Monitoreo
- Facturación
- Cobro
- Servicio a Clientes
- Estados financieros
- Compra de equipos
- Ventas
- Contabilidad

Proceso	Sistema	Manual	Ambos
Contratación del servicio			
Help-desk externo / interno			
Monitoreo			
Facturación			
Cobro			
Servicio a Clientes			
Estados financieros			
Compra de equipos			
Ventas			
Contabilidad			

Figura 1.3.12 Procesos y automatización

Se debe considerar el estado que guardan los procesos involucrados en la definición e implementación del sistema de registro de órdenes ya que la adquisición de información puede o no estar disponible o incluso los responsables designados de algunas funciones no se han nombrado aun. Figura 1.3.13, Figura 1.3.14

Prácticamente no existen manuales de procedimiento. La gente encargada de la operación sabe cómo realizarlos y la empresa depende de ellos para realizarlos, generando una enorme dependencia del personal. Esto mismo sucede en el manejo de los programas establecidos.

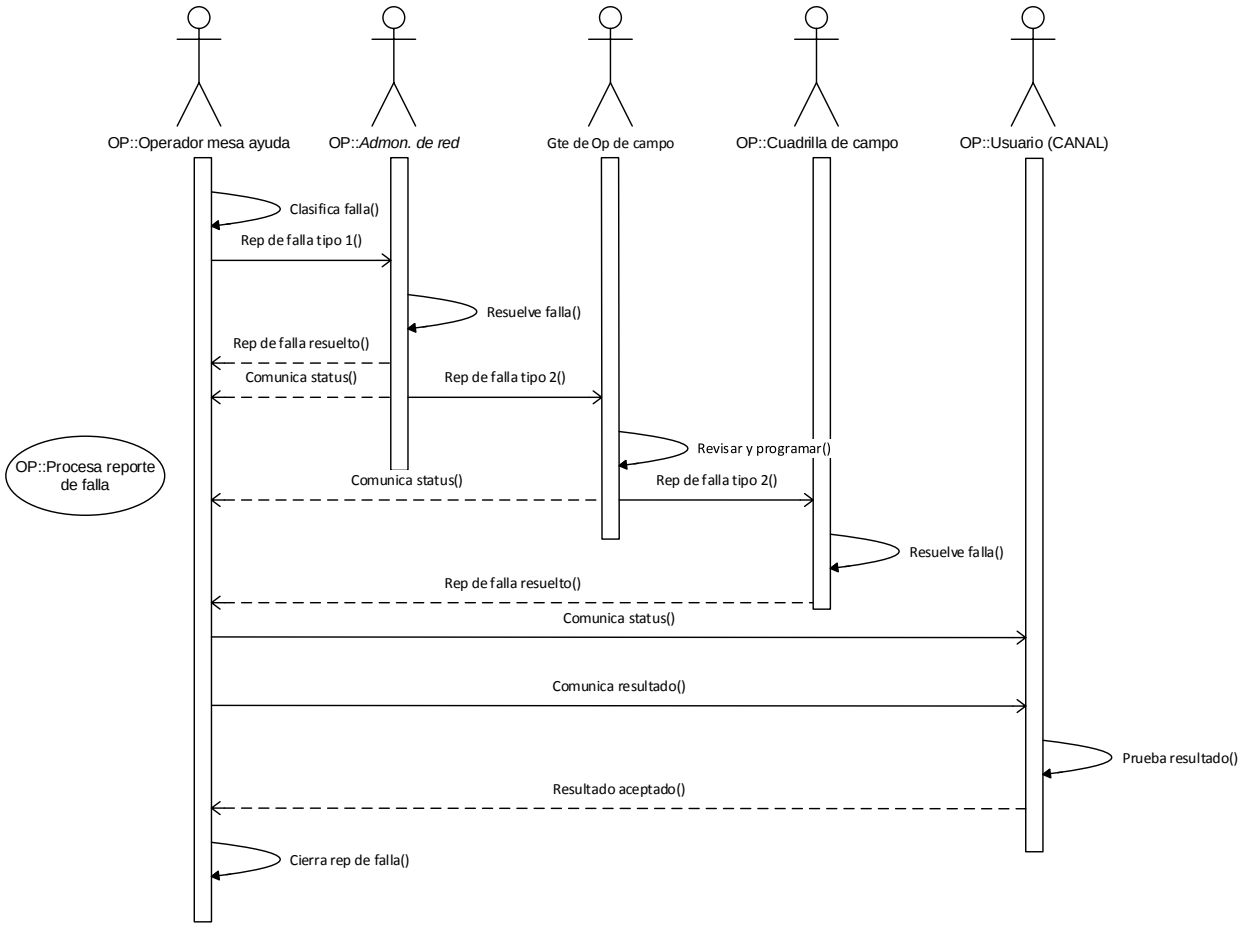


Figura 1.3.13 Proceso de Reporte de Falla

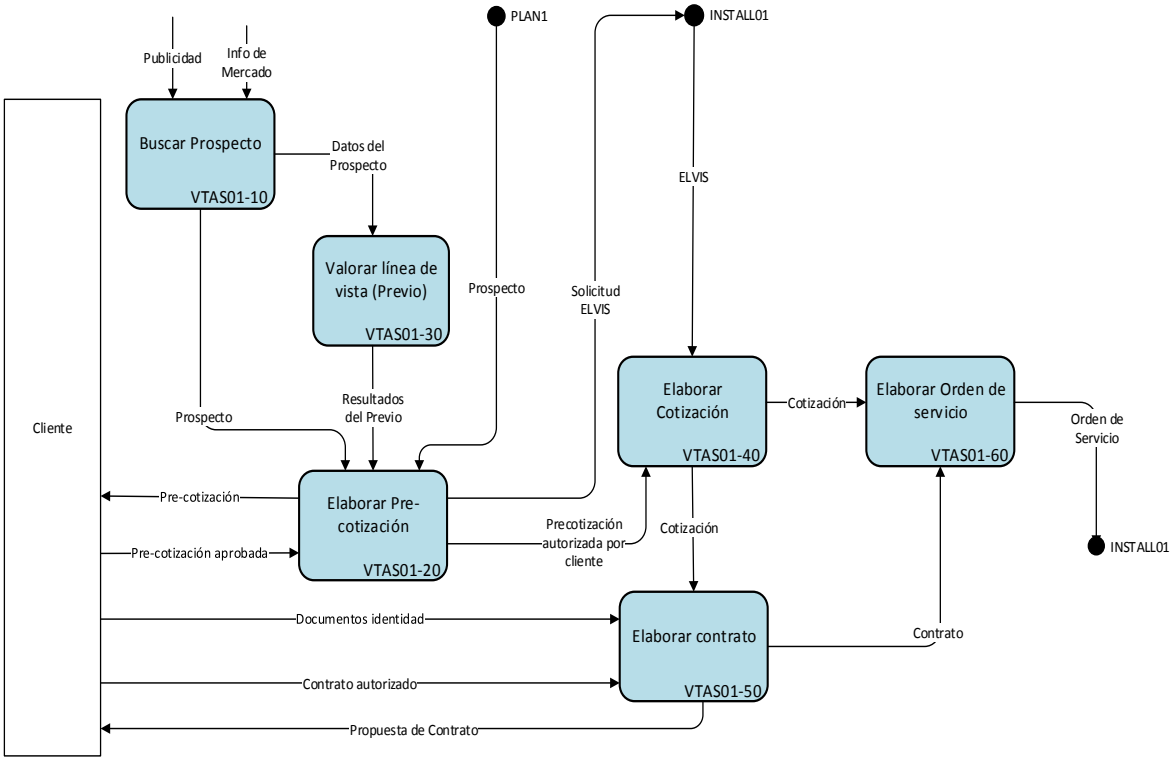


Figura 1.3.14 Proceso Básico de Ventas

Eficiencia

El estado que guardan los procesos de información y los sistemas no permite conocer o realizar de manera rápida y actualizada muchos de los procesos y reportes importantes de la empresa. Sin querer llegar a la automatización completa de todos los procesos o la medición exacta de la eficiencia de los procesos, es claro que existen muchas oportunidades de realizar algunos sistemas y la transferencia de información entre estos, evitando así los errores, el retraso u omisión de información importante que permitiría elevar la eficiencia de los procesos.
Lo pequeño de la organización permite el rápido cambio o corrección de procesos que tienen a la empresa, por ejemplo, como una de las más rápidas en la instalación y entrega de algunos servicios
Si la eficiencia se restringe a solo la velocidad con la que se instala, la empresa no requeriría de un cambio en los procesos o sistema para aumentar esta. Es hasta que se realizan las preguntas comunes a la eficiencia de la operación como son cuantas solicitudes, en donde y cuando se realizan o porque algunas tardan tanto y porque no se ha crecido o la razón de la disminución del crecimiento en compras, clientes, etc. y no se cuenta con la información o está dispersa para contestar estas u otras preguntas, que se aprecia la necesidad de implementar algunos controles o sistemas.

El registro de órdenes de contratación de servicios es un proceso que involucra a muchos de los procesos en la empresa. La información que se genera en cada uno, afecta la eficiencia de los otros. Este proceso, tan importante, no tiene controles automatizados o almacenamiento de información que permitan ver la eficiencia de la operación.
Los reportes de lo que sucede día con día son importantes en el control o manejo de los procesos y permiten tomar decisiones importantes, para la continuación o corrección de la operación.

La baja eficiencia ha costado mucho en términos de atención al cliente, crecimiento, contrataciones y desarrollo de la empresa.

4. Panorama general de la empresa

Como ya se ha dicho anteriormente, la empresa provee servicios de telecomunicaciones y está catalogada como una PYME por el número de empleados, ingresos y las empresas en el sector.

La competencia en el ramo de la empresa es muy alta, existen demasiadas empresas en el mismo negocio o que venden productos y servicios semejantes. De modo que las ventajas competitivas de la empresa radica en la formalidad, el servicio, la eficiencia, los precios, la experiencia.

Nicho del negocio
Las grandes empresas de comunicaciones están concentradas en las grandes ciudades y el negocio "fácil". Entregar servicios en las grandes ciudades y en los lugares establecidos a precios establecidos y en las condiciones o configuraciones más convenientes para el proveedor.

El nicho de negocio es entonces aquellos clientes que requieren servicios:

- Donde no los proporcionan las grandes empresas
- Más baratos
- Especiales
- Para las PYMES
- Más rápido

Para poder realizar lo anterior la empresa necesita explotar las ventajas de ser pequeño

- Implementación rápida de nuevos servicios
- Adecuación rápida para el uso de nueva tecnología
- Asociaciones con empresas del ramo
- Cambios de organización.
- Atención rápida y cercana
- Mejor uso de recursos

Los grandes proveedores dan la pauta entonces para el desarrollo y crecimiento de esta y otras empresas del tipo. La fijación de los precios, por ejemplo, depende de hasta cuanto pueden pagar los clientes por lo que ahora ofrecen o no los grandes proveedores.

Son también los grandes proveedores o carriers proveedores de la empresa. Los carriers determinan la conectividad entre las empresas. La empresa requiere para conectar a sus clientes de conexiones privadas a internet u otras redes. El precio a pagar, contratos y servicios por estos enlaces determina la utilidad de su propia oferta.
Esto constituye por ejemplo el negocio de la *última milla.* Los carriers no pueden entregar servicios que no estén cercanos a las troncales o líneas instaladas y la implementación de la conexión final al cliente puede ser cara o muy tardada
Estos proveedores son bien conocidos y se espera la llegada de mayores competidores, complicando aún más las condiciones de competencia de las pequeñas empresas. Como ejemplo podemos ver algunas de estas ofertas en términos de servicios, soluciones y precios
Ver anexo de Grandes Proveedores

Las tarifas y productos mostrados cambian constantemente, son solo un ejemplo en el tiempo. Se espera que con la reforma en telecomunicaciones la calidad, los precios y los servicios se vean mejorados en gran medida.
De la comparación entre productos, precios, servicios y cobertura es claro que la empresa requiere de especialización y muy alta eficiencia para poder seguir compitiendo. La competencia aumentará y los márgenes se reducirán

Oportunidades
Como se dijo anteriormente la empresa está catalogada como PYME así que podemos encontrar en ella características comunes a este tipo de empresa. Todos estos factores afectan el desarrollo e implementación de aplicaciones y deben tomarse muy seriamente, a saber:

Características de las PYMES
- 9 de cada 10 empresas son consideradas Pymes.
- Son responsables del 50% de la economía nacional.
- 80% de ellas muere antes de cumplir su primer año de vida.
- 65% son de carácter familiar.
- Más del 80% no cuenta con algún tipo de certificación.
- Cerca del 50% no utiliza técnicas en calidad o productividad.
- Solo el 24% maneja alguna licencia o patente.
- 83% no consolida su presencia en el exterior.

Por qué fracasan?
- 43% por errores administrativos
- 24% muere por tropiezos financieros
- 24% por problemas fiscales
- 16% por problemas de ventas y cobranza
- 4% problemas de producción
- 3% problemas con insumos

Errores más comunes que terminan con la vida de las Pymes

Ausencia de cultura empresarial
No plantean su:
- Visión
- Misión
- Valores

Falta de análisis estratégico
- Fortalezas
- Oportunidades
- Debilidades
- Amenazas

Mala administración
- El 43% de las Pymes fracasa por errores administrativos.
- Solamente 2 de cada 10 empresarios está capacitado formalmente para administrar su propia empresa.
- Casi siempre se trata de negocios de un solo dueño que hace las veces de administrador, técnico, comercializador, financiero y fiscalista.

Incompetencia personal
- Capacitación
- Conocimiento a fondo de los productos
- Atención a los clientes
- Mantenerse alerta a los cambios del mercado

Falta de especialización
- Ser todólogo por demasiado tiempo.

- Lo ideal es contratar al personal que se requiere para contabilidad, producción, ventas, recepción, logística, etc.
- Falta de enfoque en el "Core" del negocio.

Mala previsión financiera

- No determinar con anticipación los fondos necesarios para poner en marcha la empresa y cómo obtener el capital.
- No gastar más de lo que se gana.

Adquirir deuda sin previsión

- El crédito no siempre es la medicina correcta.
- Diagnosticar
- Analizar si realmente se necesita el crédito.

Centralizar el poder

- Por cada 6 empresas que se crean en primera generación, solo dos pasan a la segunda y de estas dos que pasan a manos de los hijos solo una llegará a la tercera generación.
- Problemas: vínculos afectivos, la autoridad, el manejo de los recursos, la sucesión, etc.
- Solución: Hay que institucionalizar.
- En México, El 90% de la base empresarial son negocios familiares. América Latina (95%).

Ausencia de controles

Entre más joven sea la empresa, más medidas de control se necesitan.

- Control de gastos
- Control de ventas
- Control de inventario
- Control de producción
- Se necesitan: Manuales de operación

Falta de planeación

- Plantearse que se desea lograr en la empresa y con qué medios se planea alcanzarlo. Ejemplo: Planear las ventas en los próximos 3 años. Primer año-mensual Segundo año-trimestral Tercer año-anual.

Mala comunicación

- La comunicación es la varita mágica que puede abrir o cerrar puertas con un nuevo cliente, mercado o proveedor, así como con los empleados.

Sin pretender hacer un análisis de todas las oportunidades de la empresa Figura 1.4.1, pues excede el objetivo de este trabajo, es conveniente presentar, aunque de manera somera, el ambiente en que se encuentra la empresa. El resultado de esta planeación y análisis, nos puede ayudar a prever las facilidades o dificultades que se presentaran a la hora de realizar el análisis para la definición, construcción, instalación, mantenimiento, etc. del sistema de registro de órdenes.

Figura 1.4.1 Análisis FODA

Con el análisis que se realizo a la empresa y el entendimiento de las condiciones de la misma se recomendó a entrar en un proceso global de ingeniería de procesos y replanteamiento del negocio.

Los resultados de este proceso global, esquemas, nuevos procesos, recomendaciones tendrán que tomarse en cuenta para la modificación de los sistemas actuales o el desarrollo de los nuevos.
Tenemos que considerar entonces, por ejemplo, que no todos los procesos involucrados en el registro de las órdenes estarán listos y que otros no se implementaran completamente y muchas de las funciones no podran ejecutarse en su totalidad.

Determinación de los procesos básicos. Figura 1.4.2

Procesos Básicos

Nivel Estratégico

- Plan de Negocio
- Diseño de Red y plataforma
- Servicio a clientes

Nivel de operación

- Ventas
- Instalación del Servicio
- Activación de servicio

Nivel de soporte

- Administración
- Compra e inventario

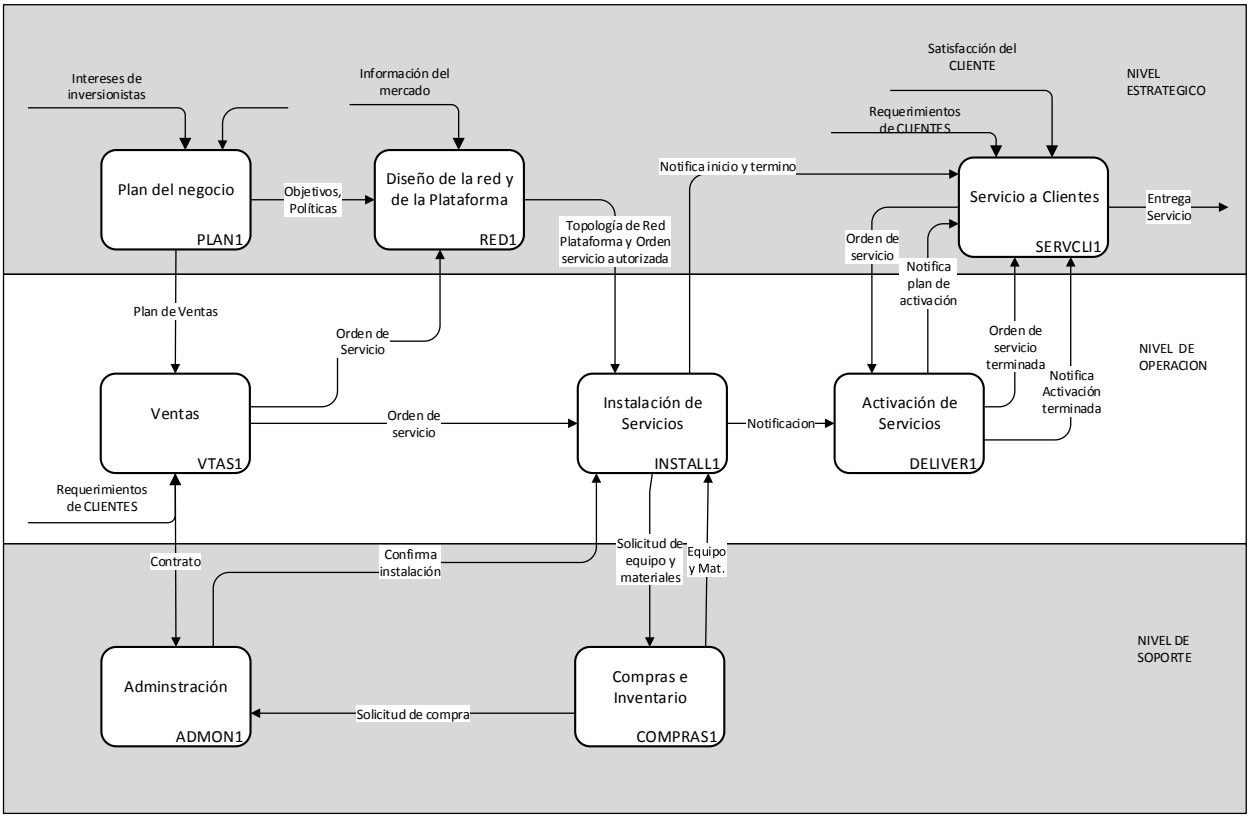


Figura 1.4.2 Procesos Básicos del Negocio

Aunque las funciones no están totalmente definidas nos permite observar como desea la empresa que sean los procesos que realiza.

El esquema nos permite seguir describiendo el panorama general de la empresa

Plan de negocios

Para llevar a cabo las actividades del plan de negocios, que se describirán a detalle más tarde, la empresa:

- Se ha asociado, para lograr mayores ventas con empresas afines y a definido lo que llama Canales de venta.
- Ha contratado e instalado equipos para definir zonas de venta de su interés.
- Ha desarrollado productos y servicios propios y en asociación con otras empresas.
- Ha gestionado permisos, contratos y asociaciones para operar su red ante organizaciones y gobierno.
- Ha especializado las funciones para crear áreas independientes y rentar funciones o contratar servicios.

Servicio a clientes

Es una de las áreas más conflictivas e importantes por el contacto con los clientes. Se han hecho numerosas adecuaciones para tratar de manejar todas las actividades asignadas como las notificaciones a los clientes, el help-desk, los expedientes de los clientes, y el seguimiento de órdenes de servicio y fallas con clientes propios, clientes externos, canales y proveedores. Es sin duda una de las áreas que más atención requiere.

Diseño de red y plataforma, instalación y monitoreo

La especialización requerida para la instalación y configuración de los equipos de comunicaciones y monitoreo a llevado a la empresa a desarrollar especialistas que tienen un alto costo en capacitación y un también una rotación muy alta.
La constante actualización de tecnológica y el alto costo para mantener la plataforma han obligado a la empresa a pensar en la contratación de la operación y monitoreo de la red. El crecimiento de la red es atractivo para las empresas que se dedican a este ramo.
La separación de personal, procesos, información, presupuestos y actividades se está realizando para lograr una unidad independiente.

Administración

El crecimiento de la empresa ha ido rebasando las capacidades y funciones asignadas. Muchas de las funciones administrativas son realizadas por personal de gerencia y se encuentra en proceso de mayor definición de funciones y contratación de personal.
La creación de área de recursos humanos es necesaria para solucionar muchos problemas que aquejan a la empresa.
Existe mucha dependencia de personal, mezcla de funciones y saturación. Muchas de estas deficiencias son arrastradas de procesos anteriores y malas prácticas.
En general el proceso es realizado con la ayuda de sistemas específicos para el control de las operaciones básicas. Proveedores, cliente, cuentas por pagar y cobrar, Manejo de inventarios, facturación, Comisiones, Contabilidad.

Compras e inventario

Los procesos de compras e inventario son manuales. Pese a que los equipos principales tienen un alto costo y están muy bien identificados, la compra de estos o nuevos equipos se realiza cuando la capacidad ha sido superada o se presenta una falla grave. Existe una distinción de compras poco clara entre todo el equipo de cómputo, el de comunicaciones, los implementos para instalaciones y el software requerido. Así para el equipo de cómputo dedicado a las áreas que no son de sistemas el equipo es rentado a una empresa subsidiaria que se supone debía de actualizar los equipos anualmente, para el equipo de cómputo especializado, donde se llevan a cabo controles o configuraciones especiales, se autorizan compras especiales y para el equipo de instalaciones existe un proceso que indica que se deben obtener dos cotizaciones con distintos proveedores y ser autorizada por el gerente o director.

Las políticas de inventario son muy pocas solo los equipos de gran valor están identificados el resto carecen de una política de sustitución o almacenamiento por lo que existen en bodegas y almacenes gran cantidad de equipos obsoletos o a reparar, ninguno tiene un proceso contable de baja. Para los equipos de comunicaciones la obsolescencia llega con la posibilidad o no de manejar nuevas funcionalidades o configuraciones. El total de inversión en equipo es un dato aproximado, se calcula en base a los valores de los equipos más caros.

Las solicitudes, cotizaciones, autorizaciones, facturas, entregas de equipo se encuentran almacenadas en carpetas por día y algunas separaciones elaboradas por el encargado de almacén y compras o quien requirió la compra.

Los equipos de monitoreo poseen la capacidad de mantener números de serie e inventario, pero esta información, si se tiene no es utilizada.

5. Entorno Administrativo

El crecimiento normal de la empresa, sin la previsión y adecuación de los procesos y sistemas ha provocado un ambiente poco organizado y con muchas deficiencias. Si atendemos a la clasificación de la información como contable y administrativa vemos que en existen en ambos dificultades e imprecisiones obstaculizando la labor de administración y control.

Entre los factores que podemos encontrar, además de los que ya se han mencionado en los sistemas, es la falta de claridad en responsabilidades y objetivos de los puestos.
Aunque existe mucho personal calificado en el trabajo que realizan, las habilidades técnicas, administrativas y de comunicación carecen de una capacitación y buen manejo. Así se tiene a personal con conocimientos técnicos excelentes pero carentes de habilidades de comunicación encargado de la entrega al cliente o la recepción de una queja o realizando labores enteramente administrativas como el inventario de los equipos.

La información almacenada en los sistemas es generalmente modificada para cumplir con los ciclos normales de proceso, estas modificaciones son realizadas por fuera en documentos de office (Excel, Word, Access), y una vez más procesadas.
Los ajustes a la contabilidad, conciliaciones, comisiones, nomina, pagos, reembolsos, créditos son muy comunes.
La mayoría de los reportes desde financieros hasta operativos requieren de modificaciones y son presentados con herramientas externas. Es costumbre pedir reportes sobre clientes especiales y desconocer el estatus de algún proceso, si no se realiza un reporte en específico.

La comunicación entre áreas, el ambiente trabajo, la capacitación, el estímulo laboral son elementos importantes para administrar una empresa, sin ellos o en la falta de estos presentan problemas serios en el control de una empresa.

La empresa carece del manejo de la mayoría de estos conceptos, para no ser repetitivo en las carecías de la empresa como se mencionó en las características de las PYMES solo se mencionara algunos que saltan a la vista.

- Existe una rivalidad entre los grupos de trabajo.
- La gerencia tiene poco o nula capacitación en el manejo y selección de personal
- La capacitación existe solo en nuevos equipos y solo en cuestiones específicas y se carece de planes de capacitación gerencial y técnica.
- No existe estímulos laborales ni para la convivencia organizacional.
- Falta de planeación del presupuesto en sus diferentes áreas.
- Centralización de funciones y procesos en dirección y gerencia

El entendimiento de la dirección general de la empresa sobre esta situación motivo la contratación de consultoría que sugirió cambios administrativos y de personal que se espera beneficien en el corto plazo la situación de la empresa.

Los Procesos

Los procesos identificados en la organización permiten observar la operación de la empresa. Figura 1.5.1. Estos procesos los podemos dividir en tres niveles:

- Estratégico
- Operación
- Soporte

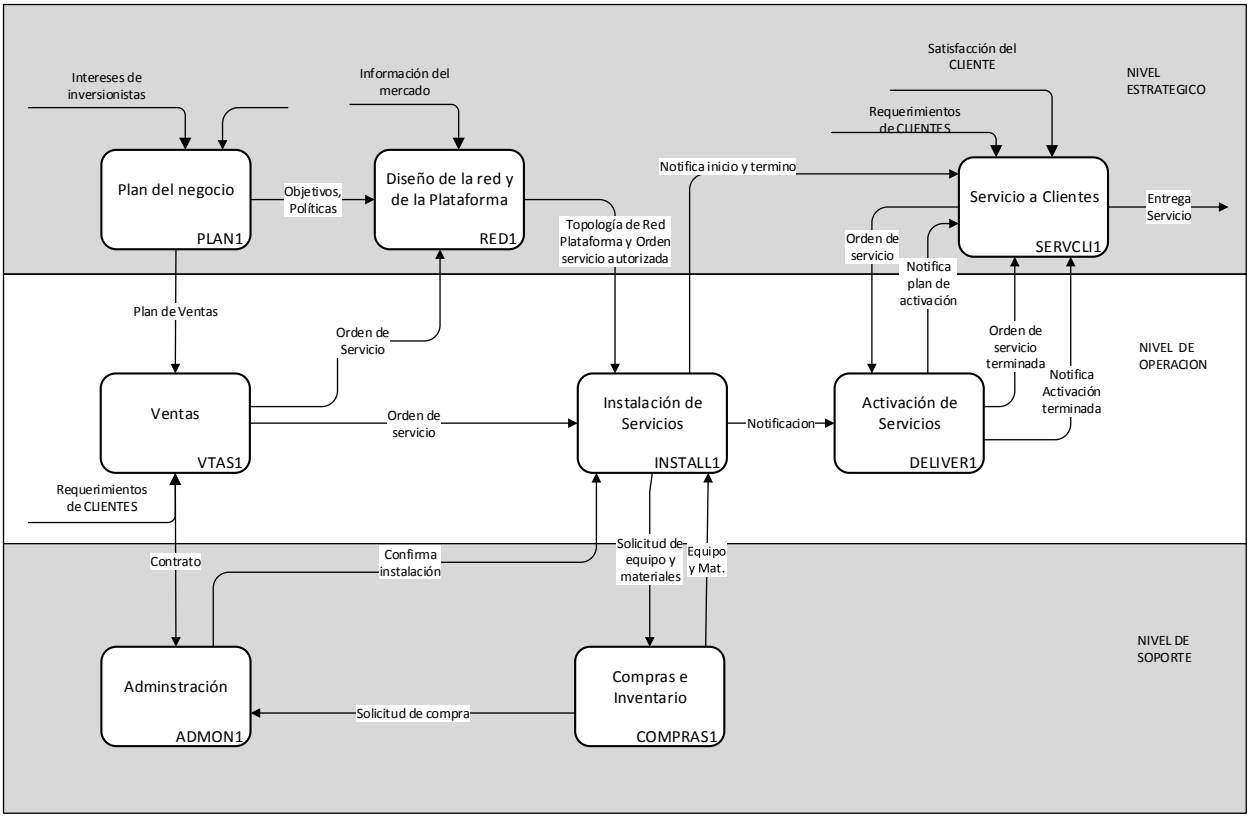


Figura 1.5.1 Descripción de los Procesos Básicos del Negocio

Nivel Estratégico

En este nivel encontramos:

Plan de negocios

De acuerdo con los intereses de los socios e inversionistas así como la información de mercado de alguna zona de interés se definen zonas y estrategias:

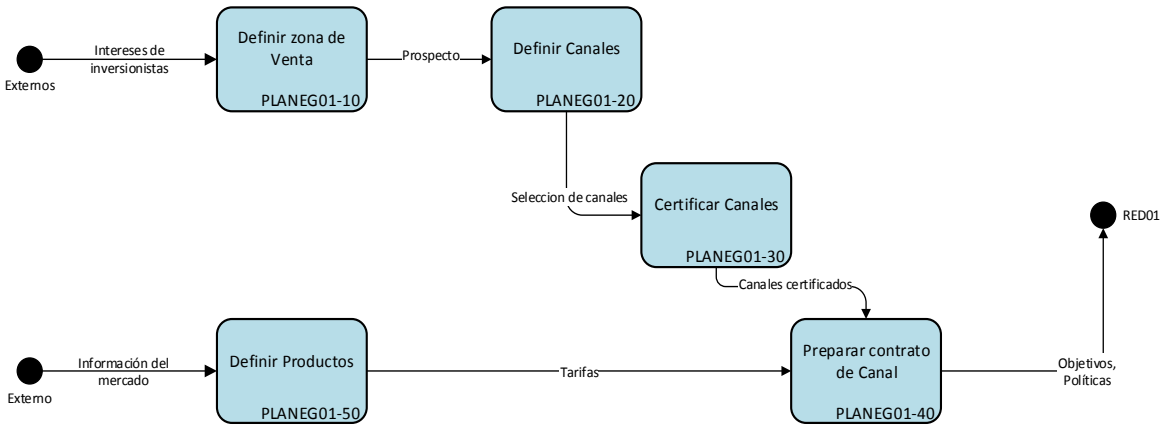


Figura 1.5.2 Plan de Negocio

- Definición zona de venta
- Definición canales de venta
- Definición de los productos
- Integrar contratos

Definición *zona de venta*

Se define el área de interés, principalmente por la cantidad de negocios y las condiciones de las comunicaciones involucradas. Estas zonas son generalmente donde se encuentran, concentraciones de empresas, como son zonas industriales o cercanas a las ciudades grandes.

Se define grosso modo las inversiones y la forma más conveniente de atacar dicho mercado.

Se realiza compra de información y bases datos de clientes potenciales. Se investiga la competencia y los requerimientos legales de licencias y derechos, además se realiza un estudio geográfico y de campo.

Si ya existe infraestructura instalada, se planea la posible adecuación o aumento de infraestructura, en términos de costo.

Definición canales de venta

Se establecen las empresas y la forma en que se hará llegar la oferta de los productos a los clientes incluyendo las características, de consumo esperado y las políticas generales que guiaran la definición de contratos con las empresas referentes a productos y comisiones. Se estipulan los planes de ventas para los canales las facilidades y recursos necesarios para la adquisición o contratación de los clientes, incluidos promociones y publicidad y el tipo de esta.

Definición de los productos
Se selecciona el grupo de productos y los precios a los que se ofertaran. Se incluyen los productos que se promocionaran en conjunto con los canales y los casos de comisiones que estos representarán. Se establece también la asignación de recursos humanos y materiales para la configuración de los productos así como los planes generales para su atención y servicio.

Integración contratos
Se estipula como han de conformarse los contratos en términos de costo, planes de pagos, descuentos, paquetes términos y condiciones de operación, recisión, reembolso, penalizaciones, comisiones, márgenes y políticas de atención y prioridades.

Servicio a clientes
Los procesos definen la relación con el cliente en cuanto a los requerimientos y sus necesidades. Se encargan de:

- Levantar y documentar Requerimientos
- Clasificar requerimientos
- Gestionar requerimientos
- Notificar al cliente
- Monitorear calidad

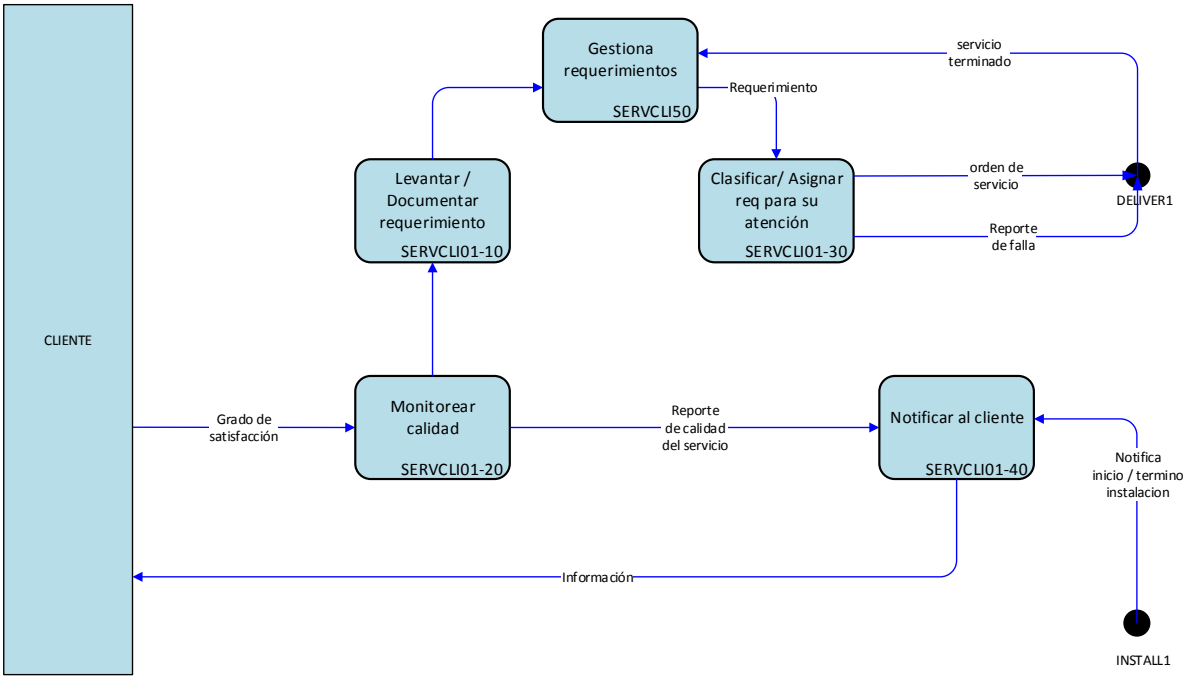


Figura 1.5.3 Servicio a Clientes

Levantar y documentar Requerimientos

Se requiere conocer las necesidades específicas y con qué producto pueden resolverse así como la definición técnica y factibilidad de dicha solución. En este momento se adquiere la información del cliente para la formación del expediente del cliente y todos los datos necesarios para una realizar instalación. El expediente contiene los datos fiscales, de contacto, de comunicaciones, geográficos, etc.

Clasificar requerimientos
De acuerdo a la importancia, las posibilidades de atención, el monto de lo contratado, las fechas establecidas o las condiciones de los clientes, los productos, etc. se establece una prioridad, un periodo de atención o un área de atención

La clasificación de estos requerimientos incluye el manejo de fallas y las prioridades de las mismas.

Gestionar requerimientos
Gestiona los requerimientos a las áreas de instalaciones, monitoreo o servicio.
Si es necesario detallar o establecer soluciones no estándares se gestiona el requerimiento al área en específico. por ejemplo si se duda de una línea de vista , de las instalaciones del cliente o de la capacidad de los equipos se turnan los requerimientos al área de ingeniería quien dará su dictamen técnico inicial para poder establecer una solución que se presentara al cliente.

Notificar al cliente
Si el requerimiento del cliente es factible, se establece la comunicación con el cliente para continuar con el proceso de contratación e instalación. Estos procesos incluyen la firma de los contratos y el plan de instalación
Este proceso también incluye notificar al cliente en caso de fallas y el estatus de sus solicitudes así como informar sobre la calidad del servicio

Monitorear calidad
Establece los parámetros de calidad contratados por los clientes y da seguimiento a las solicitudes y fallas.

Diseño de red y plataforma
Estos procesos inician con la construcción o ampliación de una red y su actualización.

- Estudios para definición de zona
- Configuración de solución
- Autorización de solicitudes

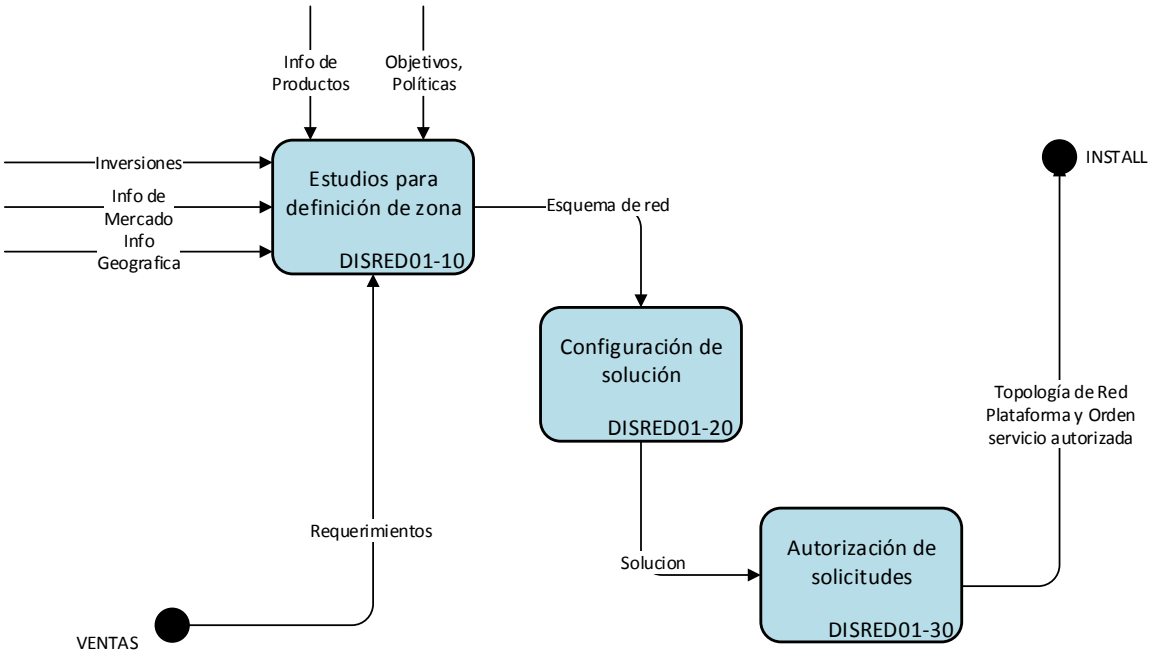


Figura 1.5.4 Diseño de Red y Plataforma

Estudios para definición de zona
Se definen las características técnicas de las redes y los equipos que se utilizaran, así como su localización geográfica y los requerimientos de las instalaciones de los clientes.

Configuración de solución
Establecen las configuraciones de la red, los equipos y el software a utilizar para establecer la solución a los requerimientos de una orden de servicio.

Autorización de solicitudes
Autorizan si una solicitud estándar es viable técnicamente. También se autorizan los tiempos de instalación y proceso de entrega.

Nivel de Operación

En este nivel encontramos:

Ventas

Busca cumplir el plan de ventas y las solicitudes de nuevos clientes, además de establecer el contacto con el cliente.

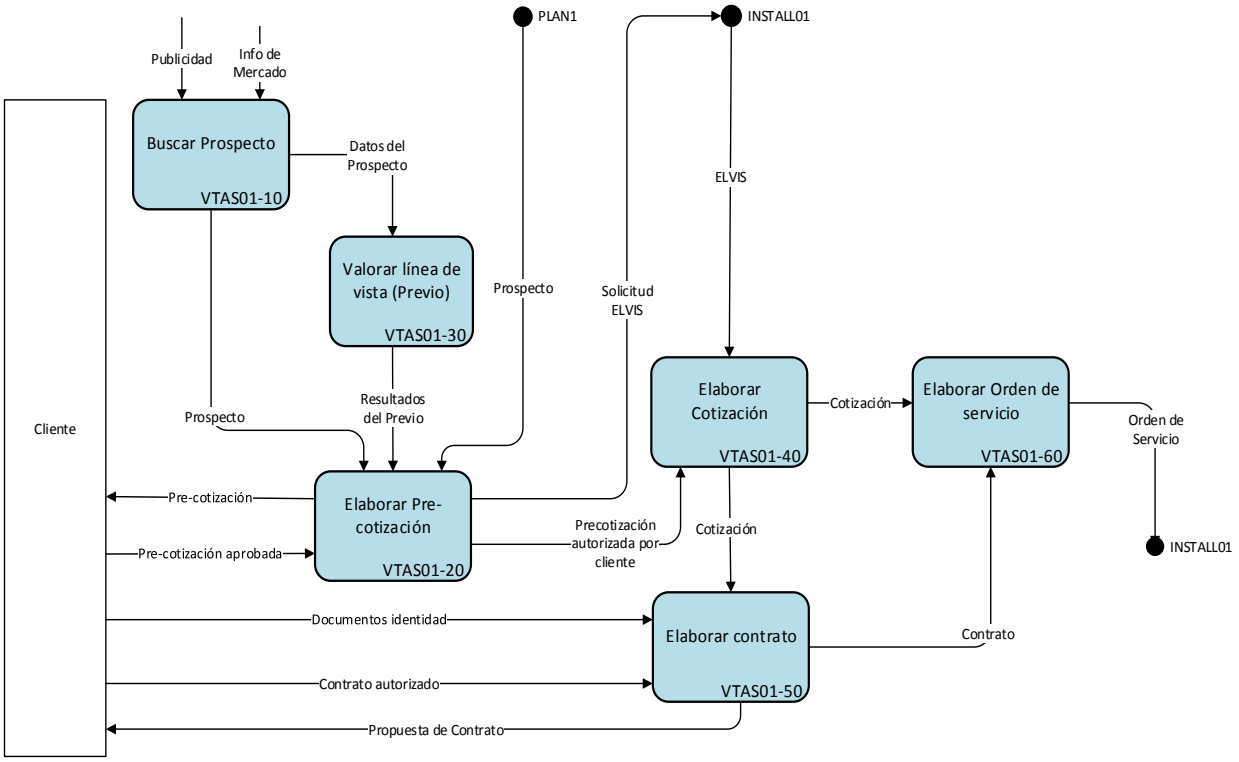


Figura 1.5.5 Proceso de Ventas

Buscar prospectos
Se buscan mediante el uso de bases de datos, visitas a las zonas de interés y la publicidad. Se elaboran las listas de prospectos

Valora previamente instalación
Se realiza una visita para establecer las características técnicas de la instalación como son lugar físico, construcciones, ductos, cuarto de comunicaciones, instalaciones del cliente y en conjunto con la cobertura y el ofrecimiento de servicios se elabora un primer diagnóstico.

Elaborar Pre cotización
De acuerdo con el diagnóstico previo y el producto o servicio seleccionado se establece una primera cotización para ofrecer al cliente.

Elaborar cotización
Una vez aceptado por el cliente el producto ofrecido y validando la solución técnica se establece la cotización del servicio que se presentara al cliente.

Elaborar contrato
Una vez aceptada la cotización por el cliente se establecen las condiciones del servicio en términos contractuales y se le envía el documento basado en los contratos base.

Elaborar orden de servicio
Si el cliente acepta y firma el documento se elabora la orden de servicio. Si requiere modificaciones se elabora nuevamente y se consultan las modificaciones.

Instalación de servicios

Una vez conocidas las órdenes de servicio se inician los procedimientos de instalación según corresponda.

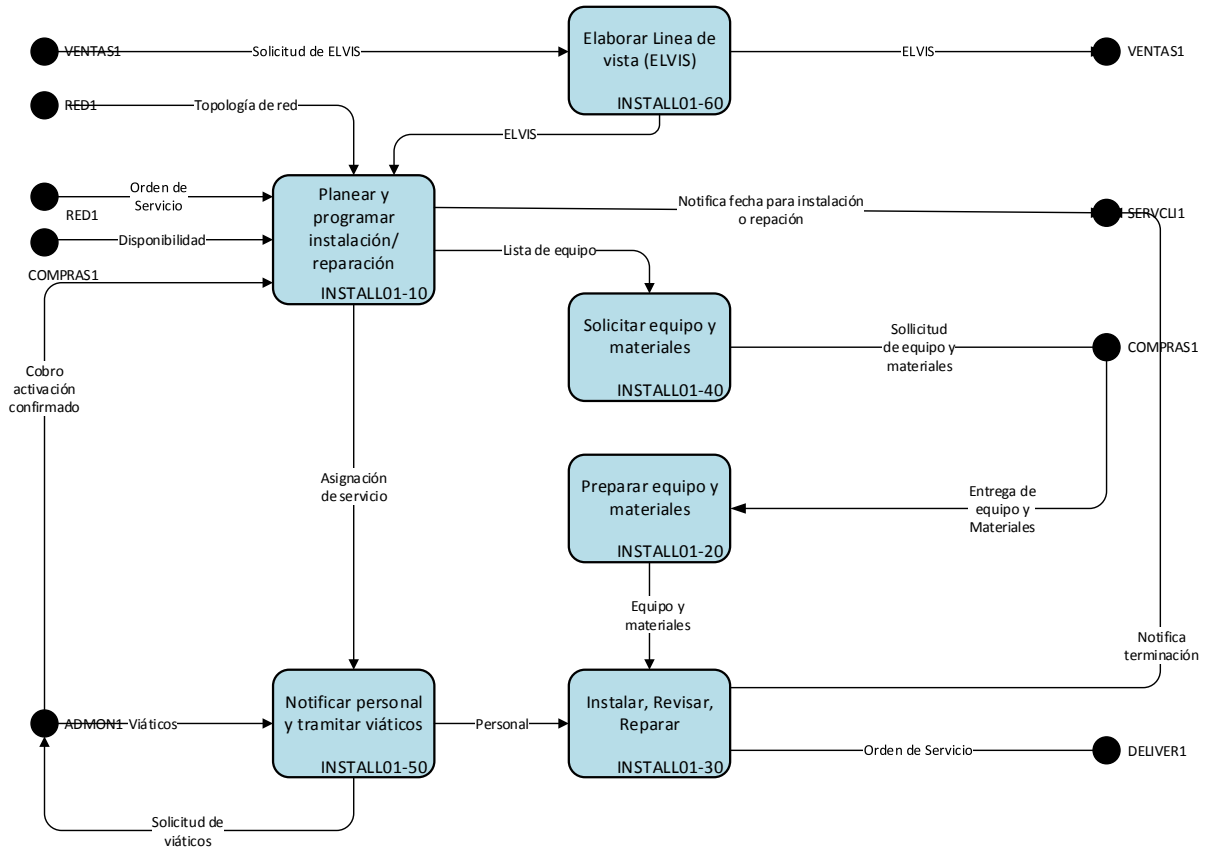


Figura 1.5.6 Instalación de Servicios

Planear y programar instalación / reparación
Se planean las visitas previas, las instalaciones, las revisiones, de acuerdo a las órdenes de servicio y las zonas visitadas, tomando en cuenta el tiempo estimado de instalación y la cantidad de personal requerido.
Se verifica si la instalación requiere de más de una visita y sitio para realizarse.
Durante este proceso también se toman en cuenta los procesos de mantenimiento y revisiones de instalaciones pendientes.

Asignar personal y tramitar viáticos
Se asigna el personal y se tramitan viáticos para personal especializado si así se requiere.
Se requieren de una a seis personas de acuerdo a la dificultad de la instalación.

Solicitar equipo y materiales
Se revisa la solución y se determina el equipo y los materiales y herramientas necesarios
Se solicita al almacén o a compras el material necesario

Preparar equipo y materiales
Se preparan y pre configuran los equipos necesarios. Se alistan para su transporte equipo, materiales y herramientas. Se verifican los planes, esquemas y permisos.

Instalar, revisar, reparar
Cuando se instalan los equipos se sigue un protocolo de instalación. Se platean las trayectorias, se prepara el material, se fijan las estructuras, se instala el equipo , se prenden y conectan, se configura, y se prueba.
Durante las visitas a sitios de acceso se realizan revisiones y mantenimientos necesarios.
Durante las reparaciones, también se efectúan revisiones y mantenimientos asi como las ampliaciones o modificaciones de equipo existente.

Activación de Servicios

Después de la instalación y prueba del equipo se realizan las actividades de activación

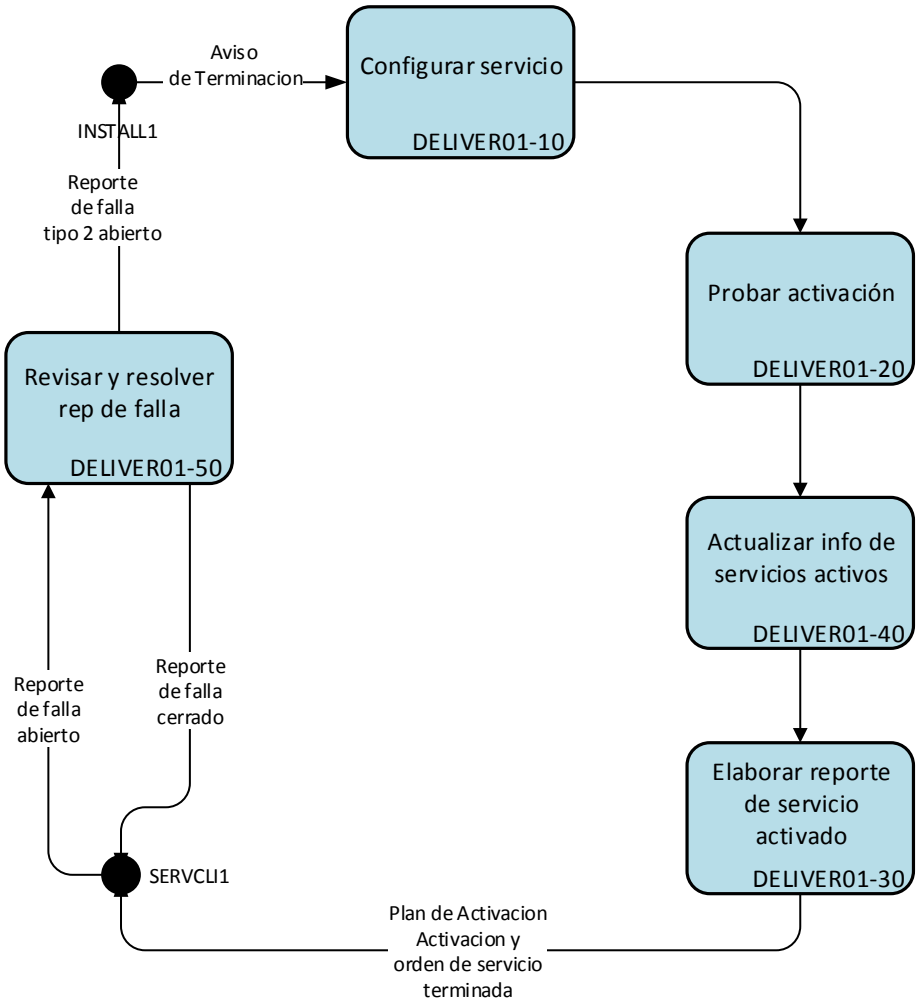


Figura 1.5.7 Activación de Servicios

Configurar servicio
Dependiendo del producto se establecen los parámetros y las condiciones para el mismo. Por ejemplo establecer ancho de banda, direccionamientos de red, usuarios y puesta a punto de equipos.

Probar activación
Se corren las pruebas designadas y se inicia ciclo de verificación. Las pruebas se realizan frente a personal técnico o de recepción del cliente.

Actualizar información de equipos activos
Se realiza un informe con los datos necesarios para la actualización de esquemas, permisos, y monitoreo.

Elaborar reporte de activación
Se prepara el reporte de activación y aceptación del cliente.

Revisar y resolver fallas
Si durante la operación de los equipos se observa una falla o deficiencia se levanta un reporte de falla o revisión. Las fallas y revisiones generan avisos al cliente sobre el estado de estas o se emiten alertas.

Nivel Soporte

En este nivel encontramos:

Administración

Son los procesos que sustentan la operación y base de la información de la empresa

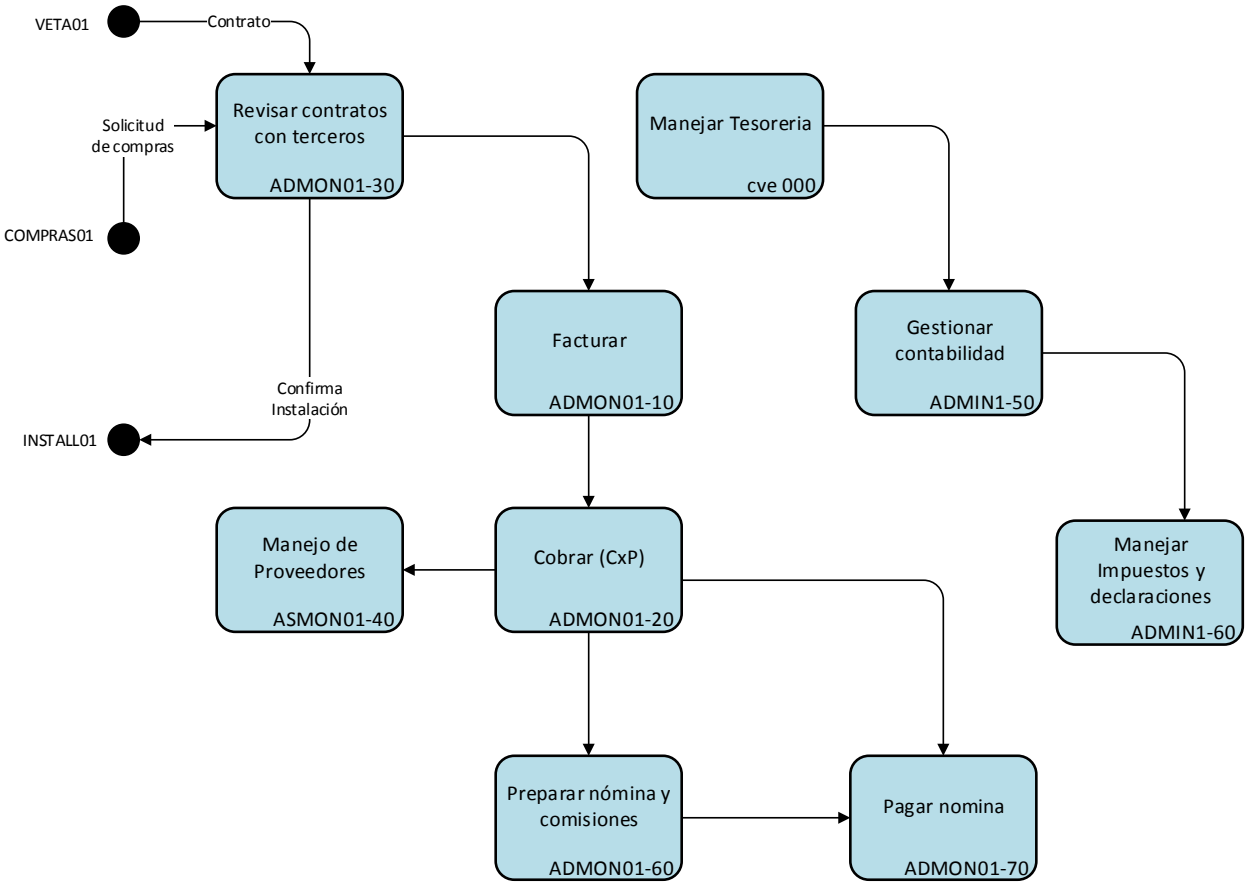


Figura 1.5.8 Procesos en Administración

Revisar contratos
Se verifican las condiciones de los contratos propios y con terceros con:

- Los clientes.
- Los canales de ventas.
- Los proveedores.
- Los empleados.

Facturar
Se elaboran las facturas por los servicio a los clientes. Se integran, tarifas, descuentos, créditos, reembolsos, planes de pago, multas, recargos, etc.

Elaborar cuentas por cobrar
Se establecen los procedimientos para el registro y cobro de las facturas, así como la elaboración de los reportes.

Preparar nómina y comisiones

Se corren los procesos para la elaboración de cheques y transferencias para el pago de los empleados, así como los procesos de corte para la asignación de comisiones a los vendedores y canales.

Pagar nómina y proveedores
Se elaboran los reportes y registros de cuentas por pagar y se realizan las transferencias o pagos a proveedores, nómina y vendedores.

Manejar Tesorería
Se elaboran los reportes y registros

Gestionar contabilidad
Las transacciones son registradas y resumidas para la obtención de los Estados Financieros. Se elaboran los reportes y registros para:

- Registrar las transacciones en el diario general.
- Pasar la información del diario general al mayor general.
- Obtener la balanza de comprobación.
- Registrar los asientos de ajuste.
- Obtener la balanza de comprobación ajustada.
- Formular los Estados Financieros.
- Hacer los asientos de cierre.
- Obtener la balanza de comprobación después del cierre.

Presentar impuestos y declaraciones
Se elaboran los reportes y registros

Compras e inventario

Se controlan los procesos de entrega de equipo y su almacenamiento

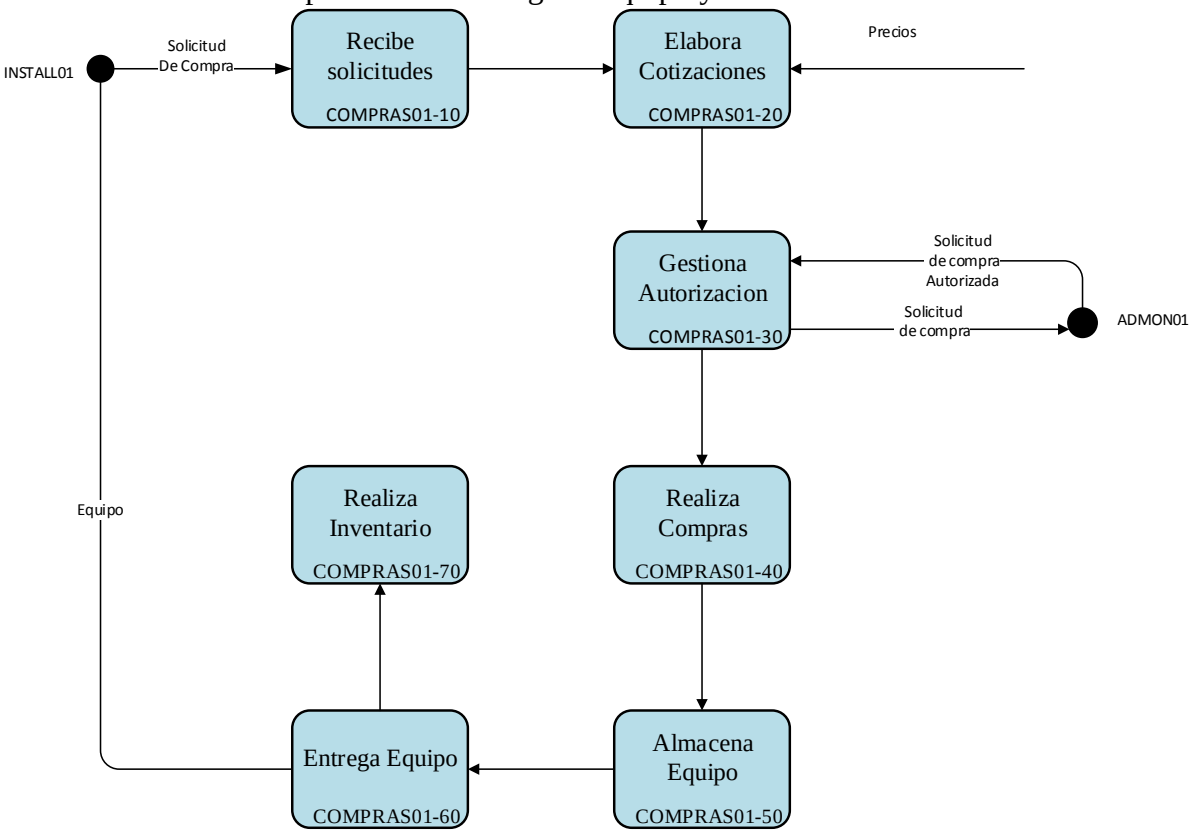


Figura 1.5.9 Compras e Inventario

Recibe solicitudes
Se reciben las solicitudes de compra de equipo, se examina y se envían a los proveedores, si no se tiene el equipo.

Elabora cotizaciones
Se comparan precios y calidad de al menos dos proveedores y se elabora una recomendación de compra

Gestiona autorización
Se solicita a la administración, la autorización para la compra del equipo necesario.

Realiza compras
Con la orden de compra autorizada se realiza la compra con el proveedor.

Almacena equipo
Se avisa los tiempos de entrega y se recibe el equipo para almacenarlo.

Entrega equipo
Se avisa de la llegada del equipo y se entrega, a la firma de recibido, al equipo de instalaciones.

Realiza inventario
Se registran las entradas y salidas de equipos, se evalúa costo.

6. Políticas de la Empresa

Las políticas de una empresa determinan muchas de las características administrativas que debe cumplir un sistema. Estas políticas fijan lo que se debe hacer, como lo debe hacer y lo que no debe hacer el sistema con el fin de alcanzar su objetivo.

Políticas

Las políticas son guías de pensamiento en la toma de decisiones. La función básica de las políticas es dar una dirección unificada a los planes. Es decir, contribuyen a que la organización no se aleje de los objetivos planteados, pero no aseguran que la compañía llegue a donde desea.

Las políticas contribuyen a estructurar los planes canalizando las decisiones operativas. Dichas políticas deben desarrollarse minuciosamente por parte de los directivos y ser perfectamente entendidas por los empleados para que la estructura de planes y procesos sea más efectiva.

La influencia de las políticas sobre la planeación es considerable. Pero también lo es para las otras áreas de la administración y para todas las estructuras de una organización.

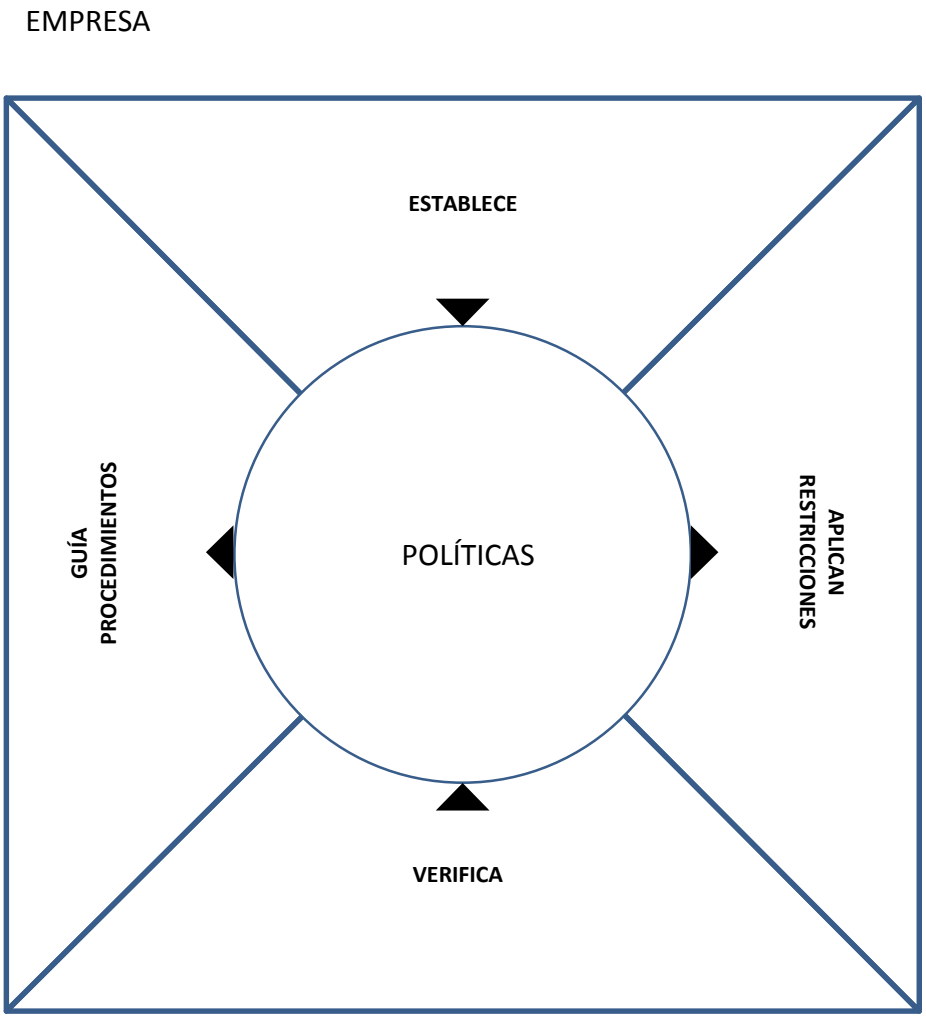


Figura 1.6.1 Política y Empresa

Las políticas bien fundadas y una dirección enérgica, son esenciales para lograr una buena administración. Las políticas bosquejan, significan y representan los principios que guían y ayudan a seguir el curso planeado.

Las políticas deben estar claras, definitivas y distribuidas a todos los interesados, evitaran malas interpretaciones, despilfarros, fricciones y pérdidas de energía.

Las políticas se parecen a las normas, pues representan las mejores ideas proyectadas en algún momento y deben obedecerse, en toda ocasión, hasta que exista un motivo que obligue a cambiarlas. Las políticas no deben ser tan inflexibles como para oponerse a un cambio. Deben modificarse cuando sea necesario para ajustarse a un cambio que beneficie a la empresa u organización.

Para lograr una buena integración del sistema de registro de órdenes con los sistemas establecidos se requiere especificar los procesos que se automatizarán y a qué nivel se realizará

Los procesos involucrados en el manejo de las órdenes de servicio y específicamente las relacionadas con enlaces digitales involucran varias áreas y funciones. La instalación del sistema y su puesta en marcha tendrá que realizarse poniendo atención en no distraer los demás procesos de la empresa.

Procesos Básicos.

Nivel Estratégico

- Plan de Negocio
- Diseño de Red y plataforma
- Servicio a clientes

Nivel de operación

- Ventas
- Instalación del Servicio
- Activación de servicio

Nivel de soporte

- Administración
- Compra e inventario

Plan de Negocio
El sistema contará con información para modificar muchas de las políticas y planes, como son las de instalación, servicio al cliente, contratos.
Permitirá ver y obtener reportes del manera global y específica y de utilidad para el nivel de la gerencia.

Diseño de Red y Plataforma
Se tendrán más elementos para la definición de zonas, atención a las órdenes y reportes.
Se podrá acceder solo a la información específica para la función de las áreas.

Servicio a Clientes

Permitirá aumentar la calidad en el proceso de monitoreo de atención al cliente. Mejorar el registro y documentación de requerimientos y mayor agilidad en la clasificación de requerimientos.

Ventas
Permitirá un mejor seguimiento de órdenes y la visualización de ventas totales. Se planea que sea donde se registre el inicio de captura de información. Deberá contar con información básica.

Instalación del Servicio
Permitirá un mejor seguimiento de proceso de instalación verificando como y donde se encuentra una o varias órdenes. Permitirá la liberación de recursos en la atención a las demás áreas.
El sistema presentara el estatus del proceso.

Activación de Servicio

Permitirá un mejor seguimiento de proceso de activación verificando como y donde se encuentra una o varias órdenes. Permitirá la liberación de recursos en la atención a las demás áreas.
El sistema presentara el estatus del proceso ya sea en detalle o en general mediante tres estatus básicos Aceptado, en procesos, terminado.
El sistema no proporcionara información sobre fallas.

Administración

Los procesos de administración no se considera que serán afectados por el sistema, salvo por algunas verificaciones del estatus de instalaciones para facturación o pago de comisiones, pago a proveedores y contratos que serán solo de referencia.

Compra e inventario
Se considera que los procesos de compras e inventario no serán afectados por el sistema. El sistema en esta etapa no considerara aun el manejo de información los procesos. El sistema servirá de referencia para la instalación del equipo.

En general
El sistema considerara el acceso desde la red y mediante acceso restringido a las funciones del sistema.

2. *MARCO TEÓRICO*

De acuerdo con los requerimientos del sistema y las características solicitadas por el cliente, se determinó que debían usarse algunas de las siguientes herramientas de desarrollo.

1. Programación ASP, VJ, VB, HTML

Para la programación de una aplicación web se requiere del uso de diversas herramientas del lado del servidor y del lado del cliente. Además se requiere de la infraestructura de programación o ambientes de trabajo y plataformas de desarrollo. Empezando por una plataforma de comunicaciones, un servidor, un sistema operativo, un servidor web, un manejador de base de datos, una plataforma de desarrollo, uno o múltiples lenguajes o intérpretes, herramientas de mantenimiento, plataformas de seguridad, etc. A continuación se describen algunos de estos elementos, tratando de mencionar los aspectos más destacados.

ASP.NET

Historia
Es un marco de desarrollo para aplicaciones web desarrollado por Microsoft. Es usado por programadores y diseñadores para construir sitios web dinámicos, aplicaciones web y servicios web XML. Apareció en enero de 2002 con la versión 1.0 del .NET Framework, y es la tecnología sucesora de la tecnología Active Server Pages (ASP). ASP.NET está construido sobre el Common Language Runtime, permitiendo a los programadores escribir código ASP.NET usando cualquier lenguaje admitido por el .NET Framework.
Es parte del Internet Information Server (IIS) desde la versión 3.0 y es una tecnología de páginas activas que permite el uso de diferentes scripts y componentes en conjunto con el tradicional HTML. El prototipo inicial fue llamado "XSP"; El desarrollo inicial fue hecho usando Java pero pronto se decidió construir una nueva plataforma sobre el Common Language Runtime (CLR), ASP.NET 1.0 fue liberado el 5 de enero de 2002

Características

Las páginas Web ASP.NET están completamente orientadas a objetos. Se puede trabajar con elementos HTML que usen propiedades, métodos y eventos. Se elimina los detalles de implementación relacionados con la separación de cliente y servidor inherente a las aplicaciones Web presentando un modelo unificado que responde a los eventos de los clientes en el código que se ejecuta en el servidor. El marco de trabajo también mantiene automáticamente el estado de la página y de los controles que contenga durante el ciclo vital de procesamiento de la página.

Las páginas de ASP.NET, conocidas como "web forms" (formularios web), son el principal medio de construcción para el desarrollo de aplicaciones web. Los formularios web están contenidos en archivos con una extensión ASPX; en jerga de programación, estos archivos típicamente contienen etiquetas HTML o XHTML estático, y también etiquetas definiendo Controles Web que se procesan del lado del servidor y Controles de Usuario donde los desarrolladores colocan todo el código estático y dinámico requerido por la página web.

Controles de usuario
ASP.NET permite la creación de componentes reutilizables a través de la creación de Controles de Usuario (User Controls). Un control de usuario sigue la misma estructura que un formulario

web, excepto que los controles derivan de la clase System.Web.UI.UserControl, y son almacenados en archivos ASCX. Como los archivos ASPX, un ASCX contiene etiquetas HTML o XHTML, además de etiquetas para definir controles web y otros controles de usuario. También pueden usar el modelo code-behind.

Los programadores pueden agregar sus propias propiedades y métodos y manejadores de eventos. Un mecanismo de eventos en burbuja proporciona la capacidad de pasar un evento disparado por un control de usuario a la página que lo contiene.

Administración del estado
Las aplicaciones ASP.NET son alojadas en un servidor web y se tiene acceso a ellas mediante el protocolo sin estado HTTP, que no guarda ninguna información sobre conexiones anteriores. Por lo tanto, si la aplicación requiere interacción entre conexiones, tiene que implementar su propia administración del estado. ASP.NET proporciona varias maneras de administrar el estado de las aplicaciones ASP.NET.

Estado de la aplicación
El estado de la aplicación (Application state) es una colección de variables definidas por el usuario que son compartidas por todas las invocaciones de una aplicación. Estas son establecidas e inicializadas cuando el evento Application_OnStart se dispara en la carga de la primera instancia de las aplicaciones y están disponible hasta que la última instancia termina. Las variables de estado o variables de sesión de la aplicación son identificadas por nombres

Estado de la sesión
El estado de la sesión (Session state) es una colección de variables definidas por el usuario, las cuales persisten durante la sesión de un usuario. Estas variables son únicas para diferentes instancias de una sesión de usuario, y son accedidas usando la colección Session. Las variables de sesión pueden ser preparadas para ser automáticamente destruidas después de un determinado tiempo de inactividad, incluso si la sesión no ha terminado. Del lado del cliente, una sesión de usuario es identificada por una cookie o codificando el ID de la sesión en la misma URL

ASP.NET proporciona tres modos de persistencia para variables de sesión:

InProc
Las variables de sesión son mantenidas dentro del proceso. Sin embargo, en este modo, las variables son destruidas cuando el proceso es reciclado o terminado.

StateServer
En este modo, ejecuta un servicio de Windows separado que mantiene las variables de estado. Como esta administración de estado ocurre fuera del proceso, tiene un impacto negativo en el rendimiento, pero permite a múltiples instancias compartir el mismo estado del servidor, permitiendo que una aplicación pueda tener su carga balanceada y escalada en múltiples servidores. También, como el servicio de administración del estado se ejecuta independiente, las variables pueden persistir a través de las finalizaciones del proceso.

SqlServer
En este modo, las variables de estado son almacenadas en un servidor de base de datos, accesible usando SQL. Las variables de sesión pueden persistir a través de finalizaciones de procesos también en este modo.

Estado de la vista
El estado de la vista (View state) se refiere al mecanismo de administración de estado a nivel de página, que es utilizado por las páginas HTML generadas por las aplicaciones para mantener el

estado de los controles de los formularios web y los widgets. El estado de los controles es codificado y mandado al servidor en cada envío del formulario en un campo oculto conocido como __VIEWSTATE. El servidor envía de regreso las variables para que cuando la página sea renderizada de nuevo, los controles volverán a su último estado. Del lado del servidor, la aplicación puede cambiar el estado de la vista, si los resultados del procesamiento actualizan el estado de cualquier control. El estado de los controles individuales son decodificados en el servidor, y están disponibles usando la colección ViewState

Motor de plantillas
Debido a que el .NET framework es orientado a objetos y permite la herencia, muchos desarrolladores podrían definir una nueva clase que herede desde "System. Web. UI.Page", escribir métodos en ella que renderizen HTML, y entonces hacer las páginas en su aplicación que hereden de esta nueva clase. Mientras esto permite que los elementos comunes estén dentro de un sitio, agrega complejidad y mezcla código fuente con lenguaje de marcado. Además, este método puede ser visto solamente al ejecutar la aplicación, no mientras se está diseñando. Otros desarrolladores han usado archivos incluidos y otros trucos para evitar la implementación de enlaces de navegación y otros elementos en cada página.

El concepto de página maestra (Master Page), permite el desarrollo de páginas basado en plantillas web. Una aplicación web puede tener una o más páginas maestras, las cuales pueden ser anidadas. Las plantillas maestras contienen controles contenedores, llamados ContentPlaceHolders para indicar dónde irá el contenido dinámico, además de HTML y JavaScript que será compartido a través de las páginas hijas.

Las páginas hijas también usan esos controles ContentPlaceHolder, que deben ser relacionados con el ContentPlaceHolder de la página maestra que contiene a esta página hija. El resto de la página está definido por las partes compartidas de la página maestra. Todo el lenguaje de marcado y controles de servidor en la página de contenido deben ser colocadas dentro del control ContentPlaceHolder.

Cuando una solicitud es hecha por una página de contenido, ASP.NET mezcla la salida de la página de contenido con la salida de la página maestra, y envía el resultado al usuario.
La página maestra permanece completamente accesible a la página del contenido. Esto significa que la página de contenidos puede manipular los encabezados, cambiar el título, configurar la cache, etc. Si la página maestra expone propiedades públicas o métodos, el contenido de la página puede utilizar estos también.

Otros archivos

Extensión	Versión requerida	Descripción
asax	1.0	Global.asax, usada para la lógica a nivel de aplicación
ascx	1.0	Controles de usuario web: Controles personalizados para ser colocados en páginas web
ashx	1.0	Manejadores HTTP personalizados
asmx	1.0	Páginas de servicios web
axd	1.0	Cuando está habilitado en el web.config la solicitud de trace.axd genera trazas de salida a nivel de aplicación. También es usado para el manejador especial webresource.axd que permite a los desarrolladores de controles/componentes empacar un control/componente con imágenes, script, CSS, etc. para el desarrollo de un archivo único (un 'ensamblado')
browser	2.0	Archivos de capacidades del navegador almacenadas en formato XML; incluye muchos de estos por defecto, para admitir a los navegadores web comunes. Estos especifican que navegadores tienen que capacidades, puede automáticamente personalizar y optimizar su salida de acuerdo al navegador. Los archivos especiales .browser están disponibles en descarga libre para manejar, por ejemplo, el validador de la W3C. Reemplaza la sección BrowserCaps que se encontraba en el archivo machine.config en ASP.NET
config	1.0	web.config es el único archivo en una aplicación web específica que usa esta extensión por defecto (machine.config tiene efectos similares en un servidor web y todas las aplicaciones en el), sin embargo proporciona la facilidad de crear y utilizar otros archivos config. Son almacenados en formato XML
cs/vb	1.0	Archivos de código fuente (cs indica C#, vb indica Visual Basic). Los archivos code-behind predominantemente tienen la extensión ".aspx.cs" o ".aspx.vb" para los dos lenguajes más comunes. Otros archivos de código (que frecuentemente contienen bibliotecas de clases) pueden también existir en las carpetas web con las extensiones cs/vb.
dbml	3.5	Archivo de clases de datos LINQ a SQL
master	2.0	Archivo de página maestra
resx	1.0	Archivos de recursos para localización y globalización. Los archivos de recursos pueden ser globales (por ejemplo, mensajes) o locales, que están hechos específicamente para un solo archivo aspx o ascx.
sitemap	2.0	Archivos de configuración sitemap
skin	2.0	Archivos de temas
svc	3.0	Archivos de servicio de Windows Communication Foundation

Figura 2.1.1 Otros Archivos ASP

Modelos de programación en ASP.NET

ASP.NET soporta tres modelos de programación: ASP.NET Web Forms, ASP.NET MVC y ASP.NET Web Pages. Aunque los tres modelos de programación se ejecutan sobre la misma base de ASP.NET, cada uno de ellos estructura la aplicación de maneras completamente distintas, promueve metodologías de desarrollo diferentes y se adapta a perfiles de desarrolladores distintos. Algunas características que son virtudes en unos modelos de programación, pueden ser consideradas debilidades en el otro..

Es importante recalcar que el hecho de elegir uno de los modelos no excluye necesariamente a los otros, sino que es posible tener aplicaciones "híbridas" y en muchos casos tendrá todo el sentido desarrollar ciertas partes de la aplicación con un modelo de programación y otras partes con otro modelo distinto.

ASP.NET Web Forms
Fue el primero de los tres modelos de programación en existir, y proporciona un gran nivel de abstracción con un modelo de programación familiar basado en eventos y controles que favorece la productividad mediante la programación declarativa reduciendo la cantidad de código necesaria para implementar una determinada funcionalidad.

ASP.NET MVC

Se concibió como alternativa a Web Forms y proporciona un modelo de programación basado en el popular patrón de arquitectura MVC. Entre sus principales características destacan su completa integración con pruebas unitarias y su separación más clara entre la lógica de presentación, la lógica de negocio y la lógica de acceso a datos.

ASP.NET Web Pages

Es el más reciente de los tres modelos de programación, y fue creado como respuesta a una creciente demanda de desarrolladores web sin experiencia previa con ASP.NET, cuya iniciación en ASP.NET Web Forms o MVC les suponía una inversión inicial de tiempo demasiado grande. Web Pages proporciona un modelo de programación más simple y rápida de aprender, sin renunciar a toda la funcionalidad y flexibilidad de ASP.NET.

Compilador de ASP.NET

Compila todo el código de ASP.NET, lo que permite el establecimiento inflexible de tipos, las optimizaciones de rendimiento y el enlace en tiempo de compilación, entre otras ventajas. Una vez que se ha compilado el código, el Common Language Runtime compila una vez más código de ASP.NET en código nativo, lo que permite un mayor rendimiento.

ASP.NET incluye un compilador que compilará todas las componentes de la aplicación, incluidas las páginas y los controles, en un ensamblado que el entorno de host de ASP.NET puede utilizar a continuación para atender las solicitudes del usuario

Infraestructura de seguridad

Además de las características de seguridad de .NET, ASP.NET proporciona una infraestructura de seguridad avanzada para autenticar y autorizar el acceso de los usuarios y realizar otras tareas relacionadas con la seguridad. Puede autenticar usuarios con la autenticación de Windows suministrada por IIS o puede administrar la autenticación con su propia base de datos de usuario

utilizando la autenticación mediante formularios ASP.NET y la suscripción ASP.NET. Además, puede administrar la autorización a las capacidades e información de su aplicación Web mediante los grupos de Windows o su propia base de datos de funciones personalizada utilizando las funciones de ASP.NET. Resulta fácil eliminar, agregar o reemplazar estos esquemas dependiendo de las necesidades de la aplicación.

ASP.NET siempre se ejecuta con una identidad particular de Windows de modo que puede asegurar su aplicación utilizando las capacidades de Windows como, por ejemplo, las listas de control de acceso (ACL) de NTFS, permisos de la base de datos, etc.

Uso actual del lenguaje
Una aplicación .NET puede ejecutarse de dos formas distintas:

Aplicaciones cliente/servidor: Estas aplicaciones están típicamente en formato de ejecutables compilados. Estos pueden integrar toda la riqueza de una interfaz de usuario, tal es el caso de las aplicaciones de desempeño y productividad, pero no se reúne la lógica de negocio como un recurso que se pueda reutilizar. Además acostumbran ser menos gestionables y escalables que las demás aplicaciones.

Aplicaciones que utilizan el navegador: Dichas aplicaciones están caracterizadas por contar con una interfaz de web rica y muy útil. La interfaz gráfica integra varias tecnologías, las cuales son el HTML, XHTML, scripting, etc.; siempre y cuando el navegador que se esté utilizando soporte estas tecnologías.

ASP.NET incluye:

1. Marco de trabajo de página y controles
2. Compilador de ASP.NET
3. Infraestructura de seguridad
4. Funciones de administración de estado
5. Configuración de la aplicación
6. Supervisión de estado y características de rendimiento
7. Capacidad de depuración
8. Marco de trabajo de servicios Web XML
9. Entorno de host extensible y administración del ciclo de vida de las aplicaciones
10. Entorno de diseñador extensible

Administración de sitios Web

La configuración de sitios Web se mejora para incluir muchas más opciones. Puede administrar con facilidad las opciones de configuración de la aplicación utilizando la herramienta de administración de sitios Web, que proporciona una interfaz similar a un asistente para la configuración y el mantenimiento de sus aplicaciones. La herramienta de administración de sitios Web es particularmente útil para administrar sitios remotos (por ejemplo, sitios alojados por un ISP compatible con ASP.NET).

Si aloja sitios para otros, puede utilizar un nuevo complemento de ASP.NET Microsoft Management Console (MMC) o una API administrativa para administrar los sitios y controlar su estado. Puede dar más valor a su sitio de alojamiento proporcionando controles o servicios que puede habilitar o deshabilitar selectivamente.

Nuevas características y herramientas de administración

ASP.NET. Una nueva API de configuración hace posible controlar la configuración mediante programación. Las nuevas herramientas proporcionan una interfaz GUI para configurar aplicaciones; la nueva herramienta de administración de sitios Web la administración de sus propios sitios (local y remotamente) mediante una interfaz basada en Web, y un complemento de MMC específico de ASP.NET permite tratar escenarios de configuración complejos mediante una herramienta estándar de Windows basada en servidor.

Carpetas reservadas para una funcionalidad especial

Los sitios Web pueden incluir una carpeta App_Code en la que los desarrolladores del sitio puedan poner código fuente que se compila automáticamente como parte del sitio Web, haciendo que sea innecesario compilar componentes o controles antes de usarlos en un sitio. La carpeta App_Data está reservada para bases de datos (por ejemplo, archivos .mdf de SQL Server Express Edition). Las carpetas especiales para recursos contienen archivos basados en XML que incluyen cadenas y otros recursos de localización que se compilan dinámicamente en ensamblados en tiempo de ejecución.

Pre compilar sitios Web para la comprobación de errores y la implementación

Ahora puede pre compilar su sitio Web, lo que permite mejorar el rendimiento evitando la sobrecarga de la compilación dinámica y hace posible detectar errores en la fase de compilación. También puede pre compilar un sitio para su implementación, generando una versión del sitio que puede copiar o instalar en un servidor de producción con facilidad. Al pre compilar para la implementación se quita código fuente, lo que ayuda a proteger su propiedad intelectual. Para obtener información detallada.

Desplazamiento

Puede agregar el desplazamiento de sitio a sus sitios Web definiendo un mapa del sitio (normalmente un archivo XML). A continuación, puede utilizar los nuevos controles de desplazamiento, por ejemplo los controles TreeView y SiteMapPath que pueden crear automáticamente un menú o una vista de árbol de páginas, o que pueden mostrar una ruta de desplazamiento que muestra la jerarquía de páginas actual.

Datos

ASP.NET incluye una compatibilidad para trabajar con datos en sus aplicaciones.

Controles de origen de datos

Para enlazar datos a controles en páginas Web, puede utilizar controles de origen de datos, que encapsulan conexiones, comandos de consulta y parámetros en un único control. Incluye controles de origen de datos que funcionan con una serie de orígenes de datos de servidor, incluidos Microsoft SQL Server, Microsoft Access, archivos XML, servicios Web, mapas de sitio de FrontPage y objetos comerciales que devuelven conjuntos de datos. Todos los controles de origen de datos admiten el mismo modelo de objetos básico, ofreciéndole una manera coherente de trabajar con datos sin tener en cuenta su origen.

Los controles de origen de datos pueden obtener datos automáticamente y administrarlos cuando se ejecuta la página. Ya no necesita escribir código para ejecutar los comandos y administrar los conjuntos de datos para los escenarios de datos comunes. Sin embargo, si su aplicación lo requiere, todavía tiene acceso a las funciones de datos de nivel más bajo expuestas por ADO.NET.

Para pasar parámetros a los controles de origen de datos, puede configurar los controles para que obtengan valores de parámetros de otros controles, del estado de sesión, de cadenas de consulta o cookies; además, puede establecer los valores de parámetro mediante programación.

Acceso a datos de nivel medio

Utilizando el nuevo control ObjectDataSource, puede agregar el acceso a datos con facilidad a una página que se basa en un objeto comercial de nivel medio. El control ObjectDataSource presenta la misma interfaz de enlace para los controles en la página, pero en lugar de realizar un acceso directo a las bases de datos, invoca métodos en un componente especificado.

Controles de visualización de datos

Incluye una compatibilidad para mostrar y actualizar los datos con controles en páginas Web. Todos los controles pueden utilizar controles de origen de datos como el origen de datos en lugar de trabajar directamente con un conjunto de datos u otro almacén. También puede aprovecharse de los nuevos controles siguientes generados específicamente para facilitar el acceso a datos:

- Los controles GridView, DetailsView, FormView para mostrar y editar datos. (El control GridView reemplaza al control DataGrid de las versiones anteriores).
- El control TreeView para mostrar la información jerárquica de archivos XML, archivos de mapa del sitio y orígenes de datos relacionales.
- Los controles SiteMapPath y Menú para proporcionar compatibilidad enlazada a datos para el desplazamiento.

Compatibilidad con XML

Puede utilizar los datos XML de diversas maneras en ASP.NET. Un control de origen de datos XML expone datos XML que se van a utilizar como datos jerárquicos o tabulares. Puede enlazar un control TreeView a los datos XML para proporcionar una vista jerárquica para los usuarios o puede enlazar un control de lista como el control GridView para mostrar XML de una manera tradicional.

Almacenamiento de cadenas de conexión

Para mejorar la seguridad del sitio Web, puede almacenar las cadenas de conexión en una sección dedicada del archivo de configuración.

Requisitos
El entorno .NET Framework

ASP.NET forma parte de .NET Framework como el espacio de nombres System.Web. Para utilizar ASP.NET, debe tener .NET Framework instalado en el equipo que aloja los sitios Web de ASP.NET.

Entornos de creación de código

Puede crear clases y páginas ASP.NET utilizando cualquier editor de texto, como el Bloc de notas de Microsoft, que se suministra con Microsoft Windows. Sin embargo, Microsoft Visual Studio y otros entornos de desarrollo integrados (IDE) proporcionan muchas características de programación, como plantillas de página, IntelliSense, llenado automático de código y compilación en tiempo de diseño. Estas características pueden acelerar el proceso de desarrollo y proporcionar la organización para su proyecto.

Servidores Web

Para trabajar con una aplicación Web ASP.NET, debe utilizar un explorador para realizar solicitudes al servidor Web que aloja la aplicación. Las aplicaciones Web ASP.NET se alojan normalmente utilizando IIS como servidor Web. Puede probar las aplicaciones ejecutando IIS localmente en el equipo o implementar páginas y componentes en un servidor compartido.

Bases de datos

Si la aplicación implica almacenamiento de datos, necesita acceso a una aplicación de base de datos como Microsoft SQL Server, y a los permisos adecuados para leer y escribir datos en la base de datos. En los escenarios típicos, una cuenta se utiliza para tener acceso a la base de datos en tiempo de diseño y se utiliza una cuenta diferente para tener acceso a la base de datos en tiempo de ejecución. De manera predeterminada, las aplicaciones Web ASP.NET se ejecutan en el contexto de una cuenta de equipo local denominada ASPNET (para Windows 2000 y Windows XP) o en el contexto de la cuenta NETWORK SERVICE (para Windows Server 2003) Además, algunas características de ASP.NET como las propiedades de perfil y suscripción requieren una base de datos.

Ciclo de vida en ASP.NET

En ASP.NET, deben producirse varios pasos de procesamiento para que una aplicación ASP.NET se inicialice y procese las solicitudes. Además, ASP.NET es sólo una parte de la arquitectura de servidor Web que atiende las solicitudes realizadas por los exploradores. Es importante que comprenda el ciclo de vida de la aplicación para que pueda escribir código en la fase apropiada del ciclo y conseguir el efecto deseado.

Las fases del ciclo de vida de la aplicación ASP.NET.

1) *El usuario solicita un recurso de aplicación del servidor Web.*

El ciclo de vida de una aplicación ASP.NET se inicia con una solicitud enviada por un explorador al servidor Web (para las aplicaciones ASP.NET, normalmente es IIS). ASP.NET es una extensión ISAPI bajo el servidor Web. Cuando un servidor Web recibe una solicitud, examina la extensión de nombre de archivo del archivo solicitado, determina la extensión ISAPI que debería procesar dicha solicitud y, a continuación, pasa ésta a la extensión ISAPI apropiada. ASP.NET procesa las extensiones de nombre de archivo que tiene asignadas, como .aspx, .ascx, .ashx y .asmx.

Si una extensión de nombre de archivo no se ha asignado, éste no recibirá la solicitud. Es importante entender esto en las aplicaciones que utilizan la autenticación de ASP.NET. Por ejemplo, dado que los archivos .htm normalmente no se asignan a ASP.NET, esta aplicación no realizará la autenticación ni las comprobaciones de autorización en las solicitudes de archivos .htm. Por consiguiente, aunque un archivo incluya únicamente contenido estático, si desea que ASP.NET compruebe la autenticación, cree el archivo utilizando una extensión de nombre de archivo asignada a ASP.NET, por ejemplo .aspx.

Si opta por crear un controlador personalizado para procesar una extensión de nombre de archivo determinada, deberá asignar la extensión a ASP.NET en IIS, así como también registrar el controlador en el archivo Web.config de la aplicación.

2) ASP.NET recibe la primera solicitud para la aplicación

Cuando ASP.NET recibe la primera solicitud para cualquier recurso de una aplicación, una clase denominada ApplicationManager crea un dominio de aplicación. Los dominios de aplicación proporcionan aislamiento entre aplicaciones para las variables globales y permiten descargar cada aplicación de forma independiente. Dentro de un dominio de aplicación, se crea una instancia de la clase denominada HostingEnvironment, que proporciona acceso a la información sobre la aplicación, como el nombre de la carpeta en la que está almacenada la aplicación.

En el diagrama siguiente se muestra esta relación:

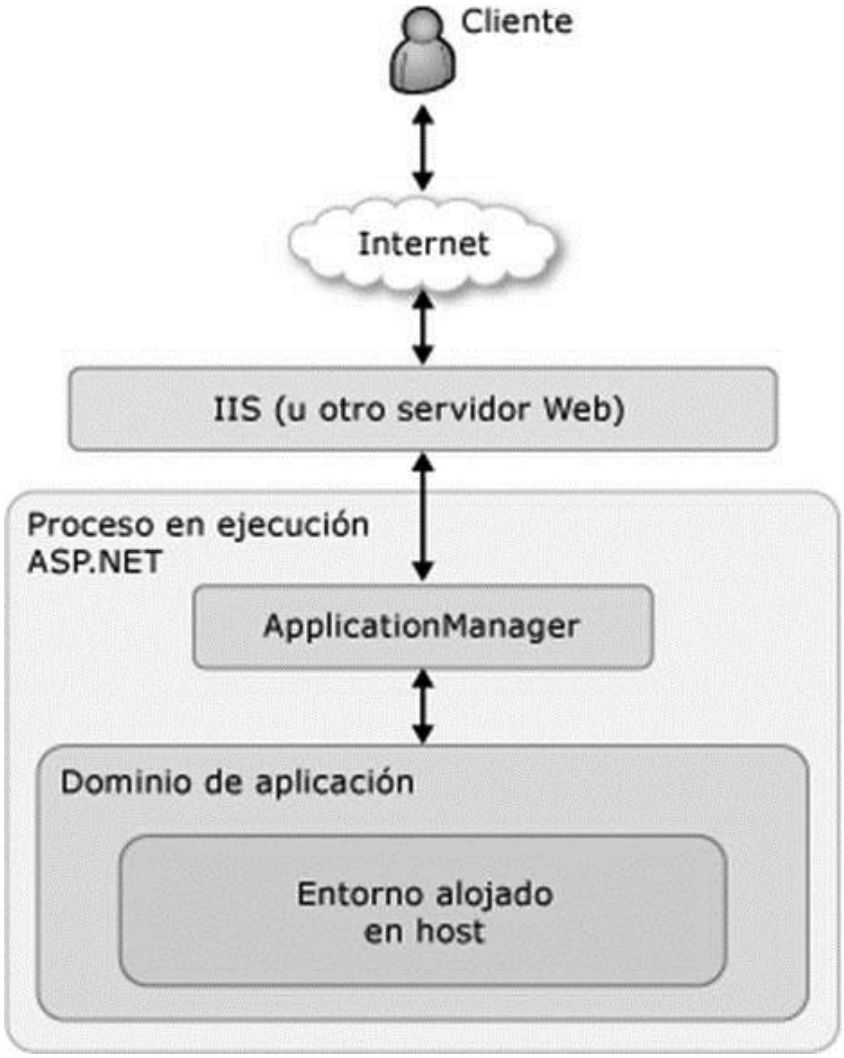


Figura 2.1.2 Solicitud de aplicación

ASP.NET también compila los elementos de nivel superior de la aplicación si es necesario, incluido el código de aplicación de la carpeta App_Code.

3) Se crean los objetos de núcleo ASP.NET para cada solicitud

Una vez creados el dominio de aplicación y una instancia del objeto HostingEnvironment, ASP.NET crea e inicializa objetos de núcleo, como HttpContext, HttpRequest y HttpResponse.

La clase HttpContext contiene objetos específicos de la solicitud de aplicación actual, como los objetos HttpRequest y HttpResponse. El objeto HttpRequest contiene datos sobre la solicitud actual, entre los que se incluyen las cookies e información del explorador. El objeto HttpResponse contiene la respuesta que se envía al cliente, la cual incluye todos los resultados presentados y las cookies.

4) Se asigna un objeto HttpApplication a la solicitud

Una vez que se han inicializado todos los objetos principales de la aplicación, ésta se inicia creando una instancia de la clase HttpApplication. Si la aplicación tiene un archivo Global.asax, ASP.NET crea una instancia de la clase Global.asax derivada de la clase HttpApplication y la utiliza para representar la aplicación.

La primera vez que se solicita una página o un proceso ASP.NET en una aplicación, se crea una nueva instancia de HttpApplication. Sin embargo, para maximizar el rendimiento, las instancias de HttpApplication se podrían reutilizar para múltiples solicitudes.

Cuando se crea una instancia de HttpApplication, también se crean los módulos configurados. Por ejemplo, si la aplicación está configurada para ello, ASP.NET crea un módulo SessionStateModule. Una vez creados todos los módulos configurados, se llama al método Init de la clase HttpApplication.

En el diagrama siguiente se muestra esta relación:

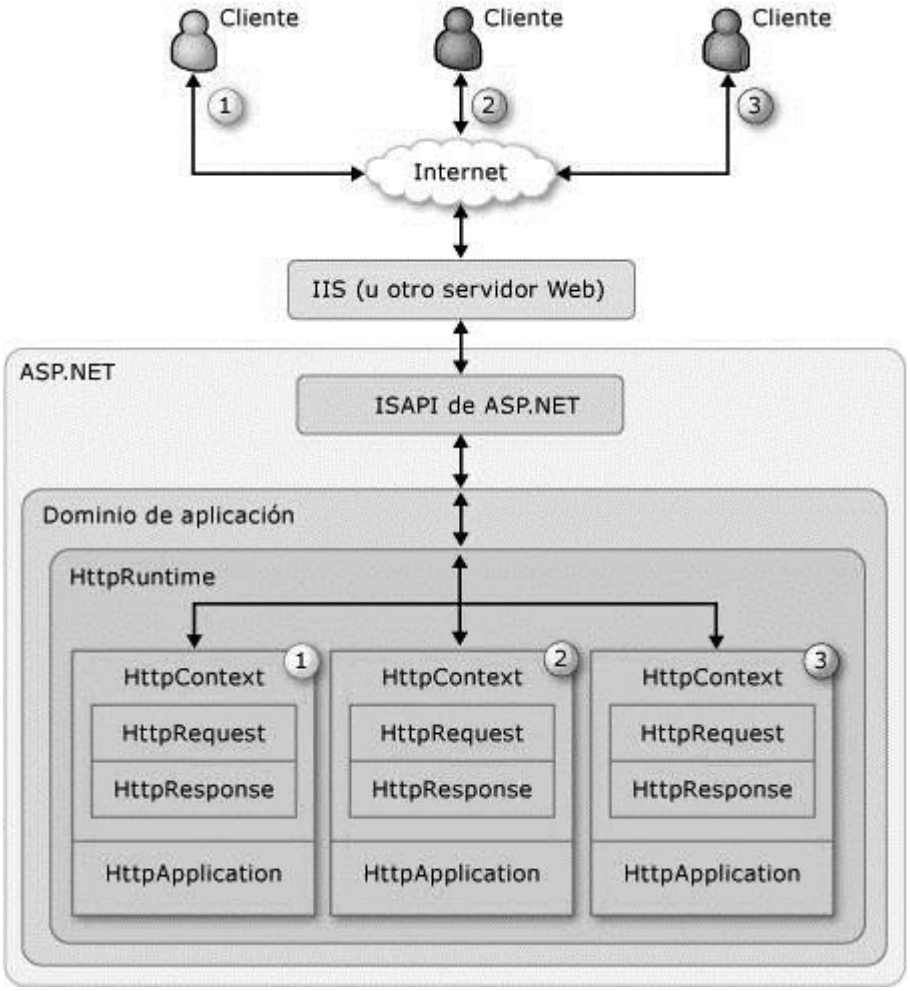


Figura 2.1.3 Asignación de http

*5) La canalización de **HttpApplication** procesa la solicitud.*

La clase **HttpApplication** ejecuta los eventos siguientes mientras se procesa la solicitud. Dichos eventos son de particular interés para los desarrolladores que desean extender la clase **HttpApplication**.

1. Valida la solicitud, que examina la información enviada por el explorador y determina si contiene formato potencialmente malintencionado.
2. Realiza la asignación de direcciones URL si se ha configurado alguna dirección URL en la sección UrlMappingsSection del archivo Web.config.
3. Produce el evento BeginRequest.
4. Produce el evento AuthenticateRequest.
5. Produce el evento PostAuthenticateRequest.
6. Produce el evento AuthorizeRequest.
7. Produce el evento PostAuthorizeRequest.
8. Produce el evento ResolveRequestCache.
9. Produce el evento PostResolveRequestCache.
10. Basándose en la extensión de nombre de archivo del recurso solicitado (asignada en el archivo de configuración de la aplicación), selecciona una clase que implemente IHttpHandler para procesar la solicitud. Si la solicitud es para un objeto (página) derivado de la clase Page y es necesario compilar la página, ASP.NET la compila antes de crear una instancia de ella.
11. Produce el evento PostMapRequestHandler.
12. Produce el evento AcquireRequestState.
13. Produce el evento PostAcquireRequestState.
14. Produce el evento PreRequestHandlerExecute.
15. Llama al método ProcessRequest (o a la versión asincrónica BeginProcessRequest) de la clase IHttpHandler apropiada para la solicitud. Por ejemplo, si la solicitud es para una página, la controla la instancia de la página actual.
16. Produce el evento PostRequestHandlerExecute.
17. Produce el evento ReleaseRequestState.
18. Produce el evento PostReleaseRequestState.
19. Realiza el filtrado de respuestas si se define la propiedad Filter.
20. Produce el evento UpdateRequestCache.
21. Produce el evento PostUpdateRequestCache.
22. Produce el evento EndRequest

Visual J #

Microsoft Visual J # 2005 permite a los desarrolladores utilizar la sintaxis de Java para crear aplicaciones y servicios en .NET Framework. Visual J # integra la sintaxis de Java en el Visual Studio entorno de desarrollo integrado (IDE).

J # 64-bit Información de soporte en tiempo de ejecución

Para satisfacer la demanda de soporte de ejecución de 64 bits, Microsoft ha publicado el Visual J # 2.0. Esto permite a los desarrolladores para compilar su Visual J # código para ejecutar de forma nativa en las versiones de 64 bits de Windows y .NET Framework, incluyendo .NET Framework 2.0, 3.0 y 3.5.Lanzada en 2007, con el apoyo continuo a través de 2017 (5 años dominantes y 5 años de soporte extendido) en lugares en-US.

Retiro de J # lenguaje y conversión del lenguaje Java de las futuras versiones de Visual Studio

Puesto que los clientes nos han dicho que la función J # existente cumple en gran medida de sus necesidades y el uso de J # está disminuyendo, Microsoft se está retirando el producto de Visual J # y Java Lenguaje Conversión Assistant para asignar mejor los recursos para otras necesidades del cliente. El J # lenguaje y herramienta JLCA no está disponible en versiones de Visual Studio después de Visual Studio 2005. Para preservar las inversiones de los clientes en J # existente, Microsoft seguirá apoyando el J # y tecnología JLCA suministrada con Visual Studio 2005 hasta 2015 de acuerdo con nuestro producto estrategia de ciclo de vida.

JavaScript

JavaScript es un lenguaje de programación creado con el objetivo de integrarse a HTML y facilitar la programación de páginas interactivas.

No se debe confundir Java con JavaScript. Java es un lenguaje completo que permite crear aplicaciones independientes, mientras que JavaScript es un lenguaje que funciona como extensión de HTML. Es un lenguaje orientado a objetos, diseñado para el desarrollo de aplicaciones cliente-servidor a través de internet

El código de programación de JavaScript, llamado script, se introduce se introduce directamente en el documento HTML, y no necesita ser compilado es el propio navegador el que se encarga de traducir dicho código.

Gracias a JavaScript podemos desarrollar programas que se ejecuten directamente en el navegador de tal manera que este pueda efectuar determinadas operaciones o tomar decisiones sin necesidad de acceder al servidor.

JavaScript (abreviado comúnmente JS) es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, y dinámico.

Existe una forma de JavaScript del lado del servidor (Server-side JavaScript o SSJS). Su uso en aplicaciones externas a la web, por ejemplo en documentos PDF, aplicaciones de escritorio (mayoritariamente widgets) es también significativo.

JavaScript se diseñó con una sintaxis similar al C, aunque adopta nombres y convenciones del lenguaje de programación Java

Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del Document Object Model (DOM).

Actualmente es ampliamente utilizado para enviar y recibir información del servidor junto con ayuda de otras tecnologías como AJAX. JavaScript se interpreta en el agente de usuario al mismo tiempo que las sentencias van descargándose junto con el código HTML.

Desde el lanzamiento en junio de 1997 del estándar ECMAScript 1, han existido las versiones 2, 3 y 5, que es la más usada actualmente (la 4 se abandonó). En junio de 2015 se cerró y publicó la versión ECMAScript 6.

Características

Imperativo y estructurado

JavaScript es compatible con gran parte de la estructura de programación de C (por ejemplo, sentencias if, bucles for, sentencias switch, etc.).. Como en C, JavaScript hace distinción entre expresiones y sentencias. Una diferencia sintáctica con respecto a C es la inserción automática de punto y coma, es decir, en JavaScript los puntos y coma que finalizan una sentencia pueden ser omitidos.

Dinámico

Tipado dinámico

Como en la mayoría de lenguajes de scripting, el tipo está asociado al valor, no a la variable. Por ejemplo, una variable x en un momento dado puede estar ligada a un número y más adelante, religada a una cadena. JavaScript es compatible con varias formas de comprobar el tipo de un objeto, incluyendo duck typing. Una forma de saberlo es por medio de la palabra clave typeof.

Objetual

JavaScript está formado casi en su totalidad por objetos. Los objetos en JavaScript son arrays asociativos, mejorados con la inclusión de prototipos (ver más adelante). Los nombres de las propiedades de los objetos son claves de tipo cadena: obj.x = 10 y obj['x'] = 10 son equivalentes, siendo la notación con punto azúcar sintáctico. Las propiedades y sus valores pueden ser creados, cambiados o eliminados en tiempo de ejecución. La mayoría de propiedades de un objeto (y aquellas que son incluidas por la cadena de la herencia prototípica) pueden ser enumeradas a por medio de la instrucción de bucle for... in. JavaScript tiene un pequeño número de objetos predefinidos como son Function y Date.

Evaluación en tiempo de ejecución

JavaScript incluye la función eval que permite evaluar expresiones como expresadas como cadenas en tiempo de ejecución. Por ello se recomienda que eval sea utilizado con precaución y que se opte por utilizar la función JSON.parse () en la medida de lo posible, pues resulta mucho más segura.

Funcional

Funciones de primera clase

A las funciones se les suele llamar ciudadanos de primera clase; son objetos en sí mismos. Como tal, poseen propiedades y métodos, como .call () y .bind (). Una función anidada es una función definida dentro de otra. Esta es creada cada vez que la función externa es invocada. Además, cada función creada forma una clausura; es el resultado de evaluar un ámbito conteniendo en una o más variables dependientes de otro ámbito externo, incluyendo constantes, variables locales y argumentos de la función externa llamante. El resultado de la evaluación de dicha clausura forma parte del estado interno de cada objeto función, incluso después de que la función exterior concluya su evaluación.

Prototípico

Prototipos

JavaScript usa prototipos en vez de clases para el uso de herencia. Es posible llegar a emular muchas de las características que proporcionan las clases en lenguajes orientados a objetos tradicionales por medio de prototipos en JavaScript.

Funciones como constructores de objetos

Las funciones también se comportan como constructores. Prefijar una llamada a la función con la palabra clave new crear una nueva instancia de un prototipo, que heredan propiedades y métodos del constructor (incluidas las propiedades del prototipo de Object). ECMAScript 5 ofrece el método Object.create, permitiendo la creación explícita de una instancia sin tener que heredar automáticamente del prototipo de Object (en

entornos antiguos puede aparecer el prototipo del objeto creado como null). La propiedad prototype del constructor determina el objeto usado para el prototipo interno de los nuevos objetos creados. Se pueden añadir nuevos métodos modificando el prototipo del objeto usado como constructor. Constructores predefinidos en JavaScript, como Array u Object, también tienen prototipos que pueden ser modificados. Aunque esto sea posible se considera una mala práctica modificar el prototipo de Object ya que la mayoría de los objetos en JavaScript heredan los métodos y propiedades del objeto prototype, objetos los cuales pueden esperar que estos no hayan sido modificados.

Otras características

Entorno de ejecución

JavaScript normalmente depende del entorno en el que se ejecute (por ejemplo, en un navegador web) para ofrecer objetos y métodos por los que los scripts pueden interactuar con el "mundo exterior". De hecho, depende del entorno para ser capaz de proporcionar la capacidad de incluir o importar scripts (por ejemplo, en HTML por medio del tag <script>). (Esto no es una característica del lenguaje, pero es común en la mayoría de las implementaciones de JavaScript.)

Funciones variádicas

Un número indefinido de parámetros pueden ser pasados a la función. La función puede acceder a ellos a través de los parámetros o también a través del objeto local arguments. Las funciones variádicas también pueden ser creadas usando el método .apply().

Funciones como métodos

A diferencia de muchos lenguajes orientados a objetos, no hay distinción entre la definición de función y la definición de método. Más bien, la distinción se produce durante la llamada a la función; una función puede ser llamada como un método. Cuando una función es llamada como un método de un objeto, la palabra clave this, que es una variable local a la función, representa al objeto que invocó dicha función.

Arrays y la definición literal de objetos

Al igual que muchos lenguajes de script, arrays y objetos (arrays asociativos en otros idiomas) pueden ser creados con una sintaxis abreviada. De hecho, estos literales forman la base del formato de datos JSON.

Expresiones regulares

JavaScript también es compatible con expresiones regulares de una manera similar a Perl, que proporcionan una sintaxis concisa y poderosa para la manipulación de texto que es más sofisticado que las funciones incorporadas a los objetos de tipo string.

Estructura de JavaScript

Dentro de un documento de HTML para definir el inicio de un programa o código se debe utilizar la etiqueta

<SCRIPT LANGUAGE=”JavaScript”>

Y terminar con

</SCRIPT>

Los códigos de JavaScript se pueden colocar en la cabecera o en el cuerpo del documento. Es muy importante recordar que JavaScript es un lenguaje case sensitive.

Comentarios

Los comentarios se pueden introducir de dos formas diferentes

- Los comentarios en una sola línea irán precedidos por //
- Los comentarios de varias líneas irán encerrados en /* y */

Corchetes

Los corchetes se utilizan para definir fragmentos de código de manera que estos no se junten con el resto del código.

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE='JavaScript'>
// Definición de una Función
For  (cuenta=1 ; cuenta < 4 ; cuenta ++)
// Inicio de código
{
        alert ("hola mundo")
}
//fin de código de la función
</SCRIPT>
</HEAD>
</HTML>
```

El punto y coma

La misión del punto y coma en JavaScript es la de separar sentencias que se encuentran en una misma línea. También puede indicar el final de una sentencia que ocupe varias líneas.

Las variables

JavaScript admite prácticamente cualquier tipo de nombre para definir una variable. Sin embargo hay una serie de consideraciones que deben tomarse en cuenta:

- El primer caracter debe ser siempre una letra o el guion de subrayado "_". Los restantes caracteres pueden ser letras o números o guiones teniendo como precaución no dejar espacios entre ellos.

- El nombre de las variables no debe coincidir con las palabras reservadas de JavaScript.

- JavaScript diferencia entre mayúsculas y minúsculas.

Para declarar variables se utiliza la palabra clave var seguida del nombre dela variable

Tipos de variables

- Variables de cadena
- Variables numéricas
- Variables Booleanas
- Variables de objetos

Variables de cadena

Es aquella que contiene texto. Se delimitan entre doble comilla o sencilla y pueden contener cualquier tipo de carácter. Para los caracteres especiales se tiene lo siguiente:

- **\b** carácter anterior
- **\f** salto de pagina
- **\n** salto de línea
- **\r** retorno de carro
- **\t** tabulador
- \\ carácter \
- \’ comilla doble
- \“ comilla simple

Variables numéricas

Son las que contienen números enteros o de punto flotante.

Enteros

Se pueden representar con tres bases distintas

- La base diez (10)
- La base hexadecimal (16)
- La octal (8)

En base diez se definen las variables normalmente

Numero = 100

En base hexadecimal se agrega el prefijo 0x y los números contienen los dígitos 0 al 9 y las letras de la A a la F

Numero= 0xff

En base octal se agrega el prefijo 0 y los números contienen los dígitos 0 al 7

Numero= 0123

Flotante

Se define como un numero entero seguido de un punto y una fracción decimal. El exponente se indica mediante una E seguida de un entero positivo o negativo.

Numero= 10.134
Numero= 10.12E+2

Variables Booleanas

Se expresan mediante valores de verdadero o falso (true, false)

Contento=false;

Variables Objeto

Se refiere a las predefinidas o las programadas. Se crean con la palabra **new** objeto. Ver en definición de objetos de JavaScript.

Obj1 = new obj

JavaScript no necesita de la declaración del tipo de variable, ya que en función de las operaciones que debe realizar el tipo ira cambiando automáticamente

Var var1 = “50” // valor de cadena
Var var2 = 10 // valor entero
Resultado_1= var1 + var2
Resultado_2= var2+ var1

Resultado_1= “5010”
Resultado_2= 60

Palabras Reservadas

Palabras Reservadas			
abstract	Boolean	break	byte
case	Catch	char	class
const	Continue	default	do
double	Else	extends	false
final	Finally	float	for
function	Goto	int	implement
input	In	Instance of	interface
long	Native	new	null
package	Private	protected	public
return	Short	static	super
switch	Synchronized	this	throw
throws	Transient	true	try
var	Val	while	with

Figura 2.1.4 Palabras reservadas JavaScript

Operadores

Los operadores son la forma en que realizamos acciones sobre las variables y los valores.
Se subdividen en los siguientes grupos

- Aritméticos
- Lógicos
- Comparación
- Condicionales
- Bit a bit
- Asignación

Estos son los operadores más comunes en JavaScript:

Tipos	Operador	Ejemplo	Descripción
Aritméticos	+	V1 + V2	Suma
	-	V1 – V2	Resta
	*	V1 * V2	Producto
	/	V1 / V2	División
	%	V1 % V2	Resto
	++	++V1	Pre incremento
	++	V1++	Pos incremento
	--	--V1	Pre decremento
	--	V1--	Pos decremento
	-	-V1	Negación
Comparación	==	V1 == V2	Igualdad
	!=	V1 != V2	Distinto
	<	V1 < V2	Menor que
	<=	V1 <= V2	Menor o igual
	>	V1 > V2	Mayor que
	>=	V1 >= V2	Mayor o igual
Lógicos	&&	V1 && V2	Y
	||	V1 || V2	O
	!	!V1	NOT
Condicionales	( )?:	(Condición)? V1:V2	Si condición v1 si no v2
Asignación	x=	V1 x= V2	V1 = V1 + V2
	-=	V1 -= V2	V1 = V1 - V2
	*=	V1 *= V2	V1 = V1 * V2
	/=	V1 /= V2	V1 = V1 / V2
	%=	V1 %= V2	V1 = V1 % V2
	=	V1 = V2	V1 = V2
Tipo	typeof	typeof V1	Entero, numérico…
Propiedad	.	Obj.a Obj[a]	Objeto Obj con propiedad a

Figura 2.1.5 Operadores JavaScript

Estructuras condicionales

Para controlar el desarrollo de un programa JavaScript cuenta con los siguientes comandos

- if
- for
- while
- do while
- break
- Switch ()

La tabla siguiente muestra el uso de estas estructuras

Estructura	Operador	Ejemplo	Descripción

if	if (condición)
	{
	código
	}
		If (V1=V2)
		{
		document.write (“hola”)
		}
			Si v1 es igual a v2
			escribe hola

If .. else	if (condición)
	{
	código
	}
	else
	{
	código
	}
		If (V1=V2)
		{
		document.write (“hola”)
		}
		Else
		{
		document.write (“adiós”)
		}
			Si v1 es igual a v2
			escribe hola si no escribe
			adiós

for	for (valor inicial;
	condición;
	actualización)
	{
	código
	}
		for (cont; cont< 3;
		cont++)
		{
		document.write (“Num ”
		+ cont + “
		”)
		}
			Num 0
			Num 1
			Num 2

for .. in	for (variable in objeto)
	{
	código
	}
		for (a.name; next a)
		{
		V1==a.name;
		document.write (“Name ”
		+ V1 + “
		”)
		}
			Name uno
			Name dos
			Name tres

while	while
	(condición)
	{
	código
	}
		while (cont< 3)
		{
		cont++)
		document.write (“Num ”
		+ cont + “
		”)
		}
		alert(“fin de while”)
			Num 0
			Num 1
			Num 2
			fin de while

do while	do { código	do { password = promt	Fin de while cuando se escribe “bueno”

	} While (condición)	(“introduzca el password”,”psw”); } While (password! =”bueno”) alert(“fin de while”)	
break	while (condición) { código break }	while (nom!=”fin”) { nom=promt(“introduce un nombre ”) if (nom==”fin”) { alert(“Fin de la secuencia”) break; } }	Pide nombres mientras no se introduzca la palabra “fin”
Switch () Case default	switch (var) { case value1: código break; case value2: código break; default: código }	switch (a) { case 10: alert(“var “ + a) break; case 20: alert(“var “ + a) break; default: alert(“var “ + a) }	a= 20 var 20

Figura 2.1.6 Ciclos JavaScript

Objetos, funciones y métodos

Los objetos

Un objeto es una agrupación de variables denominadas propiedades que realizan operaciones con las variables propias. En JavaScript podemos crear nuestros propios objetos o utilizar los ya implementados.

Todos los objetos tienen una serie de propiedades asignadas. Estas propiedades se acceden

mediante el nombre del objeto seguido del “**.**”

Objeto.propiedad

La s funciones

Una función es un conjunto de sentencias que realizan alguna tarea específica.
Una función se define con:

- Nombre

- Argumentos
- Las sentencias encerradas entre {}

Las funciones pueden ser definidas con otras funciones definidas anteriormente.
La sintaxis de una función es:

```
function nombre (parametro1, parametro2…..)
{
instrucciones
}
```

Los parámetros pueden ser variable, número u objetos.
Para que una función devuelva el resultado de sus operaciones se utiliza la palabra:

return

Los métodos

Un método es una función asociada a un objeto y particular del tipo que las define. Un método es una acción que ejecutamos sobre los datos de un objeto.

Los métodos se definen en el mismo lugar de las funciones asociándolos posteriormente a un objeto
La sintaxis para definir un método es la siguiente:

Objeto.nombremetodo=nombre_funcion

Para hacer la llamada al método se usa;

Objeto.nombremetodo (parametro1, parametro2)

Creación de objetos

Para definir un objeto se debe:

1. Definir el objeto a través de una función
2. Crear una instancia mediante la palabra new

La sintaxis

```
function nombredeObjeto (propiedad1, propiedad2…..)
{
this.propiedad1=propiedad1
this.propiedad2=propiedad2
.
.
.
}
```

Alternativa de definición

```
var obj = {
   a: "hello world",
```

```
    b: 42
};
```

Ejemplo

```
var b = "a";

obj[b];        // "hello world"
obj["b"];      // 42
```

Objetos y funciones predefinidas

JavaScript dispone de una serie de objetos y funciones predefinidos. Estos son los más comunes:

Objeto	**Método**	**Descripción**
String()	anchor(name)	Crea un enlace
	big()	Con carácter grande
	blink()	Se muestra intermitente
	bold()	Con negritas
	charAt(n)	Carácter n
	fixed()	Con fuente no proporcional
	fontcolor(color)	Con el color color
	fontsize(n)	Con tamaño n (1..7)
	indexof(smallstring,start)	Posición de un fragmento
	italics()	En cursiva
	lastindexesOf()	Ultima posición de un carácter
	link(url)	Se convierte en url
	small()	En fuente pequeña
	strike()	Subrayado
	sub()	En subíndice
	substring(x,y)	Muestra un fragmento de x a y
	sup()	En superíndice
	toLowerCase()	En minúsculas
	toUpperCase()	En mayúsculas
	lenght()	Tamaño
	prototype()	Propiedades
Math()	propiedades	
	E	Constante de Euler
	LN2	Logaritmo natural base 2
	LN10	Logaritmo natural base 10
	LOG2E	Logaritmo de e en base 2
	LOG10E	Logaritmo de e en base 10
	PI	Numero pi
	SQRT1_2	Raíz cuadrada de 0.5
	SQRT2	Raíz cuadrada de 2
	Metodos	
	abs(n)	Absoluto de n
	acos(n)	Arcocoseno de n
	asin(n)	Arcoseno de n
	atan(n)	Arco tangente de n
	ceil(n)	Redondea hacía el superior n
	cos(n)	Coseno de n

	exp(n)	Exponencial de n
	floor(n)	Redondea hacia el inferior n
	log(n)	Logaritmo n
	max(x,y)	Mayor entre x y y
	min(x,y)	Menor entre x y y
	pow(x,y)	Potencia de dos números
	random(n)	Número aleatorio
	round(n)	Redondea al número más cercano n
	sin(n)	Seno de n
	sqrt(n)	Raíz cuadrada de n
	tan(n)	Tangente de n
Date()	Meses de 0 a 12	
	getDate()	Dia del mes actual
	getDay()	Dia de la semana actual
	getHour()	La hora
	getMinutes()	Los minutos
	getSeconds()	Los segundos
	getTime()	La hora
	getTimeZoneoffset()	Diferencia entre la hora actual y la GTM
	getYear()	Año actual
	parse(datestring)	Tiempo entre 00000 del 1 de enero 1970 y datestring
	setDate(valor)	Establece el dia del mes
	setHour(valor)	Establece la hora
	setMinutes(valor)	Establece los minutos
	setMounth(valor)	Establece el mes
	setSeconds(valor)	Establece los segundos
	setTime(valor)	Establece el tiempo
	setYear(valor)	Establece el año
	setGMTString()	Devuelve en formato GMT
	toLocateString()	Devuelve en formato string
	UTC()	Devuelve en milisegundos
Array()	length	Número de elementos
	join()	Convierte en string separado por comas
	reverse	Invierte el orden de los elementos
	sort()	Devuelve ordenados los elementos
Boolean()		Convierte datos no booleanos en boolanos
Eval()		Funcion que evalua expresiones
parseInt()	parseInt(cadena,base)	Convierte cadena en número de base
parseFloat()		Convierte a número real
escape()		Convierte a Código ASCII
unescape()		Regresa carácter de modo ASCII

Figura 2.1.7 Objetos, metodos y funciones JavaScript

Objetos del Navegador

Cuando se carga una página se crean objetos característicos de acuerdo al contenido de la misma.

Los objetos y propiedades más comunes son:

Objeto	Descripción
window	De más alto nivel, contiene las propiedades de la ventana, en el supuesto que se trabaje con frames. Existe un objeto window para cada frame
location	Contiene las propiedades de la URL
history	Contiene las propiedades de la URL que el usuario ha visitado. Cache
document	Propiedades del documento actual. Color, imágenes, enlaces, etc.

Figura 2.1.8 Objetos html en JavaScript

Cada documento tiene una jerarquía empezando por el objeto window o el document.
Jerarquía

window
- history
- location
- parent, frames, self, top
- document
  - links
  - anchor
  - form
    - elements

El objeto window posee una serie de propiedades y complementos básicos

Objeto	Método, Propiedad	Descripción
window	closed	Determina si está cerrada
	defaultStatus	Mensaje de estado
	frames	Matriz de los frames
	length	Numero de frames
	name	Nombre de la ventana
	outerHeight	Altura de la ventana
	outerwidth	Ancho de la ventana
	parent	Refiere a la ventana con <frameset>
	self	Ventana activa
	top	Referencia a la ventana superior
	status	Mensaje de la barra estatus del navegador
	window	Referencia a la ventana activa
	Métodos	
	open()	Abre una ventana
	back()	Retrocede a la pagina anterior
	blur()	Quita el foco de la ventana
	captureEvents()	Captura todos los eventos de un tipo
	clearInterval()	Cancela el tiempo de espera
	close()	Cierra una ventana

	alert()	Muestra una ventana con mensaje y botón aceptar
	confirm()	Muestra ventana con mensaje y dos botones de cancelar y aceptar
	find()	Abre ventana de búsqueda
	prompt()	Abre ventana de dialogo con campo de entrada
	setTime()	Retrasa la ejecución de una instrucción
	ClearTimeout()	Anula el timeout
	Componentes	
	toolbar	Barra de herramientas
	location	Barra de dirección
	directories	Botones de directorio
	status	Muestra barra de estado
	menubar	Barra de menú
	scrollbar	Barras de desplazamiento
	resizable	Permite ajustar tamaño
	width	Ancho en pixeles
	height	Altura en pixeles
location	protocol	Inicio de la dirección
	hash	Nombre del enlace de la URL
	host	Nombre del computador
	href	Dirección completa
	port	Puerto de comunicaciones
document	bgColor	Color de fondo
	fgColor	Color de texto
	vlinkColor	Color de los enlaces visitados
	alinkColor	Color del enlace en el momento de selección
	Métodos	
	clear()	Borra la página del navegador
	close()	Cierra el documento
	write()	Permite escribir en un documento
history	back()	Carga la URL anterior al actual
	forward()	Carga la URL siguiente
	go()	Muestra una URL de la lista de history

Figura 2.1.9 Jeraquia de window en JavaScript

Hojas de estilo CSS

Las hojas de estilo en cascada o CSS (Cascading Style Sheet) son muy importantes en los diseños actuales de páginas web

Muchos de los módulos de CSS están en constante variación y estabilidad por lo que es recomendable consultar las listas de CSS del World Wide Web Consortium (W3C).

Las hojas de estilo se definen a manera de los scripts de JavaScript. La definición está delimitada por las etiquetas

<STYLE> y </STYLE>

Por ejemplo:

```
<HEAD>
<STYLE TYPE = “text/css”>
<!--
Estilos
//-->
</STYLE>
</HEAD>
```

Las etiquetas <DIV> y </DIV> se utilizan para definir estilos en una sección.
Las etiquetas <SPAN> y </SPAN> se utilizan para definir estilos en un párrafo.

Las etiquetas style pueden aplicarse a cualquier etiqueta HTML en específico y solo modificaran la etiqueta donde se define.

Los estilos genéricos se definen así:

Etiqueta{propiedad:valor;propiedad:valor……}

```
<HEAD>
<STYLE TYPE = “text/css”>
<!--
H1{color:#205596}
A{ color:#303030;Font-style:italic}
//-->
</STYLE>
</HEAD>
```

Aquí todas las etiquetas H1 tendrán el color #205596 gris y las A estarán en color azul e itálico.
Los estilos también pueden ser guardados en hojas de estilo y aplicada a multitud de documentos HTML. La terminación para las hojas de estilo es .css y no necesita las etiquetas HTML, BODY O HEAD y se llaman por referencia usando la etiqueta LINK en la cabecera del documento.

```
<LINK REL=”stylesheet” type=”text/css” HREF=”estilo.css”>
```

Los estilos se pueden definir también mediante las etiquetas CLASS Y ID

```
<HEAD>
<STYLE TYPE = “text/css”>
<!--
.miestilo {color:#205596;text-aline:center}
.izquierda {text-aline: left;Font-style:italic}
//-->
</STYLE>
</HEAD>
<BODY>
<P CLASS=miestilo>
Un texto.
<P CLASS=izquierda>
Texto dos.
</P>
```

```
</BODY>



<HEAD>
<STYLE TYPE = “text/css”>
<!--
#miestilo {color:#205596;text-aline:center}
#izquierda {text-aline: left;Font-style:italic}
//-->
</STYLE>
</HEAD>
<BODY>
<P ID=miestilo>
Un texto.
<P ID=izquierda>
Texto dos.
</P>
</BODY>
```

Existe un gran número de propiedades que pueden utilizarse en las hojas de estilo estas se pueden agruparse en :

Grupo	Descripción CSS
animacion y transcicion	Animación y transición.
animaciones	Habilitar animaciones.
At-rules	at-rules.
Bordes y fondos	Fondos y bordes.
Modelo básico de caja	Modelo básico de caja
Interface básica de usuario	Interface básica de usuario
Color	color.
CSSOM	Modelo de objetos de CSS (CSSOM)
Adaptación de dispositivo	CSS Device Adaptation.
Estilo DOM y APIs	Document Object Model (DOM) colecciones, métodos, objetos, propiedades.
Exclusiones	Exclusiones
Caja flexible	Layout de ("Flexbox") .
Fuentes	Fuentes (fonts).
Pantalla completa	Fullscreen API.
Generar y remplazar contenido	Generar y remplazar contenido
Gradientes	Gradientes
Grid Layout	Grid Layout.
Modo contrastado	JavaScript for High Contrast Mode en Windows.
Media Queries	Queries.
Extensiones de Microsoft para CSS	Extensiones de Microsoft para CSS.

Layout de Multi-columnas	Tablas multicolumnas.
Paged Media	Paged Media
Regiones	Regiones
Text Ruby	Text Ruby
Selectores	Selectores
Selectores API	Selectores API
Tablas	Tablas
Texto	Texto
Tocar: expander y panear	Tocar: expander y panear
Transformar	Transformar
Transiciones	Transiciones
Efectos Visuales	Efectos Visuales
Modelo de formateo visual	Modelo de formateo visual
Modos de escritura	Modos de escritura

Figura 2.1.10 Grupos CSS en JavaScript

El nombre de las propiedades puede consultarse el índice alfabético en el anexo *Índice alfabético de CSS*

Visual Basic.NET

Visual Basic .NET (VB.NET) es un lenguaje de programación orientado a objetos que se puede considerar una evolución de Visual Basic implementada sobre el framework .NET. Su introducción resultó muy controvertida, ya que debido a cambios significativos en el lenguaje VB.NET no es retro compatible con Visual Basic, pero el manejo de las instrucciones es similar a versiones anteriores de Visual Basic, facilitando así el desarrollo de aplicaciones más avanzadas con herramientas modernas. Para mantener eficacia en el desarrollo de las aplicaciones. La gran mayoría de programadores de VB.NET utilizan el entorno de desarrollo integrado Microsoft Visual Studio en alguna de sus versiones (desde el primer Visual Studio .NET hasta Visual Studio .NET 2013, que es la última versión de Visual Studio para la plataforma .NET),

Visual Studio 2010

El 12 de Abril del 2010, Microsoft publica Visual Studio 2010 y .NET Framework versión 4. Compatible con Visual Basic.net, con una interfaz rediseñada, más sencilla y con soporte para diseño de aplicaciones en Windows 7.

Microsoft proporciona servicios Web XML en la plataforma .NET Framework, diseñado desde el principio para permitir a los programadores escribir e implementar fácilmente aplicaciones Web complejas.
Visual Basic .NET es un pilar de .NET Framework y otro paso hacia adelante en la evolución del lenguaje. Es un lenguaje de programación de alto nivel de .NET Framework y proporciona el punto de entrada más fácil a .NET.

Gramática léxica

La compilación de un programa de Visual Basic .NET convierte la cadena sin formato de caracteres Unicode en una serie de líneas lógicas compuestas de un conjunto ordenado de símbolos (token) léxicos. La amplitud de la línea lógica abarca desde el inicio de una secuencia o un terminador de línea hasta el siguiente terminador de línea que no esté precedido de una continuación de línea o hasta el final de la secuencia.

Start ::= [LogicalLine+]
LogicalLine ::= [LogicalLineElement+] [Comment] LineTerminator
LogicalLineElement ::= WhiteSpace | Token
Token ::= Identifier | Keyword | Literal | Separator | Operador

Directivas de pre procesamiento
Una vez analizado léxicamente el archivo, se producen varias clases de pre procesamiento del código fuente. La más importante, la compilación condicional, determina el código fuente que va a procesar la gramática sintáctica; otros dos tipos de directivas, las directivas de origen externo y las directivas de región, proporcionan meta información acerca del código fuente.

Archivos de código fuente
Un programa de Visual Basic .NET consta de uno o más archivos de código fuente. Cuando se compila un programa, todos los archivos de código fuente se procesan juntos; de esta forma, estos archivos dependen unos de otros, posiblemente de forma circular, sin ningún requisito de declaración adelantada. Por tanto, el orden textual de las declaraciones en el texto del programa no tiene en general ninguna importancia.
Un archivo de código fuente se compone de un conjunto opcional de atributos, directivas de opción y directivas de importación, seguidas por un cuerpo. El cuerpo de las funciones del archivo de código fuente funciona como declaración de espacio de nombres implícito del espacio de nombres global, lo que quiere decir que todas las declaraciones en el nivel superior de un archivo de código fuente se colocan en el espacio de nombres global.

Source ::=
[OptionDirective+]
[ImportsDirective+]
[Attributes]
[NamespaceBody]

Tipos
Las dos categorías fundamentales de tipos en Visual Basic .NET son los tipos de valores y tipos de referencia. Los tipos primitivos, enumeraciones y estructuras son tipos de valores. Las clases, cadenas, módulos estándar, interfaces, matrices y delegados son tipos de referencia.
Salvo una excepción, todos los tipos son tipos de valores o tipos de referencia. El tipo de la raíz Object, que es un alias para el System.Object, es especial, porque no es un tipo de referencia ni un tipo de valor y puede que no se pueda crear una instancia con él. Por tanto, una variable del tipo Object puede contener tanto un tipo de valor como un tipo de referencia.

TypeName ::= QualifiedIdentifier | Object | PrimitiveTypeName | ArrayTypeName

Miembros de tipo
Los miembros de tipo pueden ser métodos, constructores, eventos, constantes, variables y propiedades.

Instrucciones
Las instrucciones representan código ejecutable.

La ejecución de un programa se inicia con un método Shared que determina el entorno de compilación. Cuando el método termina, el programa termina.
Un método se ejecuta inicializando primero todos sus parámetros y variables locales a sus valores correctos y después ejecutando el bloque del cuerpo del método. Cuando se ha ejecutado el bloque del método, la ejecución vuelve al llamador del método.

Expresiones
Una expresión es una secuencia de operadores y operandos que especifican un cálculo. Normalmente una expresión se evalúa como un valor en tiempo de ejecución.

```
Expression ::=
  SimpleExpression |
  InvocationExpression |
  MemberAccessExpression |
  IndexExpression |
  NewExpression |
  CastExpression |
  OperatorExpression
```

Operadores
Hay dos clases de operadores. Los operadores unarios tienen un operando y utilizan notación de prefijo (por ejemplo, –x). Los operadores binarios tienen dos operandos y notación de infijo (por ejemplo, x + y). En tiempo de ejecución, siempre se evalúa primero el operando de la izquierda. El orden de evaluación de los operandos de una expresión se determina por la prioridad y asociatividad de los operadores.

```
OperatorExpression ::= UnaryOperatorExpression | BinaryOperatorExpression
BinaryOperatorExpression ::=
  ArithmeticOperatorExpression |
  RelationalOperatorExpression |
  LikeOperatorExpression |
  ConcatenationOperatorExpression |
  ShortCircuitLogicalOperatorExpression |
  LogicalOperatorExpression |
  ShiftOperatorExpression
```

Conversiones
La conversión es el proceso de cambiar un valor de un tipo a otro. Las conversiones pueden ser de ampliación o de restricción: las conversiones de ampliación nunca producen desbordamiento, mientras que las conversiones de restricción suponen la pérdida de información y pueden producir errores.
Las conversiones de referencia, ya sean de ampliación o de restricción, nunca cambian la identidad de referencia del objeto que se va a convertir. Es decir, si bien una conversión de referencia puede cambiar el tipo de un valor, nunca cambia el valor en sí. Las conversiones de valor, por otra parte, pueden cambiar el valor mismo. Las expresiones de conversión de tipo permiten el uso de conversiones de ampliación y de restricción

En el anexo de Gramática de Visual Basic NET puede observarse la gramática léxica de las principales estructuras de VB.

VBScript

VBScript (abreviatura de Visual Basic Script Edition) es un lenguaje interpretado por el Windows Scripting Host de Microsoft. Su sintaxis refleja su origen como variación del lenguaje de programación Visual Basic. Ha logrado un apoyo significativo por parte de los administradores de Windows como herramienta de automatización, ya que, conjunta y paralelamente a las mejoras introducidas en los sistemas operativos Windows donde opera fundamentalmente, permite más margen de actuación y flexibilidad que el lenguaje batch desarrollado a finales de los años 1970 para el MS-DOS

El crecimiento del uso de las tecnologías de Internet ha supuesto un significativo avance para este lenguaje, dado que es parte fundamental de la ejecución de aplicaciones de servidor programadas en ASP (Active Server Pages), las cuales estuvieron en auge en el período 1997-2003, declinando actualmente en favor de tecnologías de código gestionado y máquinas virtuales, más seguras en la ejecución de procesos, y por tanto, más adaptadas para ejecuciones en entornos públicamente accesibles y distribuidos. Microsoft ha intentado competir mediante esta tecnología también en entornos de cliente, donde el lenguaje más utilizado es JavaScript o su versión estandarizada ECMAScript, sin éxito. Actualmente Microsoft no ha puesto a disposición pública nuevas versiones del lenguaje, en favor de la tecnología .NET en la que se incluye el lenguaje hermano Visual Basic, dentro del entorno de ejecución de la plataforma.NET (CLR, o Common Language Runtime). Con el advenimiento de .NET framework, el equipo de desarrollo tomó la decisión de soportar este entorno en ASP.NET para el desarrollo web y por lo tanto no hay nuevas versiones del motor de VBScript. Sin embargo sigue siendo muy útil en gestión de estaciones de trabajo y servidores en Windows.

Interpretación

VBScript es interpretado por el motor de vbscript.dll,8 que puede ser invocado por el motor ASP (asp.dll) en un entorno web, por un ejecutable (aplicación HTML) y por Internet Explorer durante la navegación web. Se puede guardar en archivos independientes y éstos tienen típicamente la extensión .vbs.
Cuando se emplea en Internet Explorer se procesa el código contenido en el documento HTML. VBScript también puede usarse para crear aplicaciones HTML independientes (extensión .hta), que necesitan Internet Explorer 5.0 o superior para poder ser ejecutados. Los desarrolladores de aplicaciones en web suelen preferir JavaScript debido a su mayor compatibilidad con otros navegadores de Internet, ya que VBScript sólo está disponible para el navegador de Microsoft Internet Explorer y no en otros como Firefox, Google Chrome u Opera (en sus diferentes versiones).

Sintaxis
VBScript similar a Visual Basic (las funciones trabajan exactamente igual), pero algunas funciones cambian radicalmente; por ejemplo:

- Execute (no existe en Visual Basic, y no tiene similar)
- Do...Loop Until (en este caso se cambia el orden poniendo Do Until...Loop)
- VBScript.Sleep (esto se sustituye por una API llamada Sleep).
- VBScript.Quit (se sustituye por UnLoad Me)

Ejemplo de una sencillísima página web que incluye código VBScript:

```
<HTML>
<HEAD>
<TITLE>Cuadro de mensaje</TITLE>
```

```
</HEAD>
<BODY>
<SCRIPT LANGUAGE = "VBScript">
   MSGBOX ("Ejemplo de mensaje")
</SCRIPT>
</BODY>
</HTML>
```

Programación VBScript y ASP

Ejemplo de página ASP:

```
<HTML>
<HEAD>
<TITLE>Ejemplo de ASP</TITLE>
</HEAD>
<BODY>
<HR>
   Esta página es una prueba de ASP
   <%
   Mi_Email = "goldenchip@informaticos.com"
   Response.Write (Mi_Email)
   %>
<HR>
</BODY>
</HTML>
```

En este código vemos dos líneas con una sintaxis sospechosamente parecida a la de VBScript:

```
Mi_Email = "goldenchip@informaticos.com"
Response.Write (Mi_Email)
```

Y, en efecto. Se trata de este lenguaje. Pero es **VBScript de servidor**. El código ya no aparece entre los tags <SCRIPT>y </SCRIPT> sino entre **<%** y **%>** . Estos últimos tags no existen en HTML, por lo que no pueden dar lugar a ningún tipo de confusión. El objeto Response es un objeto de servidor equivalente al objeto document de cliente. La ejecución de ésta página, si todo está correctamente configurado, mostrará en la pantalla lo siguiente:

```
goldenchip@informaticos.com
```

Sin embargo, la ventaja es que el usuario no podrá ver el código nativo de nuestra página.

La lista de los principales objetos de VBScript es:

- Objeto window
- Objeto location
- Objeto document
- Objeto navigator
- Objeto frame
- Objeto history
- Objeto link

- Objeto anchor
- Objeto form

Comentarios

En VBScript se pueden insertar comentarios en el código con el fin de facilitar la legibilidad y el mantenimiento del mismo. El intérprete ejecuta el código ignorando los comentarios. Al contrario de lo que muchos programadores novatos piensan, los comentarios no afectan a la velocidad ni a ningún otro aspecto de la ejecución, por lo que podemos usarlos libremente, con toda la profusión necesaria para que nuestro código sea fácil de comprender. Para insertar un comentario, tecleamos la palabra clave REM o bien una comilla simple. Todo lo que haya en la línea de ahí en adelante será considerado por el intérprete como un comentario.

Datos y Variables

En VB solo existe un tipo general de datos que se conoce con el nombre de **Variant**. En otros lenguajes existen datos de tipo String (Cadena) para almacenar contenidos alfanuméricos, distintos tipos de datos numéricos enteros y en coma flotante, datos booleanos, etc. Esta característica es muy útil, ya que permite reasignar un valor de un tipo a una variable de otro tipo. En la actualidad es el único lenguaje de alto nivel que implementa esta característica Los datos se clasifican en subtipos en función del contenido en un momento dado. Así se logra toda la funcionalidad de gestión de datos en lenguajes de alto nivel, pero con una mayor flexibilidad. Para cambiar una variable de un subtipo a otro, es suficiente con asignarle un dato de diferente tipo. Por ejemplo. El siguiente fragmento de código daría un error en lenguajes como C++ o Java; sin embargo, en VB es absolutamente correcto:

```
Variable = 1
' más código
' más código
' más código
' más código
Variable = "cadena"
```

Al pertenecer todos los datos a un tipo único, cuando le damos un valor a un dato se constituye, automáticamente, del subtipo adecuado para ese valor. Este proceso es totalmente transparente al programador quien, de esta forma, no necesita preocuparse de declarar un dato como de uno u otro tipo.

Variables

Los nombres de variables deben empezar con una letra y pueden tener letras, números o el signo de subrayado (único signo de puntuación que se admite en el nombre de una variable). Las letras que formen parte del nombre de una variable deberán ser de la alfabetización internacional, no de la española o específica de algún idioma en particular. Así pues, no deberá haber en un nombre de variable letras como la ñ, letras acentuadas, la ç, etc. Un nombre de variable no deberá contener caracteres especiales (como p.e. $,%,&,^, etc.). Tampoco deberá contener espacios en blanco, puntos, comas, ni ningún otro signo de puntuación. Por supuesto, el contenido de una variable alfanumérica SI podrá contener cualquier cosa que deseemos. Además deberemos tener cuidado de no emplear como nombres de variable las palabras clave del lenguaje.

Para usar una variable es necesario dar dos pasos: declararla e inicializarla. La declaración es la forma de decirle al lenguaje que se va a usar una variable y se hace con la palabra reservada DIM, seguida del nombre de la variable. Así:

DIM variable

Esto reserva espacio en memoria para la variable. Sin embargo aún no le hemos asignado ningún contenido. En realidad tiene un contenido de subtipo Null (nulo) . La inicialización de la variable será la que le asigne su primer contenido aunque, éste podrá cambiar a lo largo de la ejecución. La inicialización es, simplemente una asignación. Por ejemplo:

variable = "Cacharro"

La declaración de la variable es opcional. Es decir, si no se hace la declaración de una variable, ésta se llevará a cabo, de forma automática al realizar la inicialización. Sin embargo es conveniente realizar las declaraciones de forma manual (escribiendo la instrucción DIM), a fin de incrementar el nivel de estructuración de nuestros programas y facilitar la legibilidad de los mismos. La declaración debe ir SIEMPRE antes de la inicialización. Lo correcto es realizar la declaración de todas las variables al principio de nuestro código. Existe una forma de asegurarnos de que tengamos que hacer las oportunas declaraciones. Es incluyendo la instrucción OPTION EXPLICIT en nuestro código. Si incluimos esa línea, el programa no podrá usar ninguna variable que no haya sido declarada.

Matrices

Una matriz es un conjunto de variables que reciben todas el mismo nombre, identificamos cada variable o elemento usamos un índice. El primer elemento se le conoce con el número 0 para cinco elementos, del 0 al 4 (0, 1, 2, 3 y 4). Una vez declarada, no podrá redimensionarse, una matriz debe ser declarada con el número MÁXIMO de elementos que deberá contener. Si no sabemos cuántos elementos podrá contener una matriz como máximo, no podremos usar matrices.
Las matrices, también llamadas arrays se emplean para almacenar un conjunto de datos relacionados entre sí

La forma de declarar una matriz bidimensional es la siguiente:

DIM matriz_bidimensional (10,20)

Se pueden usar matrices de más de dos dimensiones (hasta un máximo de 60 dimensiones)

Salida por pantalla

La instrucción MSGBOX () Nos permite sacar una cadena alfanumérica, el contenido de una variable o combinaciones de ambas en un cuadro don el aspecto típico de Windows. Este cuadro está dotado de un botón Aceptar que el usuario debe pulsar para que prosiga la ejecución, al pulsar el botón el cuadro desaparece. VBScript es un lenguaje orientado a objetos. Y el documento activo es un objeto llamado document. Este objeto tiene un método de escritura, llamado write (), que nos permite escribir en la pantalla.

Uso de Document.Write

```
<HTML>
<HEAD>
```

```
<TITLE>Prueba de document.write ()</TITLE>
</HEAD>
<BODY>
<SCRIPT LANGUAGE = "VBScript">
   DOCUMENT.WRITE ("Hola desde VBScript")
</SCRIPT>
</BODY>
</HTML>
```

El resultado de este código sería ver en pantalla la frase Hola desde VBScript. Como en el caso de MSGBOX, la impresión puede ser una combinación de cadena /s alfanumérica /s y/o variable /s

Entrada por teclado

La instrucción INPUTBOX ().abre una caja de diálogo

En esta caja tiene lo siguiente:

1. La pregunta que se le hace al usuario.
2. Una línea en blanco (llamada "Caja de texto" donde el usuario introduce su respuesta). Esta caja de texto puede aparecer vacía o con una posible respuesta por defecto. En este último caso la respuesta por defecto aparece seleccionada (texto blanco sobre fondo azul).
3. Un botón de Aceptar. Una vez introducida la respuesta el usuario pulsa la tecla INTRO del teclado o el botón Aceptar para validarla.
4. Un botón Cancelar.
5. En la parte superior de la ventana esta la barra de título.

La sintaxis de INPUTBOX (). Es la siguiente:

INPUTBOX (Pregunta, Título, Respuesta, pos x, pos y)

Como vemos esta instrucción puede recibir varios parámetros separados por comas. Son los siguientes:

1. Pregunta. Es la pregunta que se le formula al usuario y a la que deberá responder.
2. Título. Es un literal que aparecerá en la barra de título.
3. Respuesta. Es la respuesta por defecto que queremos ofrecerle al usuario.
4. pos x - pos y. Son las coordenadas donde queremos que se sitúe la esquina superior izquierda de la caja de diálogo. Estas coordenadas se expresan en twips.

Condicionales

```
IF (condición) THEN
   Instrucciones1
ELSE
   Instrucciones2
END IF

SELECT CASE cantidad
   CASE 1:
```

```
    Instrucciones1
    CASE 2:
    Instrucciones2
.CASE ELSE:
    Instrucciones2
END SELECT
```

Ciclos

```
FOR contador = valor_inicial TO valor_final STEP incremento
    Instrucciones
NEXT

DO WHILE (condición)
    Instrucciones
LOOP

DO UNTIL (condición)
    Instrucciones
LOOP

WHILE (condición)
    Instrucciones
WEND
```

Funciones

Una función es un fragmento de código que recoge unos parámetros (valores que se le pasan para su ejecución) y devuelve un resultado. El código de la función se incluye entre las palabras clave FUNCTION y END FUNCTION. Todo lo que haya entre estas dos líneas será considerado como parte de la función. Además, a la función se le asigna un nombre, por el que se la invocará desde el código y será el nombre del resultado que devuelva. Así pues, en VBScript una función sólo puede devolver un resultado.

Ejemplo

```
FUNCTION sumar (sumando_1, sumando_2)
    sumar = (CLNG(sumando_1) + CLNG(sumando_2))
END FUNCTION
```

Procedimientos

Un procedimiento se parece en su concepción y uso a una función, con la diferencia fundamental de que no devuelve ningún resultado. Se incluyen entre las palabras clave SUB y END SUB. Para invocar al procedimiento, simplemente teclearemos su nombre en una línea de comando, como si fuera una instrucción.
Existen dos formas de realizar la invocación. El nombre del procedimiento seguido de los parámetros necesarios, separados por comas y sin paréntesis. Así:

```
Procedimiento parám1, parám2, parám3, ... , parámN
```

La segunda

CALL procedimiento (parám1, parám2, parám3, ... , parámN)

Jerarquía de objetos

En VBScript son importantes cuatro conceptos para comprender la POO y la jerarquía de objetos.
OBJETO: Es cada una de los elementos que se gestionan en una página web o en cualquier aplicación informática que tienen una identidad propia. Un objeto podría ser la ventana de navegación, o el documento activo, o un campo de un formulario, o una tabla, etc.

PROPIEDAD: Es cada una de las características de un objeto. Una propiedad sería el color de fondo de una tabla, la barra de estado de la ventana de navegación, etc. También llamamos propiedad a un objeto derivado de otro. Se dice que el objeto derivado es propiedad del objeto padre.

METODO: Es una operación que se puede realizar dentro de un objeto. Por ejemplo. Cerrar una ventana sería un método del objeto ventana. Poner el foco en un campo de formulario sería un método del objeto que representa a ese campo en concreto.

EVENTO: Un evento es la previsión de que el usuario realice una determinada acción. En realidad el usuario puede realizar la acción o no realizarla, pero se deja prevista en la programación la posibilidad de que la realice. Un evento sería pasar el ratón sobre una imagen determinada, hacer clic en una parte de la pantalla, pulsar una tecla, etc. También existe la posibilidad de prever eventos del sistema. Un evento de sistema es, por ejemplo, la carga de una página, o un error en un proceso. Resumiendo: un evento se dispara cuando ocurre la acción prevista por el mismo.

La lista de los principales objetos de VBScript es:

- Objeto window
- Objeto location
- Objeto document
- Objeto navigator
- Objeto frame
- Objeto history
- Objeto link
- Objeto anchor
- Objeto form

Los principales métodos y propiedades son:

Objeto	Propiedad	Explicacion
window		
	defaultStatus	Se refiere al mensaje que aparecerá por defecto en la barra de estado.
	document	Representa al documento HTML en ejecución en ese momento.
	frames []	Matriz que contiene los frames de la ventana.
	history	Registro histórico de las páginas visitadas en la actual sesión de uso de Internet.
	length	Contiene el número total de frames de la ventana.

	location	Representa a la dirección (URL) actual de Internet.
	name	Contiene el nombre de la ventana activa.
	navigator	Navegador que estamos utilizando.
	self	Es el mismo objeto window.
	status	Mensaje que aparece en la barra de estado en un momento determinado.
	window	La ventana activa o a otra ventana o sub-ventana de navegación.
Metodos		
	close ()	Permite cerrar la ventana activa. Su sintaxis es self.close()
	open ()	Permite abrir una nueva ventana, como sub ventana de la actual. Su sintaxis es: `nueva_ventana = window.open ("URL", "Target", "Opciones")`
Propiedades		
toolbar	booleano	Ventana con barra de herramientas.
location	booleano	Ventana con barra de direcciones.
directories	booleano	Ventana con directorios.
Status	booleano	Ventana con barra de estado
menubar	booleano	Ventana con barra de menús.
scrollbars	booleano	Ventana con barras de desplazamiento.
resizable	booleano	Ventana de tamaño redefinible por el usuario.
width	píxeles	Anchura de la ventana
height	píxeles	Altura de la ventana.
top	píxeles	Posición Y de la ventana
left	píxeles	Posición x de la ventana
Eventos		
	onLoad	Se ejecuta cuando se carga la página.
	onUnload	Se ejecuta cuando se descarga (se cierra) la página.
document		
Propiedades		
	alinkColor	Representa el color de los enlaces activos.
	bgColor	Representa el color de fondo del documento.
	fgColor	Representa el color del texto
	lastModified	Representa la fecha de la última modificación.
	linkColor	Representa el color de los enlaces.
	location	Representa la URL del documento.
	title	Representa el título del documento.
	vlinkColor	Representa el color de los enlaces visitados.
Metodos		
	Write ()	Escribe un texto.
	Writeln()	Escribe una línea de texto.
form		
Propiedades		
	action	Representa la URL donde está el programa encargado de procesar un formulario (al que se llama al activar el botón Submit).
	length	Es el número de elementos del formulario.

	method	Es el método de envío (GET o POST).
Metodos		
	submit	Se usa para forzar el envío.
Eventos		
	onSubmit	Se produce cuando se pulsa el botón Submit del formulario.
location		
Propiedades		
	href	Representa la propia URL.
	pathname	Representa la ruta del disco del servidor donde se aloja la página.
navigator		
Propiedades		
	appName	Es el nombre del navegador.
	appVersion	Se refiere a la versión del navegador.
history		
Propiedades		
	length	Representa la cantidad total de páginas visitadas.
Metodos		
	back ()	Navega a la página anterior.
	forward()	Navega a la página siguiente.
	go (n)	Navega n páginas hacia delante (o hacia atrás, si n es negativo).
Tipos de evento		Principales eventos que se pueden asociar a una imagen, hipervínculo, cadena de texto, etc
Raton		
	ONCLICK	Se activa con un botón del ratón.
	ONDBLCLICK	Se activa si se hace un doble click.
	ONMOUSEDOWN	Se activa si se pulsa el botón izquierdo del mouse.
	ONMOUSEMOVE	Se activa si se mueve el mouse.
	ONMOUSEOVER	Se activa cuando el puntero se sitúa sobre el objeto que incluye al evento.
Teclado		
	ONKEYDOWN	Se activa si se pulsa una tecla cualquiera.
	ONKEYPRESS	Se activa si se pulsa y suelta una tecla.
	ONKEYUP	Se activa cuando se suelta una tecla pulsada.
	ONHELP	Se activa si se pulsa la tecla de ayuda (normalmente F1).
Enfoque		
	ONFOCUS	Se activa cuando se entra en el ámbito de un elemento al que está asociado el evento.
	ONBLUR	Se activa al abandonar el ámbito del elemento al que está asociado.

Formulario		
	ONRESET	Se activa al pulsar un botón de reset de un formulario.
	ONSUBMIT	Se activa al enviar un formulario.
Carga de pagina		
	ONABORT	Se activa cuando se aborta la carga de la página.
	ONERROR	Se activa cuando se produce un error inesperado durante la carga de la página.
	ONLOAD	Se activa cuando se carga la página.
	ONUNLOAD	Se activa cuando el usuario descarga la página (es decir, carga otra o pretende salir del navegador).
	ONAFTERUPDATE	Se activa si se actualiza o recarga la página.

Figura 2.1.11 Objetos, Metodos, Propiedades , Eventos VBScript

Subtipos de datos

A continuación aparecen listados los subtipos de datos aceptados por VBScript. Son los siguientes:

Tipo	Descripción
String	Datos de tipo cadena (también llamado alfanuméricos).
Byte	Números enteros del 0 al 255.
Integer	Números enteros del -32.768 al 32.767.
Long	Números enteros del -2.147.483.648 al 2.147.483.647.
Single	Números en coma flotante de simple precisión.
Double	Números en coma flotante de doble precisión.
Currency	Números en coma flotante del -922.337.203.685.477,5808 al 922.337.203.685.477,5808.
Boolean	Datos lógicos verdadero o falso (true o false).
Null	Un dato Variant sin definir contenido de ningún subtipo.
Date	Un valor de Fecha / Hora.
Object	Contiene la representación de un objeto.
Error	Identifica los errores mediante un número.

Figura 2.1.12 Subtipos de datos VBScript

Operadores

Los operadores que maneja VBScript, se presentan, clasificados según el tipo de operación que realizan. Los operadores son unos signos que se usan para realizar operaciones matemáticas, comparativas, concatenarías o lógicas.

Operador	Descripción
+	Suma aritmética. Concatenación de cadenas.
-	Resta aritmética
*	Multiplicación aritmética.
/	División (muestra el resultado en coma flotante).

\	División (muestra la parte entera del resultado).
^	Potenciación
Mod	Obtiene el módulo de una división
=	Igual que
<>	No igual que
>	Mayor que
<	Menor que
>=	Mayor o igual que
<=	Menor o igual que
&	Concatenación de datos de diferente subtipo.
And	Devuelve true si las dos expresiones a ambos lados del operador son verdaderas.
Or	Devuelve true si alguna de las expresiones es verdadera
Xor	Devuelve true si sólo una de las dos expresiones es verdadera.
Not	Cambia una expresión verdadera en una falsa y viceversa.Devuelvetrue si la expresión es falsa.

Figura 2.1.13 Operadores en VBScript

HTML

HyperText Markup Language («lenguaje de marcas de hipertexto»), hace referencia al lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia del software que conecta con la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (denominado código HTML) para la definición de contenido de una página web, como texto, imágenes, videos, juegos entre otros. Es un estándar a cargo de la W3C. Se considera el lenguaje web más importante siendo su invención crucial en la aparición, desarrollo y expansión de la World Wide Web. Es el estándar que se ha impuesto en la visualización de páginas web y es el que todos los navegadores actuales han adoptado.

El lenguaje HTML basa su filosofía de desarrollo en la diferenciación. Para añadir un elemento externo a la página (imagen, vídeo, script, entre otros.), este no se incrusta directamente en el código de la página, sino que se hace una referencia a la ubicación de dicho elemento mediante texto. De este modo, la página web contiene sólo texto mientras que recae en el navegador web (interpretador del código) la tarea de unir todos los elementos y visualizar la página final. Al ser un estándar, HTML busca ser un lenguaje que permita que cualquier página web escrita en una determinada versión, pueda ser interpretada de la misma forma (estándar) por cualquier navegador web actualizado.

El HTML se escribe en forma de «etiquetas», rodeadas por corchetes angulares (<,>,/). El HTML también puede describir, hasta un cierto punto, la apariencia de un documento, y puede incluir o hacer referencia a un tipo de programa llamado script, el cual puede afectar el comportamiento de navegadores web y otros procesadores de HTML.

HTML también sirve para referirse al contenido del tipo de MIME text/HTML o todavía más ampliamente como un término genérico para el HTML, ya sea en forma descendida del XML o en forma descendida directamente de SGML.

HTML consta de varios componentes vitales, entre ellos los elementos y sus atributos, tipos de data y la declaración de tipo de documento.

Elementos

Los elementos son la estructura básica de HTML. Tienen dos propiedades básicas:

- Atributos o propiedad y
- contenido.

Cada atributo y contenido tiene ciertas restricciones para que se considere válido al documento HTML. Un elemento generalmente tiene una etiqueta de inicio (por ejemplo, <nombre-de-elemento>) y una etiqueta de cierre (por ejemplo, </nombre-de-elemento>). Los atributos del elemento están contenidos en la etiqueta de inicio y el contenido está ubicado entre las dos etiquetas (por ejemplo, <nombre-de-elemento atributo="valor">Contenido</nombre-de-elemento>). Algunos elementos, tales como
, no tienen contenido ni llevan una etiqueta de cierre.

El marcado estructural

Describe el propósito del texto. Por ejemplo, <h2>Golf</h2> establece «Golf» como un encabezamiento de segundo nivel, el cual se mostraría en un navegador de una manera similar al título «Marcado HTML» al principio de esta sección. El marcado estructural no define cómo se verá el elemento, pero la mayoría de los navegadores web han estandarizado el formato de los elementos. Puede aplicarse un formato específico al texto por medio de hojas de estilo en cascada.

El marcado presentacional

Describe la apariencia del texto, sin importar su función. Por ejemplo, negrita indica que los navegadores web visuales deben mostrar el texto en negrita, pero no indica qué deben hacer los navegadores web que muestran el contenido de otra manera (por ejemplo, los que leen el texto en voz alta). En el caso de negrita e itálica, existen elementos que se ven de la misma manera pero tienen una naturaleza más semántica: énfasis fuerte y énfasis. Es fácil ver cómo un lector de pantalla debería interpretar estos dos elementos. Sin embargo, son equivalentes a sus correspondientes elementos presentacionales: un lector de pantalla no debería mostrar más fuerte el nombre de un libro, aunque éste esté en itálicas en una pantalla. La mayoría del marcado presentacional ha sido desechada con HTML 4.0, en favor de hojas de estilo en cascada.

El marcado hipertextual

Se utiliza para enlazar partes del documento con otros documentos o con otras partes del mismo documento. Para crear un enlace es necesario utilizar la etiqueta de ancla <a> junto con el atributo href, que establecerá la dirección URL a la que apunta el enlace. Por ejemplo, un enlace que muestre el texto de la dirección y vaya hacia web.org podría ser de la forma http://www.web.org. También se pueden crear enlaces sobre otros objetos, tales como imágenes .

Estructura básica de elementos HTML

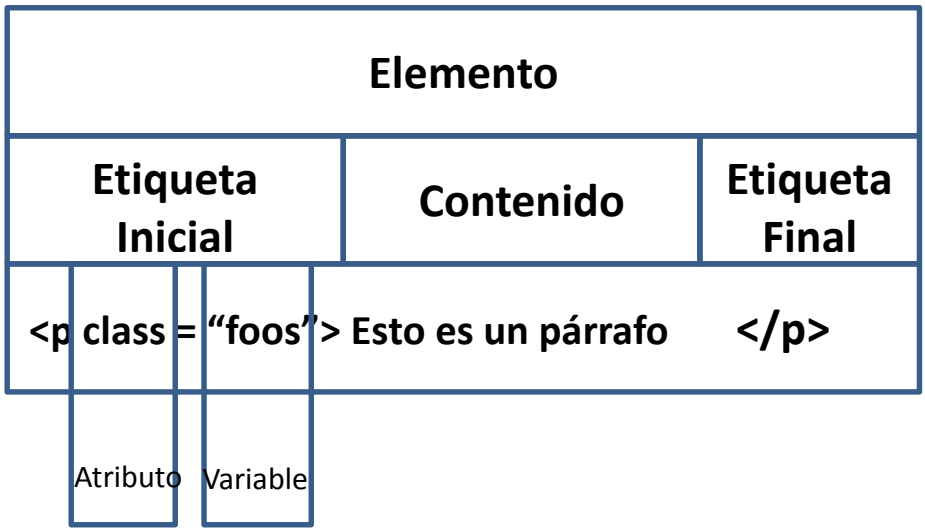


Figura 2.1.14 Elementos html

Un ejemplo de código HTML

- `<HTML>`: define el inicio del documento HTML, le indica al navegador que lo que viene a continuación debe ser interpretado como código HTML. Esto es así de facto, ya que en teoría lo que define el tipo de documento es el DOCTYPE, que significa la palabra justo tras DOCTYPE el tag de raíz.
- `<script>`: incrusta un script en una web, o llama a uno mediante `src="url del script"`. Se recomienda incluir el tipo MIME en el atributo type, en el caso de JavaScript `text/JavaScript`.
- `<head>`: define la cabecera del documento HTML; esta cabecera suele contener información sobre el documento que no se muestra directamente al usuario como, por ejemplo, el título de la ventana del navegador. Dentro de la cabecera `<head>` es posible encontrar:
  - `<title>`: define el título de la página. Por lo general, el título aparece en la barra de título encima de la ventana.
  - `<link>`: para vincular el sitio a hojas de estilo o iconos. Por ejemplo :`<link rel="stylesheet" href="/style.css" type="text/css">`.
  - `<style>`: para colocar el estilo interno de la página; ya sea hojas de estilo o u otros lenguajes similares. No es necesario colocarlo si se va a vincular a un archivo externo usando la etiqueta `<link>`.
  - `<meta>`: para metadatos como la autoría o la licencia, incluso para indicar parámetros http (mediante `http-equiv=""`) cuando no se pueden modificar por no estar disponible la configuración o por dificultades con server-side scripting.
- `<body>`: define el contenido principal o cuerpo del documento. Esta es la parte del documento HTML que se muestra en el navegador; dentro de esta etiqueta pueden definirse propiedades comunes a toda la página, como color de fondo y márgenes. Dentro del cuerpo `<body>` es posible encontrar numerosas etiquetas. A continuación se indican algunas a modo de ejemplo:
  - `<h1>` a `<h6>`: encabezados o títulos del documento con diferente relevancia.
  - `<table>`: define una tabla.
    - `<tr>`: fila de una tabla.
    - `<td>`: celda de una tabla (debe estar dentro de una fila).

- `<a>`: hipervínculo o enlace, dentro o fuera del sitio web. Debe definirse el parámetro de pasada por medio del atributo *href*. Por ejemplo: `<a href="http://www.example.com" title="Ejemplo" target="_blank" tabindex="1">Ejemplo</a>` se representa como Ejemplo).
- `<div>`: división de la página. Se recomienda, junto con css, en vez de `<table>` cuando se desea alinear contenido.
- `<img>`: imagen. Requiere del atributo *src*, que indica la ruta en la que se encuentra la imagen. Por ejemplo: `<img src="./imágenes/mifoto.jpg" />`.
- `<li><ol><ul>`: etiquetas para listas.
- `<b>`: texto en negrita (*etiqueta desaprobada. Se recomienda usar la etiqueta `<strong>`*).
- `<i>`: texto en cursiva (*etiqueta desaprobada. Se recomienda usar la etiqueta `<em>`*).
- `<s>`: texto tachado (*etiqueta desaprobada. Se recomienda usar la etiqueta `<del>`*).

La mayoría de etiquetas deben cerrarse como se abren, pero con una barra («/») tal como se muestra en los siguientes ejemplos:

- `<table><tr><td>Contenido de una celda</td></tr></table>`.
- `<script>Código de un script integrado en la página</script>`

Nociones básicas de HTML

El lenguaje HTML puede ser creado y editado con cualquier editor de textos básico, como puede ser Gedit en Linux, el Bloc de notas de Windows, o cualquier otro editor que admita texto sin formato como GNU Emacs, Microsoft Wordpad, TextPad, Vim, Notepad++, entre otros.
Existen, además, otros editores para la realización de sitios web con características WYSIWYG (What You See Is What You Get, o en español: «lo que ves es lo que obtienes»). Estos editores permiten ver el resultado de lo que se está editando en tiempo real, a medida que se va desarrollando el documento. Ahora bien, esto no significa una manera distinta de realizar sitios web, sino que una forma un tanto más simple, ya que estos programas, además de tener la opción de trabajar con la vista preliminar, tiene su propia sección HTML, la cual va generando todo el código a medida que se va trabajando. Algunos ejemplos de editores WYSIWYG son KompoZer, Microsoft FrontPage o Adobe Dreamweaver.

Combinar estos dos métodos resulta muy interesante, ya que de alguna manera se ayudan entre sí. Por ejemplo, si se edita todo en HTML y el desarrollador olvida algún código o etiqueta, basta con dirigirse al editor visual o WYSIWYG y se continúa ahí la edición o viceversa, ya que hay casos en que resulta más rápido y fácil escribir directamente el código de alguna característica que el usuario desea adherir al sitio que buscar la opción en el programa mismo.

Existe otro tipo de editores HTML llamados WYSIWYM que dan más importancia al contenido y al significado que a la apariencia visual. Entre los objetivos que tienen estos editores es la separación del contenido y la presentación, fundamental en el diseño web.

Versiones de HTML

Conforme se ha ido desarrollando el web se han desarrollado versiones del HTML:

Version	Year
HTML	1991
HTML 2.0	1995
HTML 3.2	1997
HTML 4.01	1999
XHTML	2000
HTML5	2014

Figura 2.1.15 Versiones html

Atributos HTML

Los atributos proveen información adicional a los elementos de HTML y pueden o no definirse en un elemento, siempre se especifican entre etiquetas de inicio y siempre con la definición

Nombre=”valor”

A continuación se describe una lista de atributos de HTML

Propiedad	Descripción
alt	Especifica el texto alternativo de una imagen
disabled	Especifica que un elemento puede ser des habilitado
href	Especifica la URL en una liga
id	Especifica un identificador para un elemento
src	Especifica la URL para una imagen
style	Especifica el estilo CSS
title	Especifica información extra sobre un elemento

Figura 2.1.16 Propiedades HTML

Ejemplo de HTML

```
<!DOCTYPE html>
<html>
<head>
<title>Mi pagina</title>
</head>
<body>

<h1>Mi primer encabezado</h1>
<p> Mi primer párrafo.</p>

</body>
</html>
```

Explicación del ejemplo

- La declaración **DOCTYPE** define el documento como un HTML
- El texto entre **<html>** and **</html>** describe un documento HTML

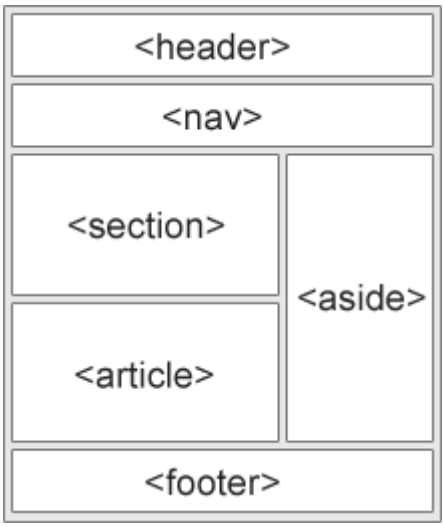
- El texto entre **<head>** and **</head>** provee información sobre el documento
- El texto entre **<title>** and **</title>** provee un titulo
- El texto entre **<body>** and **</body>** describe el contenido.
- El texto entre **<h1>** and **</h1>** define el encabezado
- El texto entre ** ** and ** ** describes un párrafo

HTML5

La recomendación de W3C HTML5 fue publicada el 28 Octubre 2014.

Nueva estructura de un Website usando HTML5

HTML5 establece nuevos objetos semánticos para definir una página:



- <header> - Define el encabezado de un a documento sección
- <nav> - Define un contenedor para los links de navegación
- <section> - Define una sección en un documento
- <article> - Define un artículo auto contenido
- <aside> - Define contenido aside como una barra lateral
- <footer> - Define un pie de pagina
- <details> - Define detalles adicionales
- <summary> - Define un encabezado para el elemento <details>

Figura 2.1.17 Estructura web en HTML5

Este ejemplo usa <header>, <nav>, <section>, and <footer> para crear una estructura múltiple

Ejemplo:

```
<body>

<header>
<h1>Galleria de la ciudad</h1>
</header>

<nav>
London<br>
Paris<br>
Tokyo
</nav>

<section>
<h1>London</h1>
<p>London es la capital de Inglaterra. Es la ciudad mas poblada del Reino Unido,
Con un área metropolitana de 13 millones de habitantes.</p>
<p>Establecida en el rio Thames, London ha sido el mayor asentamiento por dos mil
años,
```

```
Su historia se remonta hasta la era de los Romanos, quienes la llamaban Londinium.</p>
</section>

<footer>
Copyright © W3Schools.com
</footer>

</body>
```

HTML5 adiciono los siguientes elementos:

- <datalist>
- <keygen>
- <output>

Y adiciono o modificó los siguientes elementos en **form**

Elemento	Descripción
<form>	Define na forma HTML form para entradas de usuario
<input>	Define un control de entradas
<textarea>	Define un control multilinea de entradas (text area)
<label>	Define una etiqueta para el elemento <input>
<fieldset>	Agrupa elementos relacionados en una forma
<legend>	Define un caption para el elemento <fieldset>t
<select>	Define una lista drop-downt
<optgroup>	Define grupos de opciones en la listas drop-down t
<option>	Define na opción en la lista drop-downt
<button>	Define un botón clickable
<datalist>	Especifica una lista de opciones predefinidas para controles de entradas
<keygen>	Define un generador de campos
<output>	Define el resultado de un calculo

Figura 2.1.18 Elementos form en HTML5

En el anexo de HTML se encuentran in índice de los elementos de HTML y HTML 5

2. Bases de datos

Se le llama base de datos a los bancos de información que contienen datos relativos a diversas temáticas y categorizados de distinta manera, pero que comparten entre sí algún tipo de vínculo o relación que busca ordenarlos y clasificarlos en conjunto.

Una base de datos es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso. La mayoría de las bases de datos están en formato digital, que ofrece un amplio rango de soluciones al almacenamiento de datos.
Existen programas denominados sistemas gestores de bases de datos, abreviado SGBD (del inglés Database Management System o DBMS), que permiten almacenar y posteriormente

acceder a los datos de forma rápida y estructurada. Las propiedades de estos DBMS, así como su utilización y administración, se estudian dentro del ámbito de la informática.

Clasificación de bases de datos

Las bases de datos pueden clasificarse de varias maneras, de acuerdo al contexto que se esté manejando, la utilidad de las mismas o las necesidades que satisfagan.

Bases de datos estáticas
Son bases de datos únicamente de lectura, utilizadas primordialmente para almacenar datos históricos que posteriormente se pueden utilizar para estudiar el comportamiento de un conjunto de datos a través del tiempo, realizar proyecciones, tomar decisiones y realizar análisis de datos para inteligencia empresarial.

Bases de datos dinámicas
Son bases de datos donde la información almacenada se modifica con el tiempo, permitiendo operaciones como actualización, borrado y edición de datos, además de las operaciones fundamentales de consulta. Un ejemplo, puede ser la base de datos utilizada en un sistema de información de un supermercado.

Bases de datos bibliográficas
Sólo contienen un subrogante (representante) de la fuente primaria, que permite localizarla. Un registro típico de una base de datos bibliográfica contiene información sobre el autor, fecha de publicación, editorial, título, edición, de una determinada publicación, etc. Puede contener un resumen o extracto de la publicación original.

Bases de datos de texto completo
Almacenan las fuentes primarias, como por ejemplo, todo el contenido de todas las ediciones de una colección de revistas científicas.

Bases de datos de información química o biológica
Son bases de datos que almacenan diferentes tipos de información proveniente de la química, las ciencias de la vida o médicas. Se pueden considerar en varios subtipos:

- Las que almacenan secuencias de nucleótidos o proteínas.
- Las bases de datos de rutas metabólicas.
- Bases de datos de estructura, comprende los registros de datos experimentales sobre estructuras 3D de biomoléculas-
- Bases de datos clínicas.
- Bases de datos bibliográficas (biológicas, químicas, médicas): PubChem, Medline, EBSCOhost.

Modelos de bases de datos

Además de la clasificación por la función las bases de datos se pueden clasificar de acuerdo a su modelo de administración de datos.

Un modelo de datos es básicamente una "descripción" de algo conocido como contenedor de datos, así como de los métodos para almacenar y recuperar información de esos contenedores. Los modelos de datos son abstracciones que permiten la implementación de un sistema eficiente de base de datos; por lo general se refieren a algoritmos, y conceptos matemáticos.

Estos son algunos modelos:

Bases de datos jerárquicas
Los datos se organizan en forma de árbol invertido (algunos dicen raíz), en donde un nodo padre de información puede tener varios hijos. El nodo que no tiene padres es llamado raíz, y a los nodos que no tienen hijos se los conoce como hojas.
Las bases de datos jerárquicas son especialmente útiles en el caso de aplicaciones que manejan un gran volumen de información y datos muy compartidos permitiendo crear estructuras estables y de gran rendimiento.
Una de las principales limitaciones de este modelo es su incapacidad de representar eficientemente la redundancia de datos.

Base de datos de red
Se permite que un mismo nodo tenga varios nodos. Es una solución eficiente al problema de redundancia de datos; este modelo es utilizado en su mayoría por programadores más que por usuarios finales.

Bases de datos transaccionales
Son bases de datos cuyo único fin es el envío y recepción de datos a grandes velocidades, estas bases son muy poco comunes y están dirigidas por lo general al entorno de análisis de calidad, datos de producción e industrial, su fin único es recolectar y recuperar los datos a la mayor velocidad posible, por lo tanto la redundancia y duplicación de información no es un problema como con las demás bases de datos, por lo general para poderlas aprovechar al máximo permiten algún tipo de conectividad a bases de datos relacionales.
Un ejemplo habitual de transacción es el traspaso de una cantidad de dinero entre cuentas bancarias. Normalmente se realiza mediante dos operaciones distintas, una en la que se debita el saldo de la cuenta origen y otra en la que acreditamos el saldo de la cuenta destino. Para garantizar la atomicidad del sistema, las dos operaciones deben ser atómicas, es decir, el sistema debe garantizar que, bajo cualquier circunstancia, el resultado final es que, se han realizado las dos operaciones, o no se ha realizado ninguna,

Bases de datos relacionales
Éste es el modelo utilizado en la actualidad para representar problemas reales y administrar datos dinámicamente. Su idea fundamental es el uso de "relaciones". Estas relaciones podrían considerarse en forma lógica como conjuntos de datos llamados "tuplas". Pese a que ésta es la teoría de las bases de datos relacionales, la mayoría de las veces se conceptualiza de una manera más fácil de imaginar. Esto es pensando en cada relación como si fuese una tabla que está compuesta por registros, que representarían las tuplas, y campos.
En este modelo, el lugar y la forma en que se almacenen los datos no tienen relevancia. Esto tiene la considerable ventaja de que es más fácil de entender y de utilizar para un usuario esporádico de la base de datos. La información puede ser recuperada o almacenada mediante "consultas" que ofrecen una amplia flexibilidad y poder para administrar la información.
El lenguaje más habitual para construir las consultas a bases de datos relacionales es SQL, Structured Query Language o Lenguaje Estructurado de Consultas.
Durante su diseño, una base de datos relacional pasa por un proceso al que se le conoce como normalización de una base de datos.

Bases de datos multidimensionales
Son bases de datos ideadas para desarrollar aplicaciones muy concretas, como creación de Cubos OLAP. Básicamente no se diferencian demasiado de las bases de datos relacionales, la diferencia es a nivel conceptual; en las bases de datos multidimensionales los campos o atributos de una tabla pueden ser de dos tipos, o bien representan dimensiones de la tabla, o bien representan métricas que se desean aprender.

Bases de datos orientadas a objetos

Este modelo, bastante reciente, y propio de los modelos informáticos orientados a objetos, trata de almacenar en la base de datos los objetos completos (estado y comportamiento).
Una base de datos orientada a objetos es una base de datos que incorpora todos los conceptos importantes del paradigma de objetos:

- **Encapsulación** - Propiedad que permite ocultar la información al resto de los objetos, impidiendo así accesos incorrectos o conflictos.
- **Herencia** - Propiedad a través de la cual los objetos heredan comportamiento dentro de una jerarquía de clases.
- **Polimorfismo** - Propiedad de una operación mediante la cual puede ser aplicada a distintos tipos de objetos.

En bases de datos orientadas a objetos, los usuarios pueden definir operaciones sobre los datos como parte de la definición de la base de datos. Una operación (llamada función) se especifica en dos partes. La interfaz (o signatura) de una operación incluye el nombre de la operación y los tipos de datos de sus argumentos (o parámetros). La implementación (o método) de la operación se especifica separadamente y puede modificarse sin afectar la interfaz. Los programas de aplicación de los usuarios pueden operar sobre los datos invocando a dichas operaciones a través de sus nombres y argumentos, sea cual sea la forma en la que se han implementado. Esto podría denominarse independencia entre programas y operaciones.

Bases de datos documentales

Permiten la indexación a texto completo, y en líneas generales realizar búsquedas más potentes, sirven para almacenar grandes volumenes de informacion de antecedenes historicos. Tesaurus es un sistema de índices optimizado para este tipo de bases de datos.

Bases de datos deductivas
Es un sistema de base de datos pero con la diferencia de que permite hacer deducciones a través de inferencias. Se basa principalmente en reglas y hechos que son almacenados en la base de datos. Las bases de datos deductivas son también llamadas bases de datos lógicas, a raíz de que se basa en lógica matemática. Este tipo de base de datos surge debido a las limitaciones de la Base de Datos Relacional de responder a consultas recursivas y de deducir relaciones indirectas de los datos almacenados en la base de datos.

Lenguaje
Utiliza un subconjunto del lenguaje Prolog llamado Datalog el cual es declarativo y permite al ordenador hacer deducciones para contestar a consultas basándose en los hechos y reglas almacenados.

Ventajas

- Uso de reglas lógicas para expresar las consultas.
- Permite responder consultas recursivas.
- Cuenta con negaciones estratificadas
- Capacidad de obtener nueva información a través de la ya almacenada en la base de datos mediante inferencia.
- Uso de algoritmos que optimizan las consultas.
- Soporta objetos y conjuntos complejos.

Fases

- Fase de Interrogación: se encarga de buscar en la base de datos informaciones deducibles implícitas. Las reglas de esta fase se denominan reglas de derivación.

- Fase de Modificación: se encarga de añadir a la base de datos nuevas informaciones deducibles. Las reglas de esta fase se denominan reglas de generación.

3. Conceptos básicos

Sistema de Gestión de bases de datos distribuida (SGBD)

La base de datos y el software SGBD pueden estar distribuidos en múltiples sitios conectados por una red. Hay de dos tipos:

- **Distribuidos homogéneos**: Utilizan el mismo SGBD en múltiples sitios.
- **Distribuidos heterogéneos**: Dan lugar a los SGBD federados o sistemas multibase de datos en los que los SGBD participantes tienen cierto grado de autonomía local y tienen acceso a varias bases de datos autónomas preexistentes almacenados en los SGBD, muchos de estos emplean una arquitectura cliente-servidor.

Surgen debido a la existencia física de organismos descentralizados. Esto les da la capacidad de unir las bases de datos de cada localidad y acceder así a distintas universidades, sucursales de tiendas, etc .

Consulta a base de datos

Una consulta es el método para acceder a los datos. Con las consultas se puede modificar, borrar, mostrar y agregar datos en una base de datos, también pueden utilizarse como origen de registro para formularios. Para esto se utiliza un Lenguaje de consulta.
Las consultas a la base de datos se realizan a través de un Lenguaje de manipulación de datos, el lenguaje de consultas a base de datos más utilizado es SQL.

La base de datos relacional (BDR) es un tipo de base de datos que cumple con el modelo relacional y permite establecer interconexiones o relaciones entre los datos, y a través de dichas conexiones relacionar los datos entre las tablas. Las bases del modelo relacional fueron postuladas en 1970 por Edgar Frank Codd.

Características

- Una base de datos se compone de varias tablas o relaciones.
- No pueden existir dos tablas con el mismo nombre ni registro.
- Cada tabla es a su vez un conjunto de campos (columnas) y registros (filas).
- La relación entre una tabla padre y un hijo se lleva a cabo por medio de las claves primarias y claves foráneas (o ajenas).
- Las claves primarias son la clave principal de un registro dentro de una tabla y estas deben cumplir con la integridad de datos.
- Las claves ajenas se colocan en la tabla hija, contienen el mismo valor que la clave primaria del registro padre; por medio de estas se hacen las formas relacionales.

Relaciones
En una BDR, todos los datos se almacenan y se accede a ellos por medio de relaciones.

Relaciones base

Las relaciones que almacenan datos son llamadas relaciones base y su implementación es llamada "tabla".

Relaciones derivadas
Otras relaciones no almacenan datos, pero son calculadas al aplicar operaciones relacionales. Estas relaciones son llamadas relaciones derivadas y su implementación es llamada "vista" o "consulta". Las relaciones derivadas son convenientes ya que expresan información de varias relaciones actuando como si fuera una sola tabla.

Restricciones
Una restricción es una limitación que obliga el cumplimiento de ciertas condiciones en la BD.
Algunas no son determinadas por los usuarios, sino que son inherentemente definidas por el simple hecho de que la BD sea relacional. Algunas otras restricciones las puede definir el usuario, por ejemplo, usar un campo con valores enteros entre 1 y 10.

Las restricciones proveen un método de implementar "reglas" en la base de datos.
Las restricciones limitan los datos que pueden ser almacenados en las tablas.

Usualmente se definen usando expresiones que dan como resultado un valor booleano, indicando si los datos satisfacen la restricción o no.

Las restricciones no son parte formal del modelo relacional, pero son incluidas porque juegan el rol de organizar mejor los datos. Las restricciones son muy discutidas junto con los conceptos relacionales.

Dominios
Un dominio describe un conjunto de posibles valores para cierto atributo. Como un dominio restringe los valores del atributo, puede ser considerado como una restricción. Matemáticamente, atribuir un dominio a un atributo significa "cualquier valor de este atributo debe ser elemento del conjunto especificado".
Distintos tipos de dominios son: enteros, cadenas de texto, fecha, no procedurales, etc.
Cada tabla puede tener uno o más campos cuyos valores identifican de forma única cada registro de dicha tabla, es decir, no pueden existir dos o más registros diferentes cuyos valores en dichos campos sean idénticos. Este conjunto de campos se llama clave única. Pueden existir varias claves únicas en una determinada tabla, y a cada una de éstas suele llamársele candidata a clave primaria.

Clave primaria
Una clave primaria es una clave única (puede estar conformada por uno o más campos de la tabla) elegida entre todas las candidatas que define unívocamente a todos los demás atributos de la tabla para especificar los datos que serán relacionados con las demás tablas. La forma de hacer esto (relación entre tablas) es por medio de claves foráneas.

Clave foránea
Una clave foránea es una referencia a una clave en otra tabla, determina la relación existente en dos tablas. Las claves foráneas no necesitan ser claves únicas en la tabla donde están y sí a donde están referenciadas.
Por ejemplo, el código de departamento puede ser una clave foránea en la tabla de empleados. Se permite que haya varios empleados en un mismo departamento, pero habrá uno y sólo un departamento por cada clave distinta de departamento en la tabla de departamentos.

Clave índice

Las claves índices surgen con la necesidad de tener un acceso más rápido a los datos. Los índices pueden ser creados con cualquier combinación de campos de una tabla. Las consultas que filtran registros por medio de estos campos, pueden encontrar los registros de forma no secuencial usando la clave índice.

Las bases de datos relacionales incluyen múltiples técnicas de ordenamiento, cada una de ellas es óptima para cierta distribución de datos y tamaño de la relación.

Los índices generalmente no se consideran parte de la base de datos, pues son un detalle agregado. Sin embargo, las claves índices son desarrolladas por el mismo grupo de programadores que las otras partes de la base de datos.

Procedimientos almacenados
Un procedimiento almacenado es código ejecutable que se asocia y se almacena con la base de datos. Los procedimientos almacenados usualmente recogen y personalizan operaciones comunes, como insertar un registro dentro de una tabla, recopilar información estadística, o encapsular cálculos complejos. Son frecuentemente usados por un API por seguridad o simplicidad.
Los procedimientos almacenados no son parte del modelo relacional, pero todas las implementaciones comerciales los incluyen.
Estructura
La base de datos se organiza en dos marcadas secciones; el esquema y los datos (o instancia).
El esquema es la definición de la estructura de la base de datos y principalmente almacena los siguientes datos:

- El nombre de cada tabla
- El nombre de cada columna
- El tipo de dato de cada columna
- La tabla a la que pertenece cada columna

Las bases de datos relacionales pasan por un proceso al que se le conoce como normalización de una base de datos, el resultado de dicho proceso es un esquema que permite que la base de datos sea usada de manera óptima.

Los datos o instancia es el contenido de la base de datos en un momento dado. Es en sí, el contenido de todos los registros.

Integridad

La integridad es uno de las características más valiosas de los datos. Se refiere no solo a que los datos sean consistentes, sino además que los valores que posean dichos datos, sean válidos de acuerdo a las dependencias funcionales entre tablas y a las políticas del negocio.

Manipulación de la información

Para manipular la información utilizamos un lenguaje relacional, actualmente se cuenta con dos lenguajes formales el álgebra relacional y el cálculo relacional. El álgebra relacional permite describir la forma de realizar una consulta, en cambio, el cálculo relacional sólo indica lo que se desea devolver.
El lenguaje más común para construir las consultas a bases de datos relacionales es el **SQL (Structured Query Language)**, un estándar implementado por los principales motores o sistemas de gestión de bases de datos relacionales integradas.

En el modelo relacional los atributos deben estar explícitamente relacionados a un nombre en todas las operaciones, en cambio, el estándar SQL permite usar columnas sin nombre en conjuntos de resultados, como el asterisco taquigráfico (*) como notación de consultas.

Al contrario del modelo relacional, el estándar SQL requiere que las columnas tengan un orden definido, lo cual es fácil de implementar en una computadora, ya que la memoria es lineal.

En SQL el orden de las columnas y los registros devueltos en cierto conjunto de resultado nunca está garantizado, a no ser que explícitamente sea especificado por el usuario.

Gestores de base de datos relacionales
Existe un tipo de software dedicado a tratar con bases de datos relacionales, conocido como Sistema de Gestión de Bases de Datos Relacionales (SGBDR, o RDBMS del inglés Relational Database Management System), también llamados manejadores o gestores de las BDR.

Entre los gestores actuales más populares existen:

- MySQL.
- PostgreSQL.
- Oracle.
- DB2.
- Informix.
- Interbase.
- Firebird.
- Sybase.
- Microsoft SQL Server.

Ventajas y desventajas

Ventajas
- Provee herramientas que garantizan evitar la duplicidad de registros.
- Garantiza la integridad referencial, así, al eliminar un registro elimina todos los registros relacionados dependientes.
- Favorece la normalización por ser más comprensible y aplicable.

Desventajas
- Presentan deficiencias con datos gráficos, multimedia, CAD y sistemas de información geográfica.
- No se manipulan de forma manejable los bloques de texto como tipo de dato.
- Las bases de datos orientadas a objetos (BDOO) se propusieron con el objetivo de satisfacer las necesidades de las aplicaciones anteriores y así, complementar pero no sustituir a las bases de datos relacionales.

4. Modelo Relacional

Diseño de las bases de datos relacionales

El primer paso para crear una base de datos, es planificar el tipo de información que se quiere almacenar en la misma, teniendo en cuenta dos aspectos: la información disponible y la información que necesitamos.
La planificación de la estructura de la base de datos, en particular de las tablas, es vital para la gestión efectiva de la misma. El diseño de la estructura de una tabla consiste en una descripción

de cada uno de los campos que componen el registro y los valores o datos que contendrá cada uno de esos campos.
El principal aspecto durante el diseño de una tabla es determinar claramente los campos necesarios, definirlos en forma adecuada con un nombre especificando su tipo y su longitud.

La interfaz estándar de programa de usuario y aplicación a una base de datos relacional es el lenguaje de consultas estructuradas (SQL). Los comandos de SQL se utilizan tanto para consultas interactivas para obtener información de una base de datos relacional y para la recopilación de datos para los informes.
Además de ser relativamente fáciles de crear y acceder, una base de datos relacional tiene la importante ventaja de ser fácil de extender. Después de la creación original de una base de datos, una nueva categoría de datos se puede añadir sin necesidad de que todas las aplicaciones existentes sean modificadas.
La definición de una base de datos relacional resulta en una tabla de metadatos o descripciones formales de las tablas, columnas, dominios y restricciones.

El modelo relacional sigue un criterio basado en las formas normales, las cuales eliminan la redundancia de datos y las estructuras de datos representados por las relaciones pueden ser manipuladas por cualquier sistema manejador de bases de datos.
El modelo Entidad-Relación tiene como objetivo identificar y representar de manera conceptual la información de importancia para el funcionamiento de entidades, sus propiedades, y sus relaciones.

La bases de datos relacionales pasan por un proceso de normalización como se menciono antes. Como el objetivo principal del modelo es evitar la redundancia, se deben satisfacer relaciones definidas en términos de relaciones normales, ya que estas deben estar en la forma mas alta posible de normalidad y cada forma normal elimina algún tipo de redunda.

Existen cinco formas normales. Las primeras tres se refieren a redundancias relativas a dependencias funcionales, mientras que las otras eliminan redundancias relativas a dependencias multi-valuadas.

Primera Forma Normal

Una relación esta en primera forma normal si todo atributo contiene un valor atómico para cada uno de sus elementos, es decir una estructura en forma de tabla solo puede tener un solo valor en cada intersección de cada renglón y columna.

También se dice que una tabla está en primera forma normal si y solo si: Contiene una llave primaria, no contiene atributos nulos, no posee ciclos repetitivos y contiene un solo valor por celda.

Segunda Forma Normal
Una relación esta en segunda forma normal si y solo si: La relación esta en primera forma normal y si todo atributo que no sea llave es completamente dependiente de manera funcional de la llave completa. Todo atributo que no es llave necesita en forma completa la llave para poder ser identificado de manera única. La segunda forma normal permite eliminar las redundancias para que ningún atributo este determinado por una parte de una llave.

Tercera Forma Normal
Una relación esta en tercera forma normal si y solo si: La relación está en su segunda forma normal y si además todo atributo que no pertenece la llave no depende de otro atributo que no es

llave. Cuando un atributo que no es llave, se puede determinar con uno o demás atributos que también son llave, se dice que existe dependencia transitiva entre ambos. La tercera forma normal permite eliminar la redundancia transitiva.

Forma Normal Boyce-Codd

Una relación esta en forma normal Boyce-Cood si y solo si: La relación está en tercera forma normal y si además cada atributo del esquema de relación que determine otros atributos esta en una clave candidata.

Cuarta Forma Normal

Una relación esta en cuarta forma normal si y solo si: La relación está en forma normal Boyce-Cood y si además todas sus dependencias no triviales son dependencias funcionales (de valores simples). Esto es que una relación en cuarta forma normal no puede tener ninguna dependencia de valores múltiples no trivial. Básicamente cada dependencia requiere una tabla separada.

Quinta Forma Normal

Una relación esta en cuarta forma normal si y solo si: para cada dependencia de combinación no trivial, cada proyección incluye una clave de la tabla original

La quinta forma normal trata casos donde la información puede ser reconstruida de muchas piezas de información las cuales pueden ser mantenidas con poca redundancia.

Cálculo Relacional

El cálculo relacional es un lenguaje de consulta que describe la respuesta deseada sobre una base de datos sin especificar como obtenerla, y es de tipo declarativo:

Podemos definir una formula con base en combinaciones de formulas
Una fórmula es una combinación de variables (tupla o dominio) y atributos o constantes, gracias al uso de operadores como < ,>,=,!=,<=,>=
También es una formula variable ∈ Relación.
Las combinaciones de fórmulas se generan a partir del uso de operadores como NOT (¬), AND (^), ORD (v).
Los cuantificadores ∋ ∀ limitan una variable.

Algebra Relacional

Es un conjunto de operaciones simples sobre las tablas relacionales, a partir de las cuales se definen operaciones más complejas mediante composición. Las operaciones básicas son de dos tipos:
Binarias y unarias.

Binarias

- **Unión.** La unión de dos relaciones R y S con el mismo esquema, es una relación T con el conjunto de tuplas que pertenecen a R y S

- **Intersección.** La intersección de dos relaciones R y S con el mismo esquema, es una relación T y contiene las tuplas que pertenecen a R y a S a la vez.

- **Diferencia. .** La diferencia de dos relaciones R y S con el mismo esquema, es una relación T y contiene las tuplas que pertenecen a R y no a S.

- **Producto cartesiano.** El producto de dos relaciones R y S con el mismo esquema, es una relación T que contiene atributos de R concatenados con los de S y sus tuplas son todas las formadas por la concatenación de una tupla de R con todas las tuplas de S.

- **Producto o conjunción.(Join).** El producto de dos relaciones R y s según Q son el conjunto de tuplas del producto cartesiano RXS que satisfacen a Q.

Unarias

- **Proyección.** La proyección de una relación R (A1,A2,A3,A4,A5,…An) es una relación R obtenida por la eliminación de los valores de los atributos de R que no están contenidas en R" y la supresión de las tuplas duplicadas, es la eliminación de columnas en una tabla.

- **Selección.** La restricción de una relación R por un criterio de selección Q es una relación R" con el mismo esquema de R cuyas tuplas son aquellas que pertenecen a R y satisfacen a Q.

Cardinalidad.

Muestra el número de relaciones en las cuales una entidad puede aparecer. Hay cuatro tipos de cardinalidad.

- **Una a Una.** Una y solo una entidad de A esta asociada a una entidad de B y viceversa

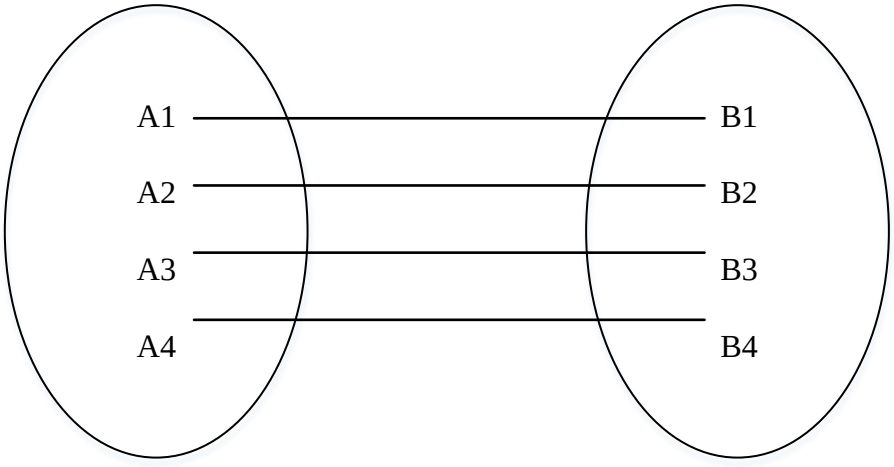
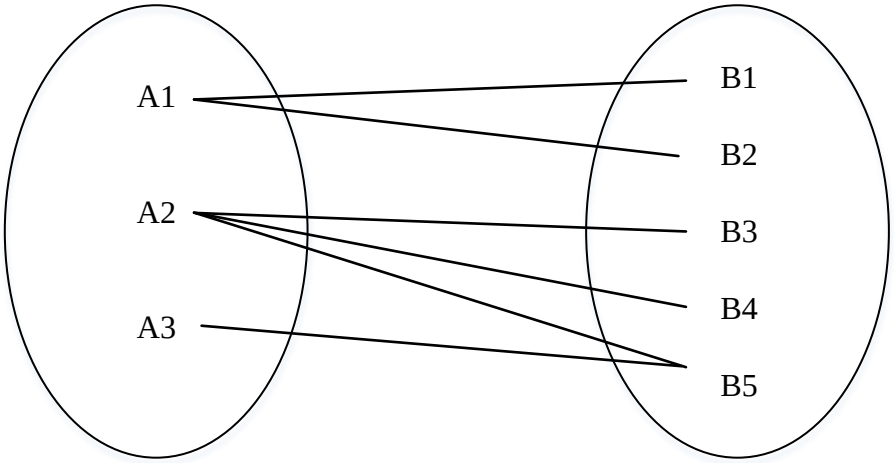


Figura 2.4.1 Cardinalidad uno a uno

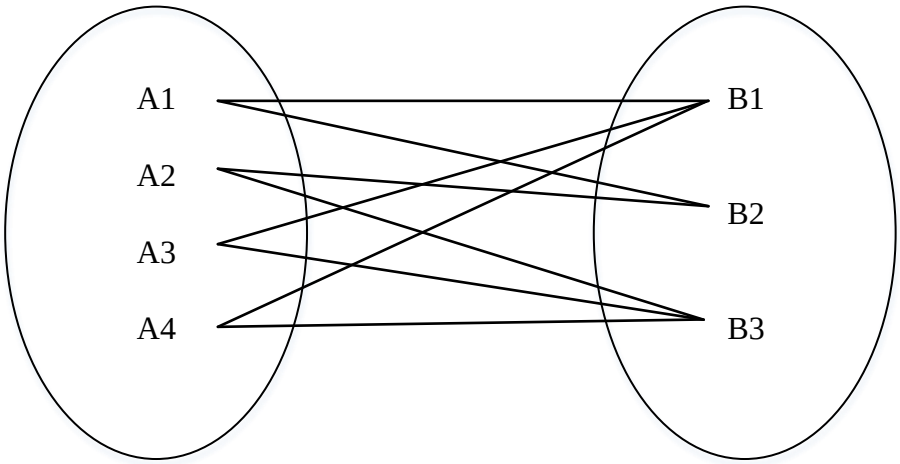
- **Uno a Muchos**. Una Entidad A esta asociada a una o varias entidades en B, en cambio una entidad en B solo puede estar asociada con una solo entidad de A.



1:N

Figura 2.4.2 Cardinalidad uno a Muchos

- **Muchos a Uno**. Una entidad de A se Asocia únicamente con una entidad de B, sin embargo una entidad de B se puede asociar con cualquier número de entidades de A



N:1

Figura 2.4.3 Cardinalidad Muchos a uno

- **Muchos a Muchos.** Una entidad de A esta asociada a cualquier cantidad de entidades en B y una entidad en B está asociada a cualquier cantidad de entidades en A

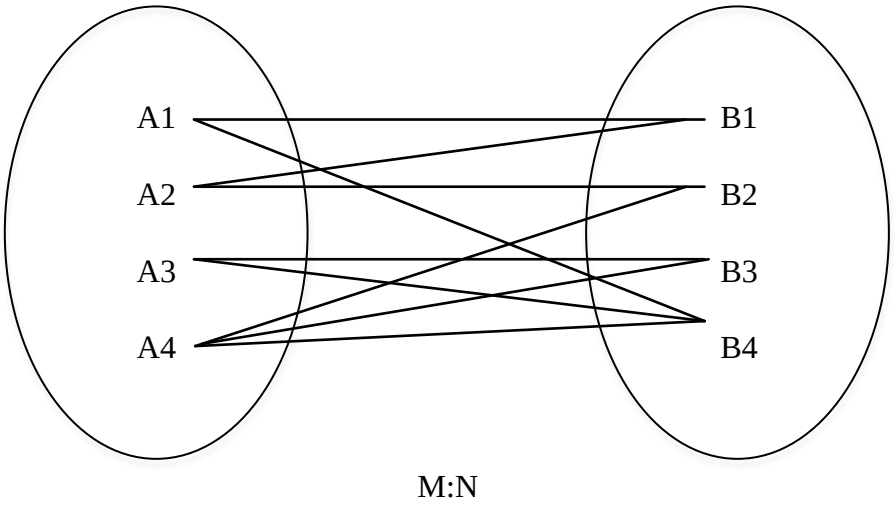


Figura 2.4.4 Cardinalidad muchos a muchos

5. Ms SQL Server

Microsoft SQL Server es un sistema de manejo de bases de datos del modelo relacional, desarrollado por la empresa Microsoft.

El lenguaje de desarrollo utilizado (por línea de comandos o mediante la interfaz gráfica de Management Studio) es Transact-SQL (TSQL), una implementación del estándar ANSI del lenguaje SQL, utilizado para manipular y recuperar datos (DML), crear tablas y definir relaciones entre ellas (DDL).

Dentro de los competidores más destacados de SQL Server están: Oracle, MariaDB, MySQL, PostgreSQL. SQL Server solo está disponible para sistemas operativos Windows de Microsoft.

Algunas de las versiones más recientes:

Versión	Año	Nombre de la versión	Nombre clave
10.25	2010	SQL Azure DB	CloudDatabase
10.50	2010	SQL Server 2008 R2	Kilimanjaro
11.0	2012	SQL Server 2012	Denali
12.0	2014	SQL Server 2014	SQL14 (antes Hekaton)

Figura 2.5.1 Versiones SQL server

Características
Entre las características más destacadas se tienen:

- Soporte de transacciones.
- Soporta procedimientos almacenados.
- Incluye también un entorno gráfico de administración, que permite el uso de comandos DDL y DML gráficamente.
- Permite trabajar en modo cliente-servidor, donde la información y datos se alojan en el servidor y los terminales o clientes de la red sólo acceden a la información.
- Además permite administrar información de otros servidores de datos.

Este sistema incluye una versión reducida, llamada MSDE con el mismo motor de base de datos pero orientado a proyectos más pequeños, que en sus versiones 2005 y 2008 pasa a ser el SQL Express Edition, que se distribuye en forma gratuita.

Es común desarrollar proyectos completos empleando Microsoft SQL Server y Microsoft Access a través de los llamados ADP (Access Data Project). De esta forma se completa la base de datos (Microsoft SQL Server), con el entorno de desarrollo (VBA Access), a través de la implementación de aplicaciones de dos capas mediante el uso de formularios Windows.
En el manejo de SQL mediante líneas de comando se utiliza el SQLCMD, osql, o PowerShell.
Para el desarrollo de aplicaciones más complejas (tres o más capas), Microsoft SQL Server incluye interfaces de acceso para varias plataformas de desarrollo, entre ellas .NET, pero el servidor sólo está disponible para sistemas operativos de windows.

Programación

T-SQL

T-SQL (Transact-SQL) es el principal medio de interacción con el Servidor, el cual permite realizar las operaciones claves en SQL Server, incluyendo la creación y modificación de

esquemas de base de datos, inserción y modificación de datos en la base de datos, así como la administración del servidor como tal. Esto se realiza mediante el envío de sentencias en T-SQL y declaraciones que son procesadas por el servidor y los resultados (o errores) regresan a la aplicación cliente.

Cliente Nativo de SQL

El Cliente Nativo de SQL, es la biblioteca de acceso a datos para los clientes de Microsoft SQL Server. Implementa de forma nativa soporte para las características de SQL Server, incluyendo la ejecución de la secuencia de datos tabular, soporte para bases de datos en espejo de SQL Server, soporte completo para todos los tipos de datos compatibles con SQL Server, conjuntos de operaciones asíncronas, las notificaciones de consulta, soporte para cifrado, así como recibir varios conjuntos de resultados en una sola sesión de base de datos. Cliente Nativo de SQL se utiliza como extensión de SQL Server plug-ins para otras tecnologías de acceso de datos, incluyendo ADO u OLE DB. El Cliente Nativo de SQL puede también usarse directamente, pasando por alto las capas de acceso de datos.

Interfaz de usuario

SQL Server proporciona unos interfaz que han cambiado durante los años, de los cuales los más conocidos son los interfaz gráficos que están utilizados como herramienta de desarrollo estándar a los desarrolladores y administradores.
El interfaz gráfico es llamado SQL Server Management Studio (SSMS), con la opción de trabajar con el Visual Studio– el interfaz estándar de desarrollo de Microsoft (a los distintos lenguajes, BI, etc.). Otro interfaz opcional es la utilización de Línea de comandos, con herramientas como SQLCmd, ISQL, OSQL que posibilita la ejecución de scripts y procesamiento por lotes. Desde 2008 se puede desarrollar con SQLCmd (SQL Command) a través del SSMS sin interconectarse al interfaz textual de Windows. Otra opción en el ámbito de scripts es la utilización del lenguaje de scripts Powershell de Microsoft.
Aparte de los interfaces estándares de SQL Server, se puede ejecutar comandos de TSQL con herramientas de conexión como ODBC y OLE-DB.10
Servicios
En SQL Server hay un número de servicios, software que se ejecutadan en la memoria del servidor por parte del sistema, y por lo tanto aprovechan las capacidades del servidor que es más potente que los clientes, previenen congestión en la red, y pueden programar tareas que corran aún cuando el cliente no esté conectado.

Los servicios principales:

- SQL Server - El "motor" del sistema
- SQL Agent - Ejecución de tareas (Jobs, scripts programados) y envió de advertencias en caso de carga pesada e irregulares en el sistema
- Full-Text Filter Daemon Launcher - La utilización en los indexes especiales del "Full text search" por búsqueda textual avanzada
- SQL Browser - El "oyente" dedicado a comandos enviados y re dirigirlos a su destino
- SSIS Server - La operación del SSIS (la herramienta de ETL)
- SSAS Server - La operación del SSAS (la herramienta de OLAP)
- SSRS Server - La operación del SSRS (la herramienta de informes)

Capacidades y herramientas básicas

Bases de datos

En cada instalación de SQL Server hay cuatro bases de datos del sistema, y la capacidad de crear nuevas bases de datos por el usuario, en los cuales los datos están almacenados en tablas.
Estas bases de datos, creadas por parte de los usuarios, incluyen básicamente un archivo de datos (con el sufijo mdf) con las tablas y los distintos objetos a nivel de la base de datos; y un archivo de registro (con el sufijo ldf) con las transacciones abiertas, y transacciones cerradas, Sujeto al modelo de recuperación seleccionado (se puede acumular en el archivo de registro todos los cambios en la base de datos desde el último respaldo). Se puede crear un conjunto de archivos de datos además del principal (con el sufijo ndf) por consideraciones de eficiencia, partición de carga de trabajo entre los discos rígidos, etc.

Las bases de datos del sistema:

- **master** - Todos los procedimientos, funciones y tablas del sistema que están utilizadas por todas las bases de datos y que están instaladas automáticamente, tanto como las que han sido creado por parte de los administradores del sistema. Además, de todas las definiciones de seguridad a nivel del servidor, están almacenadas en esta base de datos.
- **msdb** - Almacena las tareas del agente, los códigos de CLR combinados en el sistema, los paquetes de SSIS, y otros más.
- **model** - El molde de las bases de datos. Cada nueva base de datos se crea como una copia de esta base de datos.
- **tempdb** - Base de datos temporal que se crea de nuevo cada vez que el servicio reinicia. Se utiliza para almacenar tablas temporales creadas por parte de los usuarios o el sistema (por ejemplo en ordenaciones complejos).

Tablas fijas y temporales

Desde la perspectiva lógica, los datos almacenados en las bases de datos en tablas, mediante ellas se implementa la teoría de las bases de datos relacionales. La tabla se divide en filas y columnas (A veces se les conoce como registros y campos). Las tablas pueden ser fijas o temporales, mientras que en el segundo caso existen físicamente en la base de datos tempdb

Desde la perspectiva física, el sistema divide los archivos de la base datos en Extents de 64 KB, y cada cual en ocho páginas de 8 KB. Generalmente, Cada Extent se asigna a una tabla o un índice, menos las tablas pequeñas; y cada página se asigna siempre a una tabla específica. El sistema es responsable del aumento de los archivos, de acuerdo con los ajustes del usuario, y de asignar Extents y páginas a las tablas.

A las tablas se puede crear índices. Los índices se almacenan junto a la tabla (Non Clustered Index) o son la tabla en sí (Clustered Index). Los índices asisten en la búsqueda de datos en las tablas (como los ficheros en las librerías), en ordenarlas, y la definición de claves primarias.

Entre las tablas se puede crear una relación de uno a muchos.
Aparte de las tablas de los usuarios, hay tablas que almacenan meta data: datos sobre el sistema mismo, los diferentes objetos, los derechos, estadísticas sobre el rendimiento del sistema (DMV), etc.

Tipos de datos

Para cada columna en una tabla y a cada variable o parámetro, se define un tipo de datos que seran almacenados en él, entre ellos:

1. Números: Números enteros y no enteros en distintos tamaños, y en diferentes niveles de precisión; y auto incremento opcional.
2. Textos: Cadenas de distintas longitudes, y distintas capacidades para soportar distintas lenguas.
3. Fechas: Fechas en distintos niveles de precisión, desde días completos hasta fracciones menores de un segundo, que soportan fechas a partir del principio del siglo 20 o del calendario gregoriano, y la capacidad de diferenciar entre distintos usos horarios.
4. XML: Datos textuales (cadenas) que representan conjuntos estándares de datos (estándar SGML).
5. Datos binarios: Datos almacenados como datos binarios (bits y bytes), que posibilitan el almacenamiento de archivos gráficos, etc.
6. Geography: Representación estándar de información geográfica, tales como estados, zonas geográficas, localidades; y las cálculos como distancias.
7. Geometry: Representación estándar de puntas, líneas, superficies en el plano; y las relaciones entre ellas.
8. Hierarchid: Representación estándar de información jerárquica como lista de materiales, relaciones de subordinación entre empleados, etc.
9. Vistas

Las vistas representan generalmente comandos de extracción de datos, que se almacenan sin los datos (que están almacenados en las tablas). Esta opción nos posibilita crear extracciones complejas o estándares, almacenarlas como vistas, y utilizar las vistas sin la necesidad de escribir de nuevo los comandos o mantener los códigos donde ellas aparecen. Adicionalmente, es un medio muy importante para otorgar derechos selectivos de lectura (en caso que queremos posibilitar a un usuario contemplar parcialmente las columnas o las filas de una tabla).

Procedimientos almacenados
Los procedimientos son scripts de comandos de TSQL, que pueden ser ejecutados con distintos parámetros. Por ejemplo, procedimiento que obtiene número de año como parámetro, y actualiza una tabla de resumen de ventas, con las ventas de los agentes en el dicho año, basada en la tabla de registro de ventas.

Funciones definidas por el usuario
Las funciones son un objeto que combina algunas capacidades de las vistas, con otras de los procedimientos. Como las vistas, pueden extraer datos y ejecutar cálculos, y devuelven un resultado al usuario o al programa que les ejecuto. Al igual que los procedimientos, incluyen códigos de TSQL, y pueden ser ejecutados con parámetros.
Las funciones devuelven un valor o un conjunto de valores.

Transacciones
Una transacción es un conjunto de comandos, que se está ejecutado completamente o no ejecutado en absoluto: todo o nada. Por ejemplo, si una suma de dinero fue trasladada de una cuenta bancaria a otra, y hay que actualizar ambas cuentas sobre el depósito y la retirada; es obligatorio que ambas cuentas se actualizan juntas, o ninguna (en caso que una de las actualizaciones falla); para evitar consecuencias inconsistentes de un depósito sin ninguna retirada, o vice versa.

SQL Server tiene una capacidad limitada de anidar transacciones.

El Optimizador
El optimizador es una parte del software que "toma la decisión" de cómo cada comando se ejecutará, para que la ejecución sea lo más eficiente, o por lo menos bastante eficiente (es decir,

bastante eficiente para evitar seguir buscando otra solución, que aún que sea más eficiente, el precio de la búsqueda adicional "costará" más que el ahorro de recursos).

SQL es un lenguaje declarativo, en el cual el desarrollador declara que quiere extraer o actualizar sin la necesidad de indicar cómo (a contrario de los lenguajes imperativos, y por lo tanto el optimizador juega un papel protagónico, que de acuerdo con las estadísticas que el sistema almacena sobre las distribuciones de los datos en las tablas, los indexes, y reglas internas; toma la decisión adecuada.

Privilegios y seguridad de datos

Para conectarse al SQL Server, se necesita un Login (usuario a nivel del servidor). Cuando la política de seguridad se define como Windows Authentication y el servidor se combina con las definiciones del Domain, los Logins se definen en el Active Directory. Cuando la definición es SQL Server Authentication los logins (usuario y contraseña) se definen en el SQL Server mismo. Consecuentemente, en el primer caso hay que identificarse con nombre y contraseña solamente al conectarse a la red, y luego se conecta automáticamente a todos los servidores que son Windows Authentication (con el Login global); y en el segundo caso hay que identificarse al conectarse a cada servidor de SQL Server Authentication (cada vez con un Login local).

A nivel de la base de datos, el usuario se identifica como un User que está relacionado generalmente al Login (que es a nivel del servidor), y los privilegios al User existen solamente en el ámbito de la base de datos (además a los privilegios al Login). Para otorgar derechos generales puede asistirse con listas de Server Roles (roles a nivel del servidor) o Database Roles (roles a nivel de la base de datos específica), cada cual con privilegios específicos a un rol específico; y cada usuario asociado con uno de estos Roles obtiene los privilegios asociados con él. Además, el administrador puede otorgar derechos específicos, y crear otros Database Roles (no se puede crear Server Roles).
Los privilegios a nivel del servidor incluyen la capacidad de crear bases de datos, utilizar las tareas (Jobs), crear respaldos de bases de datos y restaurarlos, modificar las definiciones del servidor, etc. Los privilegios a nivel de la base de datos posibilitan extraer y actualizar datos, crear objetos como procedimientos y tablas, utilizar dichos objetos, etc. Como regla general se puede otorgar derechos (Grant), revocar privilegios existentes (Revoke), y denegar privilegios aún no existen (Deny).

Respaldos y recuperaciones

Aparte de soluciones de alternativas a nivel del sistema operativo (respaldo de los archivos de la base de datos), hay una herramienta integrada en el SQL Server que posibilita un respaldo completo o diferencial, de acuerdo con el modelo de recuperación (Recovery Model) predefinido a la base de datos; y una recuperación completa o a un punto de tiempo. Aparte de un respaldo de la base de datos se puede respaldar los datos a través de un guion (con o sin los datos) y comprimir los archivos de respaldo.

El agente y la programación de tareas

El agente es el servicio encargado de la programación de tareas, y se encarga de ejecutarlas independientemente. Generalmente el ejecuta tareas de mantenimiento, tareas complejas de ETL, respaldos, etc.

Mantenimiento

Con el fin de mejorar el rendimiento del sistema hay que mantener las estadísticas, utilizadas por el optimizador, organizar los archivos físicos, etc; y se utilizan herramientas dedicadas para estos propósitos, que se ejecutan periódicamente por tareas programadas, y de una manera coordinada con las tareas de ETL y de respaldo.

Service Broker
Una tecnología que implementa arquitectura orientada a servicios, y que posibilita ejecuciones asíncronos: primero que nada para enviar mensajes entre distintas aplicaciones que se ejecutan simultáneamente, pero también para ejecutar procedimientos asíncronamente, en la manera de dispara y olvida un procedimiento que se ejecuta en una sesión diferente de la sesión que lo inicio, y ambos procedimientos siguen ejecutado independientemente uno del otro.

Búsqueda de Textos completos
Una herramienta que posibilita indexar columnas textuales como textos y no solo como cadenas; y ejecutar búsquedas complejas dependientes en el sentido del texto y en el idioma. Por ejemplo, buscamos un verbo, y queremos obtener todas las ocurrencias de sus conjugaciones.

Rastrear
Estas herramientas incluyen el seguimiento que posibilita rastrear actividades con el fin de mantener cargas y fallos, y seguridad de datos (recuperación no permitida de datos), el Profiler que posibilita rastrear los comandos que se ejecutan y los eventos que se ocurren en el servidor, y el Extended Events y cambia el profiler gracias a su baja consumo de recursos y la influencia sobre el rendimiento del servidor.
Aparte de estos, se puede utilizar dos tipos de Trigers (disparadores) para rastrear los cambios y las actividades: DML Trigers pre definidos sobre las tablas y las vistas y que se inician por instrucciones de actualización de datos (Select / Update / Delete), y DDL Triggers que se inician por cambios en los objetos mismos (y no en los datos), en el nivel de la base de datos o del servidor.

Combinación de CLR
A partir de 2005 se puede combinar fácilmente en SQL Server procedimientos, funciones, y funciones de agregado desarrolladas en CLR. Hay que desarrollar el código en una de las herramientas de desarrollo de .NET, crear un archivo DLL, y combinarlo en el sistema. La ventaja de esta tecnología es sus capacidades en problemas que no son exclusivamente de bases de datos (manipulación de datos), e incluyen cálculos complejos o manipulaciones textuales de cadenas.

Herramientas de Inteligencia empresarial
Una instalación típica incluye también las herramientas de BI (Inteligencia empresarial):

SSIS (SQL Server Integration Services)
Una herramienta de ETL que posibilita la extracción de datos de distintos orígenes (no solo SQL Server), la transformación de dichos datos, y la carga (generalmente pero no obligatoriamente a almacén de datos).

SSAS (SQL Server Analysis Services)
Una herramienta para crear Bases de Datos Multidimensionales (no relacionales), que se puede explorar mediante extracciones de datos en distintos niveles de agrupación, profundización (Drill Down) de una suma a sus detalles, y utilización de MDX (un lenguaje parecido a SQL, adaptado a bases de datos multidimensionales).

SSRS (SQL Server Reporting Services)
Una herramienta para crear y dar formato a informes, otorgar derechos de contemplación en ellos, y su distribución. Se puede contemplarlos con un Navegador web, y se puede exportarlos a archivos de Excel, PDF, etc. los datos se extraen generalmente del almacén de datos o del OLAP.

Desventajas

En versiones de 32 bits, SQL Server usa Address Windowing Extension (AWE) para hacer el direccionamiento por encima de 4 GB. Esto le impide usar la administración dinámica de memoria, y sólo le permite alojar un máximo de 64 GB de memoria compartida. Esta limitación es exclusiva de sistemas operativos 32 bits; en sistemas operativos 64 bits, la memoria máxima que se puede direccionar en Edición Estándar es 64 Gb y en Edición Enterprise 4Tb

6. Características Generales del Lenguaje SQL

SQL (por sus siglas en inglés Structured Query Language) es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en ellas. Una de sus características es el manejo del álgebra y el cálculo relacional que permiten efectuar consultas con el fin de recuperar, de forma sencilla, información de bases de datos, así como hacer cambios en ellas.

T-SQL (Transact-SQL) es el principal medio de interacción con el Servidor, el cual permite realizar las operaciones claves en SQL Server, además Cliente Nativo de SQL, es la biblioteca de acceso a datos para los clientes de Microsoft SQL Server

El ANSI SQL sufrió varias revisiones y agregados a lo largo del tiempo:

Año	Nombre	Alias	Comentarios
1986	SQL-86	SQL-87	Primera publicación ANSI. Confirmada por ISO en 1987.
1989	SQL-89	nn	Revisión menor.
1992	SQL-92	SQL2	Revisión mayor.
1999	SQL:1999	SQL2000	Se agregaron expresiones regulares, consultas recursivas (para relaciones jerárquicas), triggers y algunas características orientadas a objetos.
2003	SQL:2003		Introduce algunas características de XML, cambios en las funciones, estandarización del objeto sequence y de las columnas autonuméricas.
2005	SQL:2005		ISO/IEC 9075-14:2005 Define las maneras en las cuales SQL se puede utilizar conjuntamente con XML. Define maneras de importar y guardar datos XML en una base de datos SQL, manipulándolos dentro de la base de datos y publicando el XML y los datos SQL convencionales en forma XML. Además, proporciona facilidades que permiten a las aplicaciones integrar dentro de su código SQL el uso de XQuery, lenguaje de consulta XML publicado por el W3C (World Wide Web Consortium) para acceso concurrente a datos ordinarios SQL y documentos XML
2008	SQL:2008		Permite el uso de la cláusula ORDER BY fuera de las definiciones de los cursores. Incluye los disparadores del tipo INSTEAD OF. Añade la sentencia TRUNCATE.

Figura 2.6.1 Revisiones ANSI SQL

SQL es un lenguaje de acceso a bases de datos que explota la flexibilidad y potencia de los sistemas relacionales y permite así gran variedad de operaciones.

Es un lenguaje declarativo de "alto nivel" o "de no procedimiento" que, gracias a su fuerte base teórica y su orientación al manejo de conjuntos de registros que permite una alta productividad en codificación y la orientación a objetos.

SQL también tiene las siguientes características:

- Lenguaje de definición de datos: El LDD de SQL proporciona comandos para la definición de esquemas de relación, borrado de relaciones y modificaciones de los esquemas de relación.
- Lenguaje interactivo de manipulación de datos: El LMD de SQL incluye lenguajes de consultas basado tanto en álgebra relacional como en cálculo relacional de tuplas.
- Integridad: El LDD de SQL incluye comandos para especificar las restricciones de integridad que deben cumplir los datos almacenados en la base de datos.
- Definición de vistas: El LDD incluye comandos para definir vistas.
- Control de transacciones: SQL tiene comandos para especificar el comienzo y el final de una transacción.
- SQL incorporado y dinámico: Esto quiere decir que se pueden incorporar instrucciones de SQL en lenguajes de programación como: C++, C, Java, PHP, Cobol, Pascal y Fortran.
- Autorización: El LDD incluye comandos para especificar los derechos de acceso a las relaciones y a las vistas.

Tipos de Datos

Algunos de los tipos de datos básicos de SQL son:

Tipo	Descripción
CHARACTER(n)	Cadena de caracteres. De largo n
VARCHAR(n) or CHARACTER VARYING(n)	Cadena de caracteres. Largo Variable. Maxima longitud n
BINARY(n)	Cadena Binaria. De largo n
BOOLEAN	Booleano TRUE o FALSE
VARBINARY(n) or BINARY VARYING(n)	Cadena Binaria. De largo variable o Maxima longitud de largo n
INTEGER(p)	Entero (no decimal). Precisión p
SMALLINT	Entero (no decimal). Precisión 5
INTEGER	Entero (no decimal). Precisión 10
BIGINT	Entero (no decimal). Precisión 19
DECIMAL(p,s)	Numero Decimal, precisión p, con dígitos decimales s.
NUMERIC(p,s)	Número Exacto, precisión p, con dígitos decimales s
FLOAT(p)	Numérico aproximado, mantisa precisión p. en base 10 notación exponencial
REAL	Numérico aproximado, mantisa precisión 7
FLOAT	Numérico aproximado, mantisa precisión 16
DOUBLE PRECISION	Numérico aproximado, mantisa precisión 16
DATE	Almacena año, mes, día
TIME	Almacena hora, minutos, segundos
TIMESTAMP	Almacena año, mes, día, hora, minutos, segundos
INTERVAL	Compuesto de campos enteros que representan un periodo de tiempo
ARRAY	Colección de elementos ordenados
MULTISET	Colección de elementos no ordenados
XML	Almacena datos XML

Figura 2.6.2 Tipos de datos SQL

Optimización

SQL es un lenguaje declarativo. O sea, que especifica qué es lo que se quiere y no cómo conseguirlo, por lo que una sentencia no establece explícitamente un orden de ejecución.
El orden de ejecución interno de una sentencia puede afectar seriamente a la eficiencia del SGBD, por lo que se hace necesario que éste lleve a cabo una optimización antes de su ejecución. Muchas veces, el uso de índices acelera una instrucción de consulta, pero alenta la actualización de los datos. Dependiendo del uso de la aplicación, se priorizará el acceso indexado o una rápida actualización de la información. La optimización difiere sensiblemente en cada motor de base de datos y depende de muchos factores.

Existe una ampliación de SQL conocida como FSQL (Fuzzy SQL, SQL difuso) que permite el acceso a bases de datos difusas, usando la lógica difusa. Este lenguaje ha sido implementado a nivel experimental y está evolucionando rápidamente.

Lenguaje de definición de datos (DDL)

El lenguaje de definición de datos (en inglés Data Definition Language, o DDL), es el que se encarga de la modificación de la estructura de los objetos de la base de datos. Incluye órdenes para modificar, borrar o definir las tablas en las que se almacenan los datos de la base de datos.

Existen cuatro operaciones básicas: CREATE, ALTER, DROP y TRUNCATE.

CREATE
Este comando permite crear objetos de datos, como nuevas bases de datos, tablas, vistas y procedimientos almacenados.
Ejemplo (crear una tabla)

```
CREATE TABLE 'CUSTOMERS';
```

ALTER
Este comando permite modificar la estructura de una tabla u objeto. Se pueden agregar o quitar campos a una tabla, modificar el tipo de un campo, agregar o quitar índices a una tabla, modificar un trigger, etc.
Ejemplo (agregar columna a una tabla)

```
    ALTER TABLE 'ALUMNOS' ADD EDAD INT UNSIGNED;
```

DROP
Este comando elimina un objeto de la base de datos. Puede ser una tabla, vista, índice, trigger, función, procedimiento o cualquier objeto que el motor de la base de datos soporte. Se puede combinar con la sentencia ALTER.
Ejemplo

```
    DROP TABLE 'ALUMNOS';
```

TRUNCATE
Este comando trunca todo el contenido de una tabla. La ventaja sobre el comando DROP, es que si se quiere borrar todo el contenido de la tabla, es mucho más rápido, especialmente si la tabla es muy grande. La desventaja es que TRUNCATE sólo sirve cuando se quiere eliminar absolutamente todos los registros, ya que no se permite la cláusula WHERE. Si bien, en un principio, esta sentencia parecería ser DML (Lenguaje de Manipulación de Datos), es en realidad una DDL, ya que internamente, el comando TRUNCATE borra la tabla y la vuelve a crear y no ejecuta ninguna transacción.
Ejemplo

TRUNCATE TABLE 'NOMBRE_TABLA';

Lenguaje de manipulación de datos DML(Data Manipulation Language)

Un lenguaje de manipulación de datos (Data Manipulation Language, o DML en inglés) es un lenguaje proporcionado por el sistema de gestión de base de datos que permite a los usuarios llevar a cabo las tareas de consulta o manipulación de los datos, organizados por el modelo de datos adecuado.

El lenguaje de manipulación de datos más popular hoy día es SQL, usado para recuperar y manipular datos en una base de datos relacional.

Algunos de los más importantes comandos en SQL.

Comandos	Descripción
SELECT	Muestra datos de una base de datos
UPDATE	Actualiza datos de una base de datos
DELETE	Borra datos de una base de datos
INSERT INTO	Inserta nuevos datos en una base de datos
CREATE DATABASE	Crea una base de datos
ALTER DATABASE	Modifica a base de datos
CREATE TABLE	Crea una nueva tabla
ALTER TABLE	Modifica una tabla
DROP TABLE	Borra una tabla
CREATE INDEX	Crea un índice (search key)
DROP INDEX	Borra un índice.

Figura 2.6.3 Comandos SQL

SELECT
La sentencia SELECT nos permite consultar los datos almacenados en una tabla de la base de datos.
Forma básica

```
SELECT [ALL | DISTINCT ]
        <nombre_campo> [{,<nombre_campo>}]
FROM <nombre_tabla>|<nombre_vista>
     [{,<nombre_tabla>|<nombre_vista>}]
[WHERE <condición> [{ AND|OR <condición>}]]
[GROUP BY <nombre_campo> [{,<nombre_campo >}]]
[HAVING <condición>[{ AND|OR <condición>}]]
[ORDER BY <nombre_campo>|<indice_campo> [ASC | DESC]
          [{,<nombre_campo>|<indice_campo> [ASC | DESC ]}]]
```

SELECT	Palabra clave que indica que la sentencia de SQL que queremos ejecutar es de selección.
ALL	Indica que queremos seleccionar todos los valores. Es el valor por defecto y no suele especificarse casi nunca.
DISTINCT	Indica que queremos seleccionar sólo los valores distintos.
FROM	Indica la tabla (o tablas) desde la que queremos recuperar los datos. En el caso de que exista más de una tabla se denomina a la consulta "consulta combinada" o "join". En las consultas combinadas es necesario aplicar una

	condición de combinación a través de una cláusula WHERE.
WHERE	Especifica una condición que debe cumplirse para que los datos sean devueltos por la consulta. Admite los operadores lógicos AND y OR.
GROUP BY	Especifica la agrupación que se da a los datos. Se usa siempre en combinación con funciones agregadas.
HAVING	Especifica una condición que debe cumplirse para que los datos sean devueltos por la consulta. Su funcionamiento es similar al de WHERE pero aplicado al conjunto de resultados devueltos por la consulta. Debe aplicarse siempre junto a GROUP BY y la condición debe estar referida a los campos contenidos en ella.
ORDER BY	Presenta el resultado ordenado por las columnas indicadas. El orden puede expresarse con ASC (orden ascendente) y DESC (orden descendente). El valor predeterminado es ASC.

INSERT
Una sentencia INSERT de SQL agrega uno o más registros a una (y sólo una) tabla en una base de datos relacional.
Forma básica

```
INSERT INTO 'tablanombre' ('columna1',['columna2,... '])
VALUES ('valor1', ['valor2,...'])
```

O también se puede utilizar como:

```
INSERT tablanombre VALUES ('valor1','valor2')
```

Las cantidades de columnas y valores deben ser iguales. Si una columna no se especifica, le será asignado el valor por omisión. Los valores especificados (o implícitos) por la sentencia INSERT deberán satisfacer todas las restricciones aplicables. Si ocurre un error de sintaxis o si alguna de las restricciones es violada, no se agrega la fila y se devuelve un error.

Formas avanzadas

Para insertar varias filas en MS SQL puede utilizar esa construcción:

```
INSERT INTO tablanombre ('columna1',['columna2,... '])
SELECT 'valor1', 'valor2'
WHERE condición
```

UPDATE
Una sentencia UPDATE de SQL es utilizada para modificar los valores de un conjunto de registros existentes en una tabla.
Ejemplo

```
UPDATE tablanombre SET columna = 'un valor' WHERE 'columna2'= 'N';
```

DELETE
Una sentencia DELETE de SQL borra uno o más registros existentes en una tabla.
Forma básica

```
DELETE FROM tabla WHERE columna1 = 'valor1';
```

Disparadores

Los disparadores, también conocidos como desencadenantes (triggers en inglés) son definidos sobre la tabla en la que opera la sentencia INSERT, y son evaluados en el contexto de la operación. Los desencadenantes BEFORE INSERT permiten la modificación de los valores que se insertarán en la tabla. Los desencadenantes AFTER INSERT no pueden modificar los datos de ahora en adelante, pero se puede utilizar para iniciar acciones en otras tablas.

Los sistemas de gestión de base de datos con soporte SQL más utilizados son, por orden alfabético:

- DB2
- Firebird
- HSQL
- Informix
- Interbase
- MariaDB
- Microsoft SQL Server
- MySQL
- Oracle
- PostgreSQL
- Progress
- PervasiveSQL
- SQLite
- Sybase ASE

Los siguientes ejemplos muestran cómo se parametrizan queries en ASP.

SELECT EN ASP.NET:

```
txtUserId = getRequestString("UserId");
sql = "SELECT * FROM Customers WHERE CustomerId = @0";
command = new SqlCommand(sql);
command.Parameters.AddWithValue("@0",txtUserID);
command.ExecuteReader();
```

INSERT INTO EN ASP.NET:

```
txtNam = getRequestString("CustomerName");
txtAdd = getRequestString("Address");
txtCit = getRequestString("City");
txtSQL = "INSERT INTO Customers (CustomerName,Address,City)
Values(@0,@1,@2)";
command = new SqlCommand(txtSQL);
command.Parameters.AddWithValue("@0",txtNam);
command.Parameters.AddWithValue("@1",txtAdd);
command.Parameters.AddWithValue("@2",txtCit);
command.ExecuteNonQuery();
```

Caracteres sustitutos en SQL (Wildcard)

En SQL, estos caracteres son usados con el operador LIKE.

Los caracteres son:

Caracter	Descripción
%	Sustituto para cero o más caracteres
_	Sustituto para un caracter
[lista de caracteres]	Conjunto y rango de caracteres
[^lista de caracteres]or [!lista de caracteres]	Empaten solo los caracteres no especificados en la lista

Figura 2.6.4 Caracteres Willcard

Uniones en SQL (JOINs)

Los diferentes tipos de JOINs que pueden usarse son:

- **INNER JOIN**: Regresa todos los registros donde exista al menos una coincidencia entre ambas tablas
- **LEFT JOIN**: Regresa todos los registros de la tabla la izquierda, y aquellos que coincidan del table de la derecha
- **RIGHT JOIN**: Regresa todos los registros de la tabla la derecha, y aquellos que coincidan del table de la izquierda
- **FULL JOIN**: Regresa todos los registros de la table que coincidan en una de las tablas

SQL INNER JOIN Syntax

SELECT *column_name(s)*
FROM *table1*
INNER JOIN *table2*
ON *table1.column_name=table2.column_name*;

o

SELECT *column_name(s)*
FROM *table1*
JOIN *table2*
ON *table1.column_name=table2.column_name*;

INNER JOIN es el mismo que JOIN.

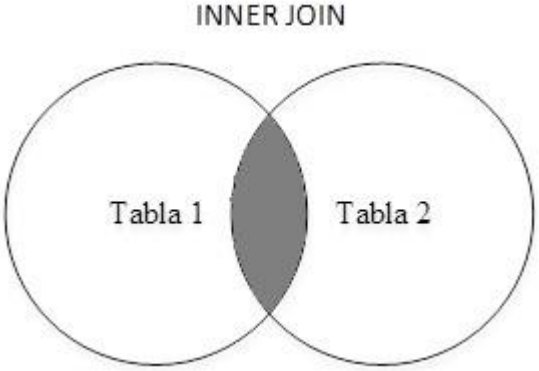


Figura 2.6.5 INNER Join SQL

SINTAXIS SQL LEFT JOIN

SELECT *column_name(s)*
FROM *table1*
LEFT JOIN *table2*
ON *table1.column_name=table2.column_name*;

o:

SELECT *column_name(s)*
FROM *table1*
LEFT OUTER JOIN *table2*
ON *table1.column_name=table2.column_name*;

En algunas bases de datos LEFT JOIN es llamado LEFT OUTER JOIN.

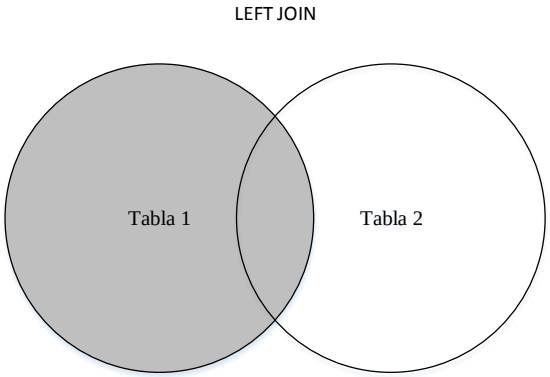


Figura 2.6.6 Left Join SQL

SINTAXIS SQL RIGHT JOIN

```
SELECT column_name(s)
FROM table1
RIGHT JOIN table2
ON table1.column_name=table2.column_name;
```

o:

```
SELECT column_name(s)
FROM table1
RIGHT OUTER JOIN table2
ON table1.column_name=table2.column_name;
```

En algunas bases de datos RIGHT JOIN es llamado RIGHT OUTER JOIN.

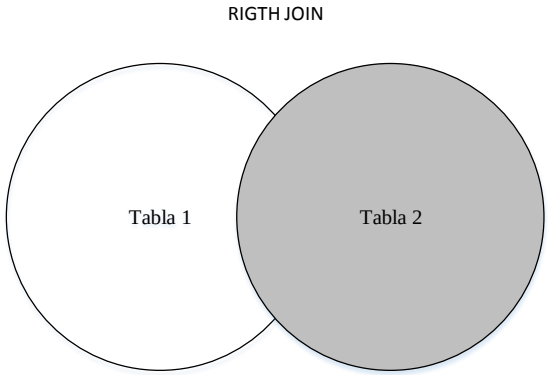


Figura 2.6.7 Rigth Join SQL

SINTAXIS SQL FULL OUTER JOIN

```
SELECT column_name(s)
FROM table1
FULL OUTER JOIN table2
ON table1.column_name=table2.column_name;
```

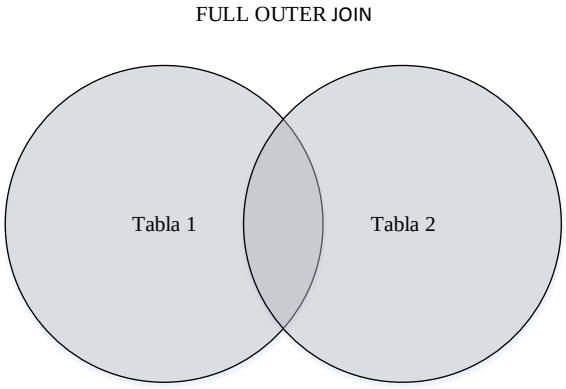


Figura 2.6.8 Full Outer Join SQL

SINTAXIS SQL UNION

SELECT *column_name(s)* FROM *table1*
UNION
SELECT *column_name(s)* FROM *table2*;

El operador UNION selecciona solo los valores distintos por default

SINTAXIS SQL UNION ALL

SELECT *column_name(s)* FROM *table1*
UNION ALL
SELECT *column_name(s)* FROM *table2*;

La sentencia SELECT INTO

Selecciona datos de una tabla y los inserta en una nueva.

SINTAXIS SQL SELECT INTO

We can copy all columns into the new table:

SELECT *
INTO *newtable* [IN *externaldb*]
FROM *table1;*

o:

SELECT *column_name(s)*
INTO *newtable* [IN *externaldb*]
FROM *table1;*

Funciones SQL

SQL tiene definidas muchas funciones internas para realizer operaciones con los datos.

Funciones de agregación SQL

Las funciones de agregación regresan un solo valor, calculado de los valores de una columna.

Funciones de agregación más usadas:

- AVG() – Regresa el promedio
- COUNT() - Regresa el número de registros
- FIRST() - Regresa el primer valor
- LAST() - Regresa el ultimo valor
- MAX() - Regresa el valor maximo
- MIN() - Regresa el valor mínimo
- SUM() - Regresa la suma

Funciones Escalares

Una función escalar regresa un solo valor basado en el valor de entrada.

Funciones escalares más usadas:

- UCASE() Convierte el campo en mayúsculas
- LCASE() Convierte un campo en minúsculas
- MID() Extraen caracteres de un texto
- LEN() Regresa el largo de un campo de texto
- ROUND() Redondea un campo numérico al número de decimales especificado
- NOW() Regresa la fecha y la hora del sistema
- FORMAT() Formatea un campo para ser desplegado

3. *ANÁLISIS Y PLANTEAMIENTO DEL SISTEMA*

1. Problemática actual

Como ya se mencionó en capítulo 1 la compañía requiere de realizar muchas mejoras en los procesos que realiza. Uno de esos problemas es el registro de la información de las órdenes de servicio (OS) y los documentos que la complementan.

A manera de resumen y resaltando la problemática con la información de las ordenes de servicio tenemos.

Una compañía que vende servicios de comunicaciones con varias sucursales en el país, principalmente en las ciudades de Querétaro, Ciudad de México, Estado de México, Hidalgo, Guanajuato, Morelos, desea conocer la información sobre las órdenes de servicio que se generan y que dicha información esté disponible para su explotación.

La empresa cuenta con varios socios que requieren de la información de las OS en la operación diaria y desarrollo de la relación comercial.

La empresa ha crecido rápidamente y los procesos manuales, en su mayoría, han sido rebasados por lo que requiere de nuevos sistemas que ayuden y eficienten la operación y toma de decisiones. Lejos de querer establecer controles sobre todas las áreas involucradas en la instalación de enlaces, desea un sistema que le permita conocer la información necesaria para verificar el número de instalaciones, órdenes de servicio, órdenes de compra, etc. sin que esta operación deteriore la operación normal de cada área.

En algunos casos el establecimiento del registro de información es prioritario pues no se cuenta con ningún registro de la operación.
La información actual de las OS y documentos relacionados está dispersa en varios sistemas, en registros en carpetas y en múltiples hojas electrónicas.

El sistema que se desea no pretende controlar toda la operación de la empresa, ni ser el web site de la empresa, su principal función es el registro de la información perteneciente a las OS y servir de referencia en los procesos. El sistema no será la base de facturación o emisión de contratos, pagos o expediente de cliente entre otros.

2. Descripción de las áreas involucradas

Todas las áreas participan en mayor o menor medida en el registro y generación de información.
De acuerdo al diagrama de los procesos básicos de la empresa figura 1.4.1 las áreas involucradas son:

- Dirección
- Gerencia
- Ventas y mercadotecnia
- Ingeniería y sistemas
- Servicio al cliente
- Compras

- Administración

Los involucrados en los procesos de la información ya sea para su aprovechamiento o generación son:
En la empresa:

- Director general
- Gerente de sucursal
- Vendedores
- Gerente de operaciones
- Operadores
- Programador
- Instaladores
- Encargado de Mesa de ayuda al cliente
- Comprador
- Administrador

En los Canales de venta:

- Vendedores
- Representante de Canal.

A nivel de los procesos la tabla siguiente indica cuales intervienen ya sea en la generación o aprovechamiento de la información de las OS y demás documentos.

Nivel	**Procesos**	**Funciones**	**Interviene**
Estratégico	Plan de negocios	Definir zona de venta	X
		Definir canales de venta	X
		Definición de los productos	X
		Integrar contratos	X
	Servicio a clientes	Levantar y documentar Requerimientos	X
		Gestionar requerimientos	X
		Notificar al cliente	
		Monitorear calidad	X
		Clasificar requerimientos	X
	Diseño de red y plataforma	Estudios para definición de zona	X
		Configuración de solución	X
		Autorización de solicitudes	X
Operación	Ventas	Busca prospectos	
		Valora previamente instalación	
		Elabora Pre cotización	X
		Elabora cotización	X
		Elabora contrato	X
		Elabora orden de servicio	X
	Instalación de servicios	Planear y programar instalación / reparación	X
		Asignar personal y tramitar viáticos	
		Solicitar equipo y materiales	X

		Preparar equipo y materiales	
		Instalar, revisar, reparar	X
	Activación de servicios	Configurar servicio	X
		Probar activación	X
		Actualizar información de equipos activos	
		Elaborar reporte de activación	X
		Revisar y resolver fallas	
Soporte	Administración	Revisar contratos con terceros	X
		Facturar	X
		Elaborar cuentas por cobrar	X
		Preparar nómina y comisiones	X
		Pagar nómina y proveedores	X
		Manejar Tesorería	
		Gestionar contabilidad	
		Presentar impuestos y declaraciones	
	Compras e inventario	Recibe solicitudes	X
		Elabora cotizaciones	
		Gestiona autorización	
		Realiza compras	
		Almacena equipo	
		Entrega equipo	X
		Realiza inventario	

Figura 3.2.1 Información en Procesos Básicos del Negocio

3. Especificación de requerimientos de información

Para obtener la información necesaria de una OS y poder seguir su ciclo es necesario obtener información de otros documentos además de la propia OS. Además de la información de los documentos, se requiere información de la plataforma y recursos técnicos con los que cuenta la empresa.

Estos documentos son manejados en las diferentes áreas entre los que tiene que ver con la información relevante en las OS son:

- Orden de Servicio
- Propuesta de cotización
- Estudio de Línea de Vista (ELVIS)
- Análisis de infraestructura de Red (AIR)
- Orden de compra /Propuesta de precios.
- Contrato
- Solicitud de adhesión
- Orden de instalación
- Plan de instalación
- Equipos para instalación
- Permiso de instalación
- Acta de recepción

- Facturas de servicio
- Documentos de cancelación

Otros documentos que se generan durante la operación y que no siempre se generan o son simplemente avisos o se encuentran en otros sistemas son:

- Solicitud de viáticos
- Autorizaciones de entrada
- Documentos de configuración
- Expediente de cliente
- Recibos de Pago
- Avisos de cancelación
- Notas de crédito
- Avisos de mantenimiento
- Ordenes de Reparación
- Ordenes de mantenimiento
- Quejas de servicio
- Facturas por servicios Extraordinarios
- Registro del help desk

Todos los documentos tiene la información relevante de la empresa ya se expresamente en los logos y papeles oficiales o escritos dependiendo del documento.

- Nombre de la Empresa
- RFC
- Director General
- Director Comercial
- Representante Legal
- Dirección
- Sucursal
- Teléfonos
- Correo
- Página web

Podemos describir el ciclo de una OS de manera muy general con el propósito de facilitar su entendimiento.

1. El cliente se contacta con la empresa
2. Se realiza una visita
3. Se elabora la orden de servicio
4. Se realiza el estudio de línea de vista
5. Se elabora la pre cotización
6. Se realiza el estudio de Factibilidad
7. Se elabora la Orden de Compra o Propuesta de Precios
8. Se elabora el contrato
9. Se firma Contrato de adhesión
10. Se realiza la orden de instalación
11. Se preparan los equipos de instalación
12. Se realizan las visitas de instalación
13. Se entrega Acta de recepción.

14. Se factura.
15. Se cobra

En la figura 3.3.1se puede ver un esquema simplificado para apreciar las entidades, actores y documentos en el proceso mencionado.

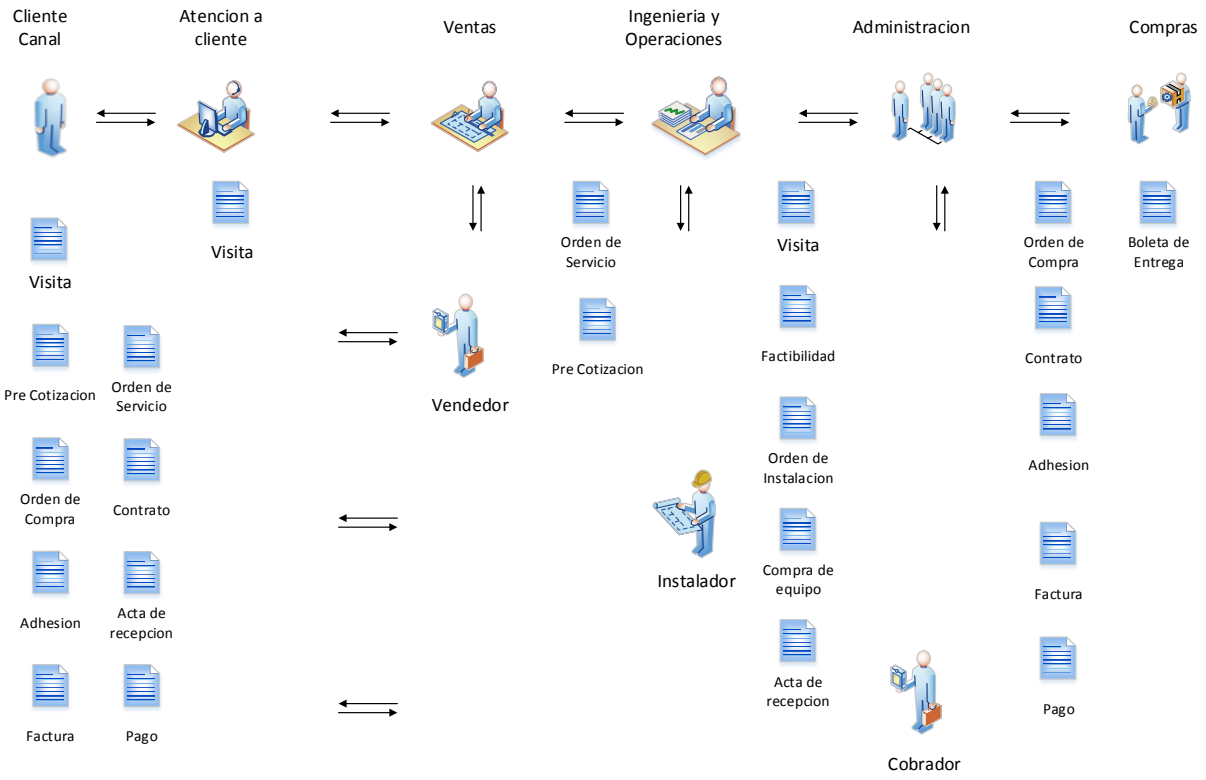


Figura 3.3.1 Entidades, Actores y Documentos

Se puede apreciar la cantidad de documentos que se requieren y que estos se encuentran en distintos estados de elaboración durante el proceso.

Más formalmente

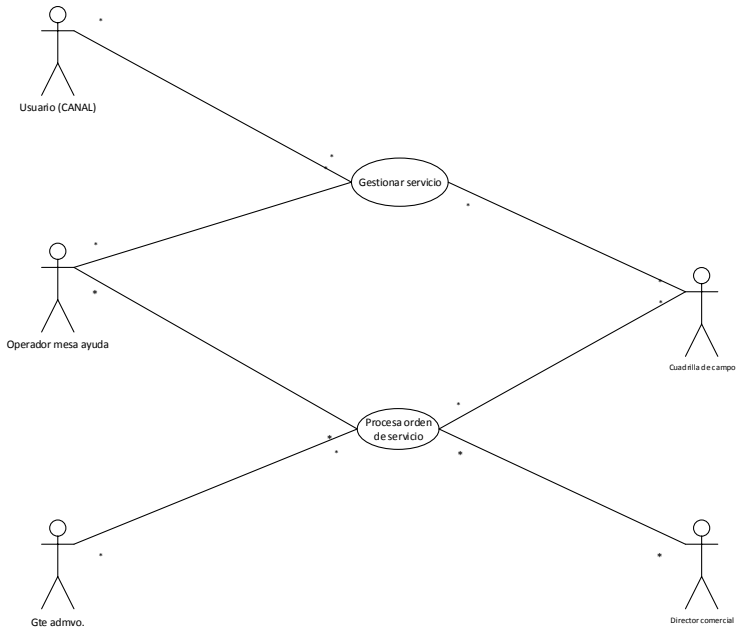


Figura 3.3.2 Flujo de órdenes

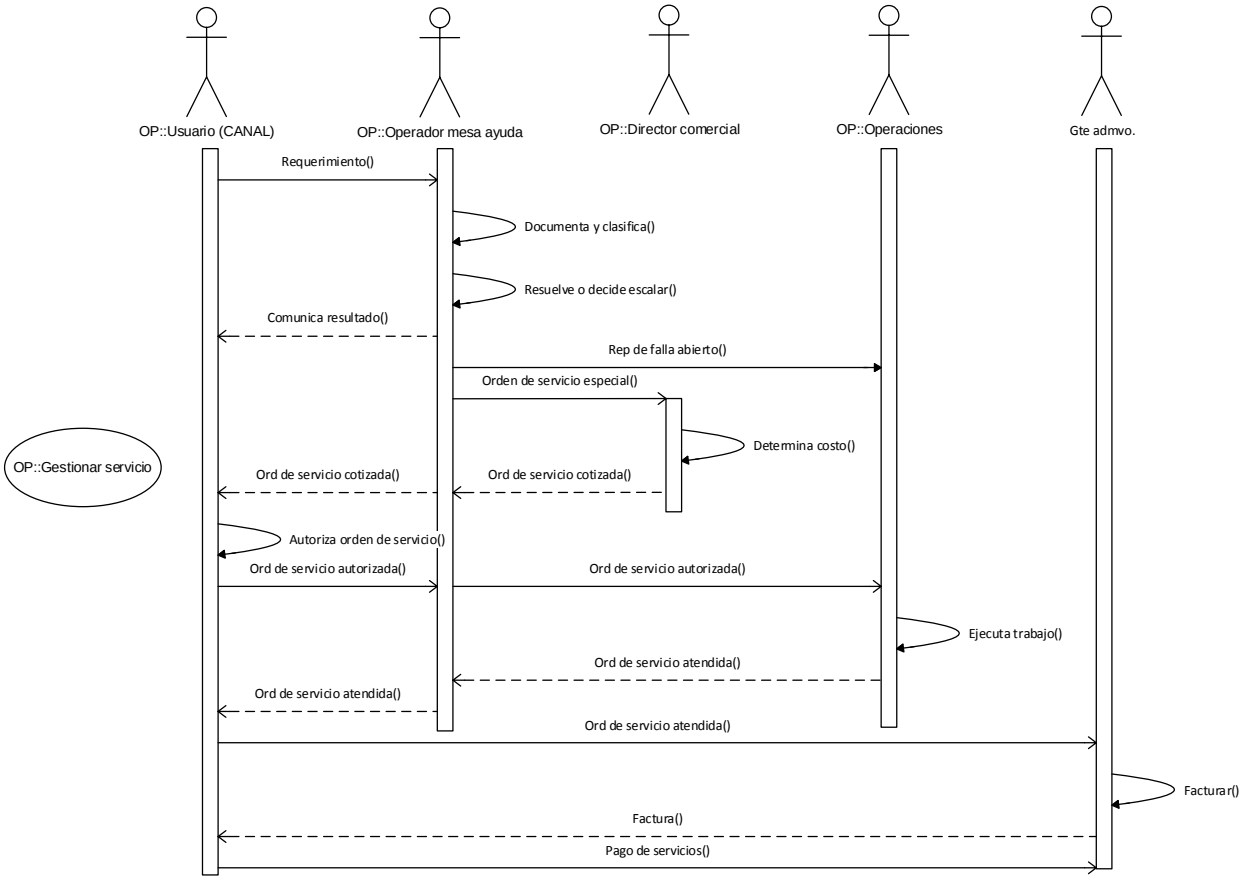


Figura 3.3.3 Gestionar orden

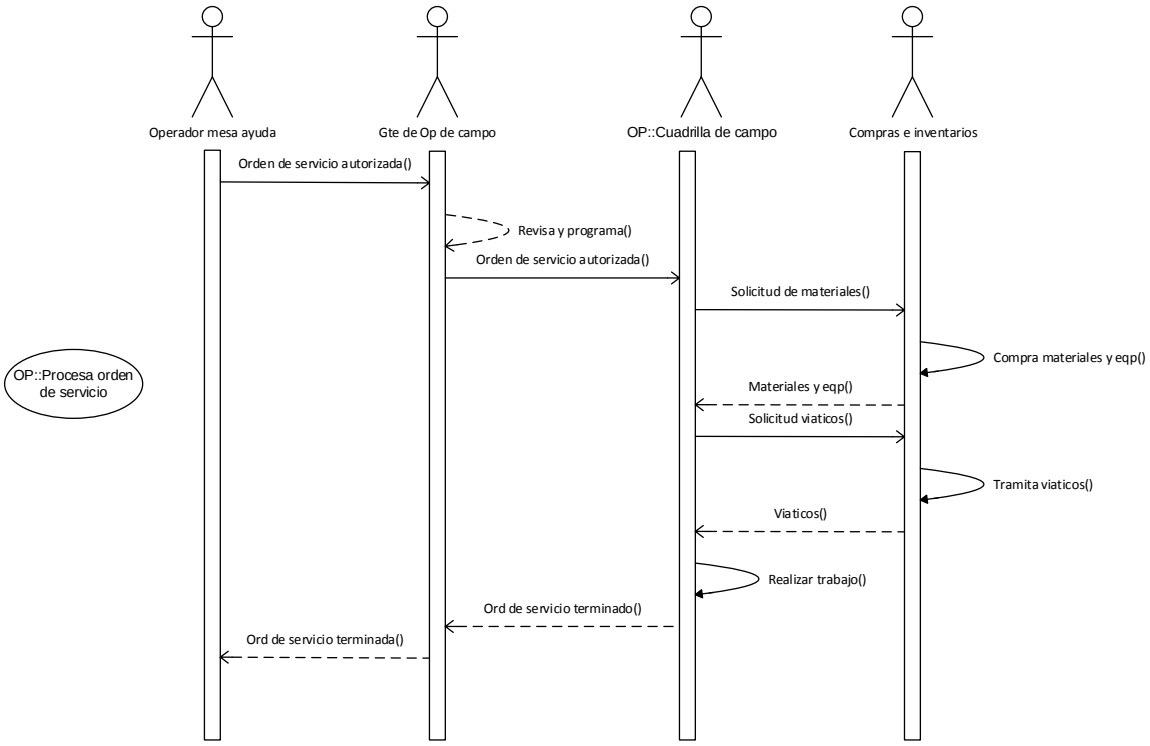


Figura 3.3.4 Procesar Orden

Para una Orden de Servicio se maneja la siguiente información:

Información en la **OS**

Dato	**Descripción**
Folio	Número consecutivo
Clave	Clave de identificación
Canal	Canal de venta
RS. Empresa	Razón Social del Cliente
Elaboración	Fecha de elaboración de OS
Dirección	Dirección del cliente calle número, colonia estado, etc
RFC	Registro federal de Contribuyentes
Contacto1	Contacto entre la empresa y el cliente
Telefono1	Número del contacto entre la empresa y el cliente
Telefono2	Número del contacto entre la empresa y el cliente alterno
Fax	Número de Fax
Producto	Descripción del producto
IP	Cantidad de Dirección IP
Contrato	Descripción del plazo contratado
Calleprox1	Entre calle
Calleprox2	Entre calle
Solicitante	Nombre del Solicitante
Fecha Solución	Fecha de terminación
Ejecutivo	Ejecutivo de cuenta
Comentarios	Con respecto a cualquier etapa
STATUS	Solicitada, en trámite, ect
Servicio	Normal, urgente, prioritario
Seguimiento	Aceptado, Rechazado, Modificado

Figura 3.3.5 Información de Orden de Servicio

A partir de la información de la orden de servicio se genera un expediente. Este expediente consta de:

1. Equipos para instalación
2. Estudio de factibilidad
3. Instalación
4. Orden de compra
5. Acta de recepción

Además de los documentos legales
Como son

1. El contrato
2. La adhesión
3. Las facturas

El **Estudio de factibilidad** contiene la siguiente información.

Dato	Descripción
Folio	Número consecutivo
Recibe	Empleado que recibe la solicitud
Recepción	Fecha de recepción de la solicitud
Límite de entrega	Fecha límite de entrega del estudio
Programación	Fecha de realización planeada
Resultado	Factible , no factible, etc.
Tipo de estudio	Prioridad
Comentarios	Comentarios de cualquier etapa
GPS N	Dirección GPS N
GPS W	Dirección GPS W
Altura	Altura del enlace
Repetidora	Torre repetidora
Distancia	Distancia del enlace
Esfuerzo	Tiempo de instalación h/h
Fecha de entrega	Fecha real de entrega
Nombre realizo EF	Empleado que realizo el estudio
Observaciones Cliente	Comentarios de cualquier etapa
Recibió	Encargado del Cliente
Estatus	Aceptado, rechazado
Seguimiento	Estado en el tiempo
Numero de OS	Orden de servicio relacionada
Ubicación	Fotografía de torre cercana
Azotea	Fotografía azotea de cliente
Centro de Computo	Fotografía centro de computo
Trayecto	Dibujo de trayectoria
Canal	Canal comercial

Figura 3.3.6 Información de Estudio de Factibilidad

La información de la **Orden de Compra o** propuesta de precios **(OCPP)**.

Dato	Descripción
Folio	Número consecutivo
Orden de Servicio	Número OS
Canal	Canal comercial
Duración del Servicio	Duración en años
Ancho de banda	Anchi de Banda del servicio
Costo de activación	Cantidad en pesos
Descuento en activación	Cantidad en pesos
Periodicidad	Periodicidad del pago
Costo de Renta	Cantidad en pesos
Descuento de Renta	Cantidad en pesos
Porcentaje de Descuento	Cantidad %
Elaboración	Fecha de Elaboración
Elaboro	Empleado OCPP
Representante	Nombre del Representante legal del Cliente
Representante P	Nombre del Representante legal de la empresa
Tipo servicio	Urgente, normal, …
Estatus	Solicitado, Rechazado, Aceptada
Elaboro	Registro
Envío	Fecha de envío de la OCPP
Comentarios	Comentarios de cualquier etapa
Respuesta Cliente	Comentarios de cualquier etapa
Seguimiento	Estado en el tiempo

Figura 3.3.7 Información de OCPP

Una OCPP Impresa

LOGO

ANEXO B

PROPUESTA DE PRECIOS

Información del Cliente	**Denominación o Razón Social**	[redacted]
	Usuario Final	[redacted]
	Contacto y cuenta de correo	[redacted]
	Dirección	Av. Del Márquez #36-A P. Ind. Bernardo Quintana
	Teléfonos y Fax	(442) 2157747

Duración del Servicio	X 1 año	_ 2 años	_ 3 años

Resumen de Precios

Cargos por evento (una sola vez)	**Velocidad**	**Precio lista**	**Descuento**	**Precio final**
Activación ADP (renta de un puerto a Internet)	256 Kbps			
Activación hospedaje				
Activación Co-ubicación de servidores				
Activación solución corporativa [redacted]				
Subtotal				

Cargos recurrentes (de acuerdo al periodo de pago)	**Periodicidad**	**Precio x periodo**	**Descuento**	**Total**
Renta ADP	Mensual	$ 5,300.00	$ 530.00*	$ 4,770.00
Hospedaje				
Co-ubicación de servidores (housing)				
Solución corporativa [redacted]				
		$ 5,300.00	$ 530.00	$ 4,770.00
Subtotal				

Periodicidad: mensual, trimestral, semestral y anual
* Descuento especial de 10% por contrato a un año

Total Cargos	**$ 5,300.00**	**$ 530.00**	**$ 4,770.00**

Precios arriba expresados pesos M.N. y NO incluyen 15% de IVA

Condiciones

La fecha de inicio del servicio se tomará del acta de recepción del servicio-
El cliente llenará una PROPUESTA DE PRECIOS por servicio solicitado

Firmas Propuesta presentada por:

Nombre del Representante de [redacted]	Firma del Representante de [redacted]	Fecha
[redacted]		28/NOVIEMBRE/20

Aceptación de la propuesta:

Nombre del Cliente o Representante Legal	Firma del Cliente o Representante Legal	Fecha
[redacted]		28/NOVIEMBRE/20

La presente propuesta forma parte integrante del contrato de Suministro de Servicios de Valor Agregado, por lo que el Cliente acepta los precios aquí establecidos.

Figura 3.3.8 OCPP Impresa

La información de un **Acta de Recepción**

Dato	Descripción
Folio	Número consecutivo
Consecutivo	Número OS
Fecha elaboración	Fecha en que se elaboro
Subred	Direcciones IP de subred
Mask	Direcciones IP de la Mascara
Ancho de banda	Ancho de banda contratado
Distancia	En Kilómetros
Punta AN	Coordenadas punta AN
Punta AW	Coordenadas. Punta AW
Punta BN	Coordenadas. Punta BN
Punta BW	Coordenadas. Punta BW
Base	Base de conexión
Punta A	Lugar Torre de comunicaciones
Punta B	Lugar edificio cliente
Frecuencia	Numero de Banda
netmonlogin	Usuario de acceso para monitoreo
netmonpass	Clave de acceso para monitoreo
Comercial	Nombre director comercial
Administrativo	Nombre director administrativo
Técnico	Nombre encargado técnico
Representante	Nombre del representante legal
Estatus	Aceptado, Rechazado, Cancelado
Seguimiento	Estado en el tiempo
Canal	Canal comercial

Figura 3.3.9 Información de Acta de Recepción

La información que también es relevante.

- Canales Comerciales
- Sucursales
- Empleados
- Clientes
- Torres de Comunicaciones
- Vendedores

La información relevante de los **Canales Comerciales**

Dato	Descripción
Identificación	Clave de identificación
Razón Social	Razón Social
RFC	Registro Federal de Contribuyentes
Dirección	Dirección Completa
Representante Legal	Representante Legal del Canal
Contacto	Nombre del contacto
Teléfono	Teléfono
Email	Email
Director	Nombre del Director
NEMO	Nombre corto

Figura 3.3.10 Información de Canales Comerciales

La información relevante de los **Empleados**

Dato	Descripción
Número	Número de empleado
Identificación	Clave de Identificación
Nombre	Nombre completo
RFC	Registro Federal de Contribuyentes
Dirección	Dirección completa
Teléfono	Teléfono
Celular	Celular
Nacimiento	Fecha de Nacimiento
Departamento	Departamento
Puesto	Puesto
Seguridad	Seguridad

Figura 3.3.11 Información de Empleados

Información Almacenada

La obtención de las bases o registros tanto documentales como electrónicos es fundamental para formar la base del momento actual (al instalar la aplicación), además de proporcionar, en el caso de existir, información para definir a los mismos.

4. Alcance

Conforme a las políticas de la empresa y los requerimientos iniciales de la empresa, se pueden definir los alcances del proyecto en las áreas de :

- Equipos y Plataforma
- Sistema
- Funciones
- Personal y Capacitación

Equipos y Plataforma

Debido al presupuesto y recursos asignados para este rubro.
El equipo, programas, instalaciones, licencias serán proporcionados por la empresa en el tiempo acordado para tal propósito.

La plataforma y sistema funcionará en su totalidad en las instalaciones de la Empresa.

Las capacidades de los equipos de la empresa serán las tomadas en cuenta para el desarrollo del sistema.

Las versiones de los programas requeridos estarán actualizadas hasta una versión anterior a la más nueva en el mercado. La empresa realizará las actualizaciones necesarias del software actual y las licencias necesarias.

Durante el desarrollo del sistema se le informará a la empresa de otros programas y licencias que se requieren para el funcionamiento del sistema.

Una vez desarrollada la aplicación la empresa se compromete a establecer un tiempo en los equipos y sistemas para la instalación y pruebas en la plataforma.

Las modificaciones requeridas en las configuraciones de los equipos y sistemas serán realizados por personal de la empresa bajo las especificaciones de los programas requeridos.

En el caso de que la empresa no pueda realizar las modificaciones o instalaciones necesarias, estas serán realizadas por el desarrollador previa solicitud y acuerdo.

Dominios, registros, direcciones ip, etc., de la plataforma web serán responsabilidad de la empresa.

Sistema
El sistema:

Se desarrollará en otras instalaciones para no interrumpir la operación.

Será una aplicación web cliente servidor accesible desde cualquier lugar mediante un explorador

No será programado como sistema de uso crítico porque ni la adquisición de la información ni su explotación será determinante en el mismo momento.

De acuerdo al esquema de usuarios y manejo de información la cantidad de usuarios simultáneos se considerara una aplicación no crítica en ese aspecto.

Sera entregado junto con el código a un encargado capacitado para ello.

Podrá configurarse solo por personal capacitado.

Los respaldos, depuraciones y mantenimientos necesarios son responsabilidad de la empresa.

Sera desarrollado en plataforma y productos de Microsoft.

No se comunicará con los demás sistemas

Podrá accederse desde cualquier área.

Contemplará un sistema de seguridad de acceso

Integrará un módulo de reportes

Funciones

Tanto el cliente como el servidor se entregarán con las configuraciones básicas para su funcionamiento.

Solo contendrá usuarios básicos y ejemplo de tipos usuarios.

La seguridad se manejará independiente de los otros sistemas y plataformas.

La empresa proporcionará la información histórica y hasta el momento de la instalación para ser procesada en la nueva plataforma.

La empresa será responsable, una vez superadas las pruebas, de la liberación de la aplicación.

El desarrollador no será responsable por cambios en la versión o configuración en los sistemas y equipos.

Los cambios que por actualización en la plataforma, requieran realizarse al sistema serán responsabilidad de la empresa.

El acceso a clientes finales no será considera en la funcionalidad de sistema.

La empresa será responsable del ciclo de modificaciones al sistema una vez liberado

Las pruebas a la aplicación consideran ejemplos de captura y configuración del sistema únicamente. No se consideran en las pruebas la creación de funciones, bases, tablas, esquemas, etc. de la plataforma.

La información en la base de datos podrá explotarse fuera de la aplicación.

Personal y Capacitación

La capacitación en el sistema será proporcionada a una persona designada por la empresa y será la encargada de capacitar al personal necesario.

La empresa nombrará al personal capacitado, que considere necesario para la entrega de la aplicación y plataforma.

5. Soluciones propuestas

Basándose en lo anteriormente expuesto se propone el sistema de registro de órdenes que permitirá la captura y explotación de la información referente a las órdenes de servicio y otros documentos, con lo cual se espera disminuir errores y tener mayor información sobre los procesos involucrados.

En el esquema Figura 3.5.1 se puede ver los módulos propuestos.

El sistema se desarrollará como una aplicación web, que permita el acceso desde cualquier lugar mediante un explorador y las ventajas que conlleva el desarrollo con esta técnica. En la medida de lo posible se establecerán diseños que puedan ser modificados en su presentación sin alterar su funcionalidad.

La visualización de la información manejada por la aplicación debe contener facilidades que permitan el manejo de la información como ordenamientos búsquedas y presentación de documentos.

Contendrá una base de datos robusta para el crecimiento y explotación. Se espera que esto permita el crecimiento de nueva funcionalidad sin la necesidad de cambiar de bases de datos además de explotar las características mencionadas de las bases robustas, entre ellas su administración, respaldos, y oportunidad de explotación por diversas plataformas.

El sistema contendrá un sistema de seguridad que permitirá el acceso a la información de acuerdo al usuario y las funciones que realiza. Esto permitirá que la información clasificada como privada para algunos procesos no esté disponible para todos los usuarios, cumpliendo así con algunas políticas establecidas por la empresa a este respecto.

El módulo de reportes deberá permitirá la explotación de reportes específicos. Si bien se requiere el módulo de reportes este se planea como un beneficio adicional pues lo importante es la información almacenada y con orden para su explotación no solo con la aplicación.

Deberá contener una presentación de la información de la empresa o canal de distribución. Debido al modelo de negocios establecido para las ventas por canales de distribución estos desean estar presentes de alguna forma en la aplicación.

Deberá contener funciones que permitan observar el proceso de una orden de servicio desde su creación hasta su conclusión mediante estaciones de trabajo. La principal dificultad para el establecimiento de esta funcionalidad es que no se cuenta con todos los departamentos y todas las funciones como se menciono anteriormente.

El sistema permitirá la configuración de la presentación de la información sin tener la necesidad de cambiar la programación. Esto para menor dependencia de los programadores.

El sistema tendrá la capacidad para definir el flujo del proceso mediante configuración con el fin de soportar una mayor especialización en algunas funciones o la creación de nuevas.

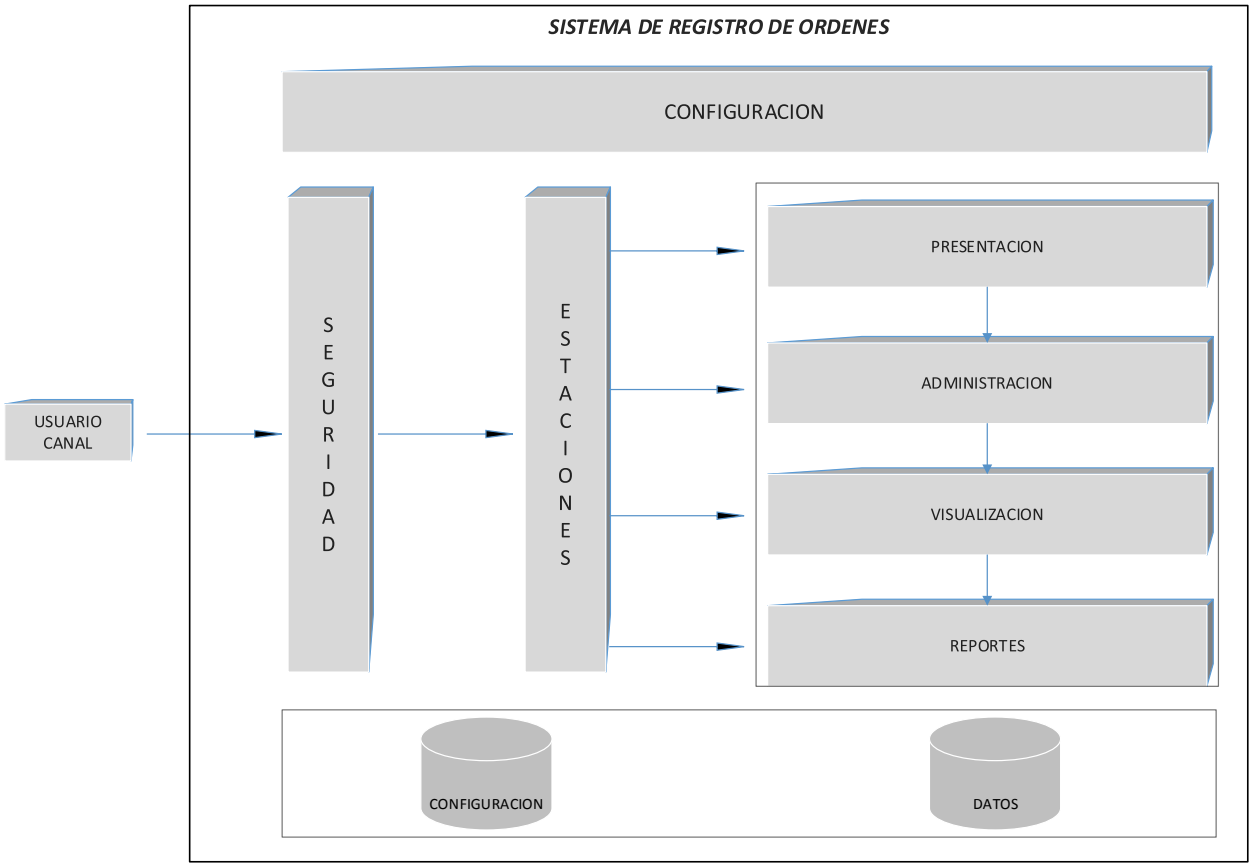


Figura 3.5.1 Módulos del Sistema de Registro de Ordenes

A continuación se describen los módulos prototipo de la aplicación y las definiciones de usuario de la aplicación.

Usuarios

Los usuarios están definidos en varios ambientes que determinan algunos componentes del sistema:

- Entorno de la empresa
- Entorno de Socios Comerciales
- Entrono Cliente Final.

Dentro de estos entornos los usuarios son

Entorno de la empresa

Vendedores, Instaladores, Coordinadores, Administradores.

Entorno de Socios Comerciales

Coordinadores, Gerentes.

Entrono Cliente Final.

Cliente, Encargados de sistemas o comunicaciones, Representantes Legales.
Para esta versión del sistema el cliente final esta fura de los alcances del proyecto.

Configuración.

Se contemplan la posibilidad de definición y cambio de las diferentes configuraciones iniciales de los elementos del sistema principalmente:

Estaciones de trabajo,
Tipos de Usuarios
Flujo de trabajo.
Visualizaciones

Seguridad

Se encargara, de acuerdo al tipo de usuario, de permitir el acceso a la aplicación. Se realizará mediante perfiles asignados y almacenados en la base de datos.
Como se puede ver en el esquema implica la definición de esquemas en cada módulo es decir:

Configuración
Presentación
Administración
Visualización
Reportes.

La seguridad incluye el acceso desde la red que será manejado por las herramientas de los sistemas seleccionados para implementar la aplicación.

Estaciones

Las estaciones de trabajo se definen como el conjunto de facilidades y permisos que puede utilizar un usuario en el sistema para realizar las funciones asignadas a su trabajo.
En este esquema, de acuerdo al usuario, el sistema solo mostrara información que le a sido permitido ver.
Si por ejemplo el usuario es un vendedor podrá ver la estación de recepción y control y no podrá ver reportes generales de la empresa y solo podrá ver y no modificar documentos del área de ingeniería

Presentación

Aquí se definen los colores y presentación de las pantallas tamaños y aspectos, así como el mapa de la aplicación o acceso a las diferentes funcionalidades de la aplicación.

Implica la posibilidad de mostrar en pantalla una presentación del socio comercial.

Se toman aquí todos los aspectos de la empresa como la definición de los colores, información y logo de la empresa que serán incluidos en pantallas y reportes.

Administración

Sea contemplan los permisos para realizar las altas bajas y cambios de los catálogos y documentos previstos para el sistema. Entre ellos:

El alta, baja o modificación de usuarios
El alta, baja o modificación de Estaciones de trabajo
El alta, baja o modificación de Estatus.

Visualización

Aquí se definirán las funcionalidades que tendrán los usuarios para ver y manejar la información disponible en las pantallas del sistema. Entre ellos:

Columnas visibles
Nombres
Ordenamientos
Búsquedas
Ligas
Listas
Vista Total.

Reportes

En el módulo se definirán los reportes impresos o visuales comunes a :

La administración del Sistema
Los documentos almacenados

Dentro de los reportes de la administración del sistema se contemplan

Definiciones de Flujo.
Catálogos

Dentro de los reportes de los documentos del sistema se contemplan reportes normales de pie y barras considerando la combinación de los elementos que los contemplan.
Entre ellos:

Órdenes de servicio
Órdenes de compra
Estudios de factibilidad
Actas de recepción

Base de datos

Se contempla la definición de dos grupos de datos:

Datos de Configuración
Datos de Catálogos y documentos.

Los datos de configuración se refieren a todos aquellos que requiere el sistema de inicio o que requiere para realizar la funcionalidad asignada. Como son
Nombre de las columnas.
Flujo de documentos.

Los Datos de catálogos y documentos

Es la información almacenada de documentos como OS y catálogos como los usuarios estaciones, estatus, etc.

Se define también la forma de acceso a los datos y los esquemas de restricciones.

Algunos ejemplos

Usuarios

- Gerente Canal
- Instalador
- Coordinador Operaciones.
- Vendedor
- Gerente Ventas
- Almacenista
- Administrador

Seguridad

- Instalador no puede dar de alta OS
- Gerente Canal solo puede ver reportes de su canal.
- El administrador tiene acceso a la modificación del Flujo.

Estaciones

- Recepción y Control
- Comercial
- Ingeniería

Configuración.

- Se definió un nombre para visualización de columnas igual al nombre de columna en la tabla que lo contiene en la base de datos mientras los usuarios encargados definen el nombre.

Presentación

- Existen siete empresas aliadas que desean mostrar su presentación en la aplicación.
- Los colores verdes y azules y blanco son los colores para el diseño de las páginas.

Administración

- El alta, baja o modificación de usuarios
- El alta, baja o modificación de Estaciones de trabajo
- El alta, baja o modificación de Estatus.

Visualización

- La OS tiene 29 campos, para visualizar, se seleccionaron 10 para mostrar según la importancia que le dé el cliente o la importancia para el proceso.
- Se podrá ordenar la información por cualquier columna presente.
- Se puede ver la totalidad de la información de un documento por medio de una liga

Reportes

- Reportes de pie del número de OS por canal
- Reportes de barras de número de OS por fecha

Base de datos

- La tabla de configuración de nombre de columnas
- Tabla de Ordenes de compras.

6. Documento de requerimientos

Para el verificar el cumplimiento de los requerimientos antes expuestos se les describe y enumera.

Requerimientos

Núm.	Nombre	Descripción	Modulo	Entregable
100	Equipos y plataforma	La empresa proporcionara: el equipo, licencias y programas necesarios para la instalación del sistema.	Sistema	Lista de requerimientos
200	Instalaciones de Cliente	La plataforma y sistema funcionará en su totalidad en las instalaciones del cliente.	Sistema	Verificación
300	Capacidad equipos	Las capacidades de los equipos de la empresa serán las tomadas en cuenta para el desarrollo del sistema.	Sistema	Lista de equipos y configuraciones
400	Actualización programas	Las versiones de los programas requeridos estarán actualizadas hasta una versión anterior a la más nueva en el mercado.	Sistema	Verificación
500	Otros Programas	Durante el desarrollo del sistema se le informará a la empresa de otros programas y licencias que se requieren para el funcionamiento del sistema.	Reportes, Seguridad	Lista de Programas y licencias
600	Pruebas	Una vez desarrollada la aplicación la empresa se compromete a establecer un tiempo en los equipos y sistemas para la instalación y pruebas en la plataforma.	Sistema	Plan de tiempos de pruebas e instalación
700	Configuración Plataforma	Las modificaciones requeridas en las configuraciones de los equipos y sistemas serán realizados por personal de la empresa bajo las especificaciones de los programas requeridos.	Sistema	Verificación
800	Solicitud configuración	Cuando la empresa no pueda realizar las modificaciones o instalaciones necesarias en la plataforma, estas serán realizadas por el desarrollador previa solicitud y acuerdo.	Sistema	Solicitud
900	Plataforma web	Dominios, registros, direcciones ip, etc., de la plataforma web serán responsabilidad de la empresa.	Sistema	Entrega de plataforma
1000	Instalaciones del desarrollo	El sistema se desarrollará en otras instalaciones para no interrumpir la operación.	Sistema	Verificación
1100	Sistema web	Será una aplicación web cliente servidor accesible desde cualquier lugar mediante un explorador	Sistema	Verificación
1200	ASP net	El sistema se desarrollará con tecnología de Microsoft para desarrollo web con páginas web ASP.net.	Sistema	Verificación
1300	Visual Script	El lenguaje principal será Visual script en el Servidor.	Sistema	Verificación
1400	Sistema no critico	El sistema No será programado como sistema de uso crítico porque ni la adquisición de la información ni su explotación será determinante en el mismo momento.	Sistema	Recomendación
1500	Código	El sistema será entregado junto con el código.	Sistema	Verificación
1600	Encargado Desarrollo	El código se entregara a un encargado capacitado para ello.	Sistema	Verificación
1700	Responsable Configuración	Podrá configurarse solo por personal capacitado.	Configuración	Asignación
1800	Responsabilidad	Los respaldos, depuraciones y	Sistema	Verificación

	Plataforma	mantenimientos necesarios de la plataforma son responsabilidad de la empresa.		
1900	Plataforma Microsoft	Sera desarrollado en plataforma y productos de Microsoft.	Sistema	Especificaciones
2000	Interfaces	El sistema no tendrá interfaces con otros sistemas	Sistema	Verificación
2100	Reportes	Integrará un módulo de reportes	Reportes	Verificación
2200	Configuración base	Tanto el cliente como el servidor se entregarán con las configuraciones básicas para su funcionamiento.	Configuración	lista Configuración
2300	Usuarios Base	El sistema solo contendrá usuarios básicos y ejemplo de tipos usuarios.	Sistema	Verificación
2400	Seguridad tipo	La seguridad se manejará independiente de los otros sistemas y plataformas.	Seguridad	Verificación
2500	Información Histórica	La empresa proporcionará la información histórica y hasta el momento de la instalación para ser procesada en la nueva plataforma.	Base de datos	Documentación
2600	Información Transformación	El desarrollador transformará la información histórica para la aplicación	Base de datos	Base inicial
2700	Liberación	La empresa será responsable, una vez superadas las pruebas, de la liberación de la aplicación.	Sistema	Verificación
2800	Responsabilidad cambios	El desarrollador no será responsable por cambios en la versión o configuración en los sistemas y equipos.	Sistema	Verificación
2900	Responsabilidad cambios de plataforma	Los cambios que por actualización en la plataforma, requieran realizarse al sistema serán responsabilidad de la empresa.	Sistema	Verificación
3000	Clientes Finales	El acceso a clientes finales no será considera en la funcionalidad del sistema.	Sistema	Verificación
3100	Pruebas	Las pruebas a la aplicación considera ejemplos de captura y configuración del sistema únicamente. No se consideran en las pruebas la creación de funciones, bases, tablas, esquemas, etc. de la plataforma.	Sistema	Acuerdo
3200	Acceso de información Externo	La información en la base de datos podrá explotarse fuera de la aplicación.	Sistema	Verificación
3300	Responsable de uso	La capacitación en el sistema será proporcionada a una persona designada por la empresa y será la encargada de capacitar al personal necesario.	Sistema	Asignación, Capacitación
3400	Responsable plataforma	La empresa nombrará al personal capacitado, que considere necesario para la entrega de la aplicación y plataforma.	Sistema	Asignación, Capacitación
3500	Capacitación Reportes	El desarrollador capacitara al encargado en los programas adicionales de la aplicación como el módulo de reportes.	Sistema	Asignación, Capacitación
3600	Responsabilidad BD	La empresa será responsable de la administración de la base de datos.	Base de datos	Acuerdo
3700	Responsabilidad Ciclo del software	La empresa será responsable de la administración del sistema que contempla, manejo de versiones, respaldos, modificaciones, actualizaciones. En general del ciclo de vida del sistema.	Sistema	Acuerdo
3800	Diseño UI	En la medida de lo posible se establecerán diseños que puedan ser modificados en su presentación sin alterar su funcionalidad.	Sistema	Verificación
3900	Facilidades UI	La visualización de la información manejada por la aplicación debe contener facilidades que permitan el manejo de la información como ordenamientos búsquedas y presentación de documentos.	Visualización	Verificación

4000	Base de Datos	Contendrá una base de datos robusta para el crecimiento y explotación.	Base de datos	Lista de Capacidades
4100	Seguridad Información	La información clasificada como privada para algunos procesos no estará disponible para todos los usuarios, cumpliendo así con algunas políticas establecidas por la empresa a este respecto	Seguridad, Visualización	Verificación
4200	Modulo de reportes	El módulo de reportes deberá permitir la explotación de reportes específicos. Si bien se requiere el módulo de reportes este se planea como un beneficio adicional pues lo importante es la información almacenada y con orden para su explotación no solo con la aplicación.	Reportes	Verificación
4300	Interface de usuario canales	El sistema deberá contener una presentación de la información de la empresa o canal de distribución.	Visualización	Especificación de presentación
4400	No web site	La aplicación no será el web site de la empresa	Sistema	Acuerdo
4500	Proceso de OS	Deberá contener funciones que permitan observar el proceso de una orden de servicio desde su creación hasta su conclusión mediante estaciones de trabajo.	Estaciones de Trabajo, Visualización	Verificación
4600	Configuración UI	El sistema permitirá la configuración de la presentación de la información sin tener la necesidad de cambiar la programación	Visualización	Verificación
4700	Flujo de proceso	El sistema tendrá la capacidad para definir el flujo del proceso mediante configuración	Configuración	Verificación
4800	Configuración Usuarios	El sistema será capaz de manejar los diferentes tipos de usuarios.	Seguridad	Verificación
4900	Configuración Básica	El sistema contendrá la configuración básica de manejo de ordenes	Configuración	Verificación
5000	Documentos Estatus	Los documentos contendrán un estatus que permitan verificar su avance	Sistema	Verificación
5100	Estaciones de Trabajo	El sistema manejará las Estaciones de trabajo.	Estaciones de Trabajo, Visualización	Verificación
5200	Usuario Administrador	El sistema permitirá a un usuario administrador configurar y dar de alta catálogos y configuraciones.	Seguridad	Verificación
5300	Catálogos	El sistema permitirá visualizar los catálogos necesarios.	Visualización	Verificación
5400	Documentos	El sistema permitirá ver y actualizar los documentos configurados.	Visualización , seguridad	Verificación
5500	Graficas Reportes	El módulo de reportes presentara de manera gráfica los reportes solicitados.	Reportes	Verificación
5600	Módulo de reportes Externo	El módulo de reportes utilizara un programa de graficación prediseñado.	Reportes	Acuerdo
5700	Reportes de documentos	El módulo de reportes permitirá reportes sobre todos los documentos configurados.	Reportes	Verificación
5800	Presentación Reportes	El módulo de reportes presentara la información base con la que se obtuvo el reporte.	Reportes	Verificación
5900	Base de Datos Microsoft	La base de datos se desarrollara en SQL SERVER	Base de datos	Acuerdo
6000	Manuales	Se entregaran documentos técnicos sobre la base de datos y la aplicación	Sistema	Documentos técnicos
6100	Consultas	Los accesos de la base de datos se programarán en SQL estándar.	Sistema, base de datos	Acuerdo

Figura 3.6.1 Requerimientos del Sistema de Registro de Ordenes

Diferencia entre Especificación de Requerimientos del Negocio (BRS) y Especificación de Requerimientos de Software (SRS).

En un proyecto de desarrollo de software el BRS es un documento que detalla los requerimientos de un cliente. Este documento contiene la información acerca del negocio y los detalles de los procesos que se pretende automatizar. El SRS es un documento que especifica los requerimientos de software de un sistema. Esto incluye una descripción del sistema que necesita ser desarrollado e incluye información de cómo el usuario interactúa con el sistema y requerimientos no funcionales entre otros.

El documento de BRS

La Especificación de Requerimientos del Negocio es un documento que detalla los requerimientos de un cliente. Este documento será la referencia para el equipo de desarrollo de software y para el equipo de pruebas durante la fase de pruebas. El documento contiene detalles acerca de los procesos que requieren ser automatizados y las nuevas características que se requieran. En general el BRS contiene información como quien será el usuario, el número máximo de usuarios concurrentes que usaran el sistema, los tipos de usuario, computer literacy of the uses, los problemas que dicen tener los usuarios, los niveles de seguridad requeridos por la aplicación, restricciones de hardware y plataforma que por el uso del software se presentaran. También incluye una descripción del sistema o proceso actual y sus posibles expansiones. El BRS también incluye los entregables o lo que espera el cliente de estos además de describir las capacidades esperadas del sistema. Lo más destacado es que el BRS se escribe sin usar tecnicismos.

El documento de SRS

La Especificación de Requerimientos de Software (SRS). Incluye una descripción del sistema que requiere ser desarrollado. Esto incluye como el usuario interactúa con el sistema utilizando casos de uso. Los casos de uso proveen una descripción de las acciones que ocurren entre los usuarios y el sistema. Usualmente UML es usado para especificar formalmente los casos de uso en el SRS. También contiene los requerimientos no funcionales como son los requerimientos de capacidades, estándares requeridos por sistema y cualquier otra restricción del mismo. El SRS debe también ser siempre correcto y consistente porque será usado por los desarrolladores durante el proceso. Y por supuesto tampoco debe ser ambiguo. Generalmente un SRS debe contener al menos las siguientes secciones: Una introducción, una descripción del sistema y requerimientos específicos. La introducción debe definir claramente los alcances del proyecto además de otra información sobre el propósito del sistema y una descripción general del sistema. Las descripciones generales proveen las interacciones con el usuario, las dependencias y las restricciones del sistema. Los requerimientos específicos contienen cualquier requisito de capacidad, bases de datos, etc.

Diferencias entre BRS y SRS

El BRS es un documento que detalla los requerimientos del usuario usando términos no técnicos mientras que el SRS especifica los requerimientos del software de una manera formal. El SRS describe como el usuario interactúa con el sistema usando casos de uso mientras el BRS provee una descripción de las interacciones del usuario. Ambos BRS y SRS son usados para los procesos de desarrollo y pruebas.

4. *DISEÑO Y CONSTRUCCIÓN DE LA APLICACIÓN*

1. Aplicación de la metodología elegida

Desarrollo de Sistemas

El desarrollo de sistemas se ha trasformado al igual que las empresas.

El entorno de trabajo de las empresas ha cambiado. Antes los productos y servicios salían al mercado hasta estar cien por ciento probados en todas sus características, y el análisis y diseño de estos requería de estas consideraciones. Hoy las necesidades han cambiado y muchas empresas lanzan producto y servicios no terminados y su funcionalidad y formas de crearlos no están bien definidas, los tiempos son cada vez más cortos y la adaptación es lo único fijo.

Formalidad y Metodología

La formalidad y la metodología siguen garantizando los mismos conceptos por los que fueron creados.

1. Permiten la toma de decisiones
2. Se trabaja en forma ordenada y consistente
3. Determinan con precisión las actividades y tareas
4. Permiten la unificación de los trabajos
5. Permiten la comunicación entre organizaciones.

Esto permite entre otras cosas que:

- Los requerimientos sean conocidos y acotados
- La comunicación con el usuario sea tomada en cuenta.
- Se modele de acuerdo a la capacidad y el ambiente.
- Se elija la mejor herramienta.
- Se realicen pruebas específicas.
- Tener documentación entendible.

Sus premisas siguen siendo válidas considerando el enfoque anterior.

“Todo sistema, a ser desarrollado, tiene su origen en una necesidad real.”
Esto implica la realización de un diagnóstico previo.

Se pueden resumir algunas características comunes de los métodos formales

Obtener las ventajas de los métodos formales es necesario seguir una serie de pasos sistemáticos para que los diferentes grupos de desarrollo posean una buena comunicación. Estos pasos son brindados por los modelos de ciclo de vida, los cuales están constituidos por diferentes etapas:

Especificación de requerimientos: Se realizan entrevistas con el usuario identificando los requerimientos y necesidades del usuario.

Análisis: Modela los requerimientos del usuario.

Diseño: Se modela la solución del sistema, teniendo en cuenta el ambiente de implementación a utilizar, por ejemplo, si el sistema es centralizado o distribuido, la base de datos a utilizar, lenguaje de programación, performance deseada, etc.

Implementación: Dado el lenguaje de programación elegido se implementa el sistema.

Testeo: En esta etapa se verifica y valida el sistema teniendo en cuenta algunos criterios determinados por el grupo correspondiente.

Mantenimiento: Es la etapa más difícil de desarrollo del sistema, actualiza y modifica el sistema si surgen nuevos requerimientos.
Existen varios métodos para describir el ciclo de vida de un sistema, uno de ellos es el desarrollo estructurado en cascada (Figura 4.1.1).

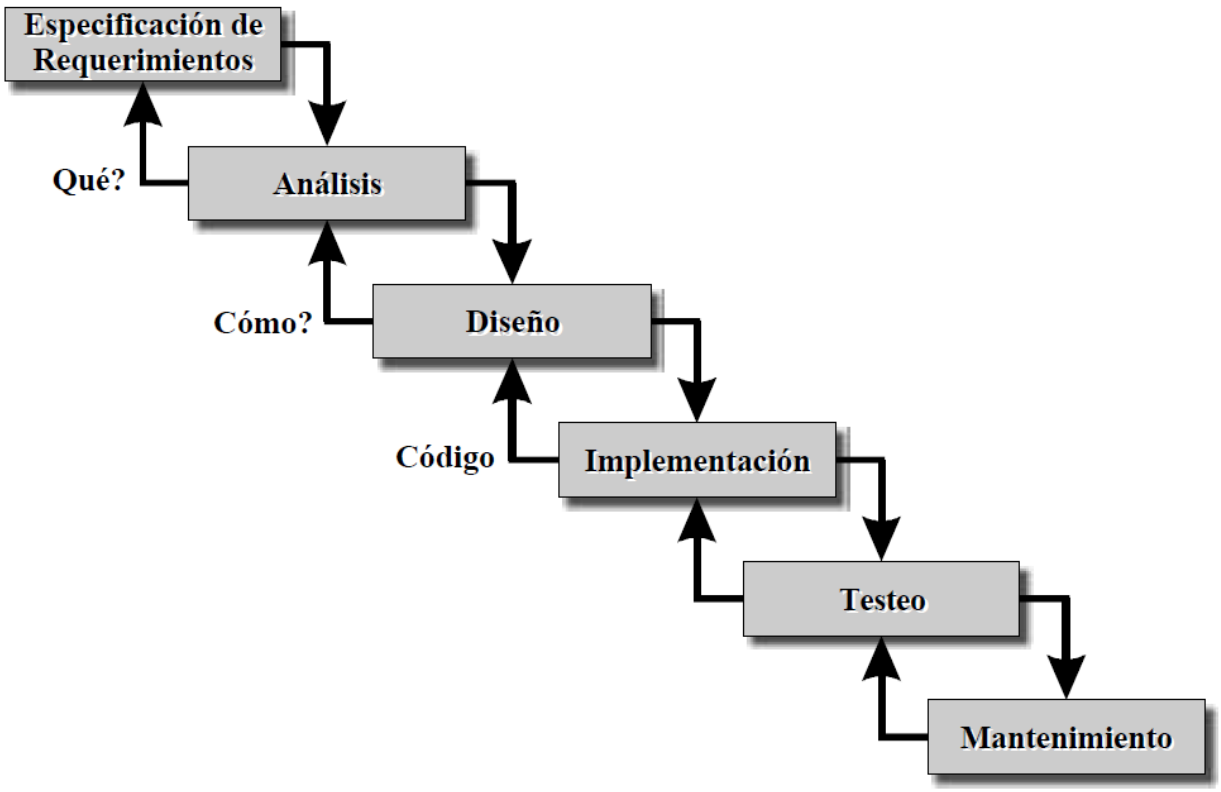


Figura 4.1.1 Modelo de Ciclo de Vida en Cascada

En un principio fue de gran utilidad pero el problema es que para pasar de una etapa a la otra había que terminar la primera, produciendo un gran problema si algún cambio era requerido. La etapa de Mantenimiento consumía el 80% del costo de producción.
Debido a los nuevos requerimientos en el desarrollo de software, surgieron muchos otros modelos que trataban de solucionar los problemas existentes, que se basaron en el modelo en Cascada.
Por ejemplo, el Modelo en Espiral, en el cual el sistema se desarrolla incrementalmente (Fig. 2).

Los modelos propuestos poseen básicamente las mismas etapas, pero varían en:

- Los métodos y herramientas utilizadas en cada actividad
- Los controles requeridos, paralelismo en las actividades
- Las salidas de cada etapa

No es aconsejable elegir un modelo y seguirlo al detalle sino que se debe adaptar a las características del proyecto que está siendo desarrollado.
Los métodos de desarrollo de software pueden dividirse en dos grupos: *función/dato* y *orientados a objetos*.

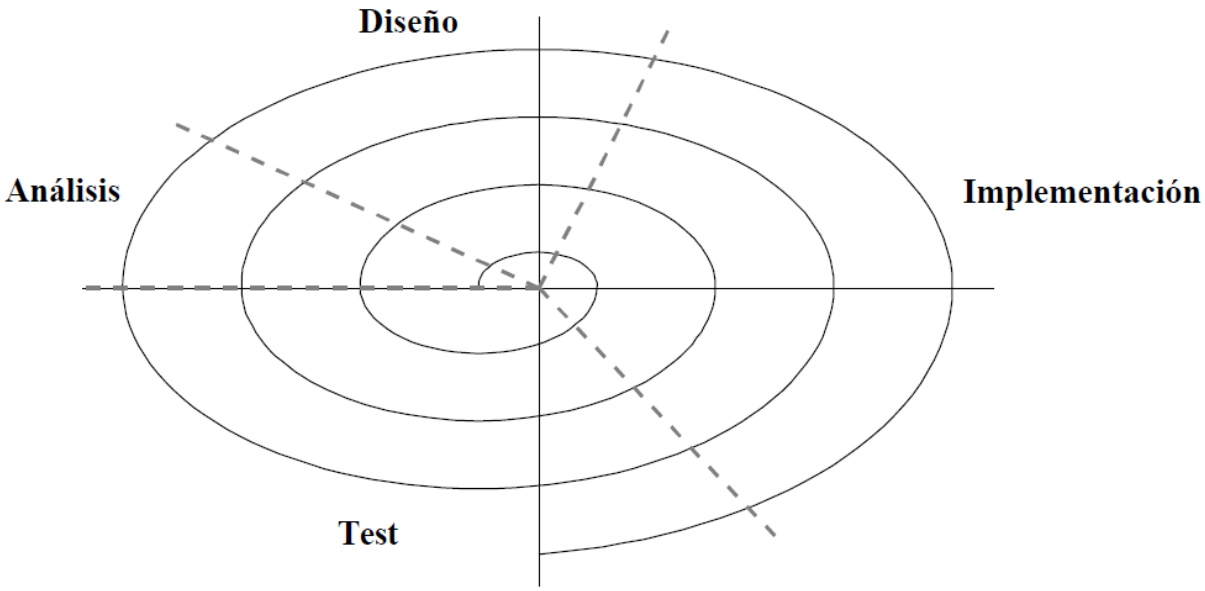


Figura 4.1.2 Modelo de Ciclo de Vida en Espiral

Orientado a Función/Dato	Orientado a Objetos
Énfasis en la transformación de datos.	Énfasis en la abstracción de datos.
Funciones y datos tratados como entidades separadas.	Funciones y datos encapsulados en entidades fuertemente relacionadas.
Difícil de entender y modificar.	Facilidades de mantenimiento.
Funciones, usualmente, dependientes de la estructura de los datos.	Mapeo directo a entidades del mundo real.

Figura 4.1.3 Métodos de desarrollo de software.

Orientado a Función/Dato: Aquellos métodos en los cuales las funciones y/o los datos son tratados como entidades independientes. Estos sistemas resultan difíciles de mantener. El mayor problema es que las funciones generalmente dependen de la estructura de los datos. A menudo diferentes tipos de datos tienen distintos formatos y se necesita verificar el tipo del dato (con sentencias If-Then o CASE), produciendo programas difíciles de leer y modificar. Si se desea hacer alguna modificación en la estructura de los datos se debe modificar en todos los lugares donde es utilizado.
Otro problema es que una persona no piensa naturalmente en términos de una estructura. La especificación de requerimientos se hace en lenguaje común, se especifica la funcionalidad que debe tener el sistema y no en cómo se deben estructurar los datos.

Orientado a Objetos: Son aquellos métodos en los cuales datos y funciones están altamente relacionados. El énfasis está centrado en la abstracción de datos. Se piensa en forma natural, los

objetos son mapeados a entidades del mundo real. Los programas son de fácil mantenimiento y extensibles por medio de la construcción de subclases.
Varios métodos de desarrollo de software han sido propuestos para cada uno de estos grupos, algunos de los cuales son descriptos en la Figura 4.1.5. Dónde:

Siglas	Nombre	Autor
RAD	Rapid Application Development	James Maslow 80
SADT	Structured Analysis and Design Technique	Ross85
RDD	Requirement Driven Design	Alford85
SA/SD	Structured Analysis and Structured Design	Yourdon&Constantine79
OOSE	Object-Oriented Software Engineering	Jacobson94
OOA	Object-Oriented Analysis	Goldberg
OMT	Object Modeling Technique	Rumbaugh93
UP	Unified Process	Booch&Jacobson&Rumbaugh98
Catalysis	Catálisis	D'Souza98

Figura 4.1.4 Métodos de desarrollo de software Siglas.

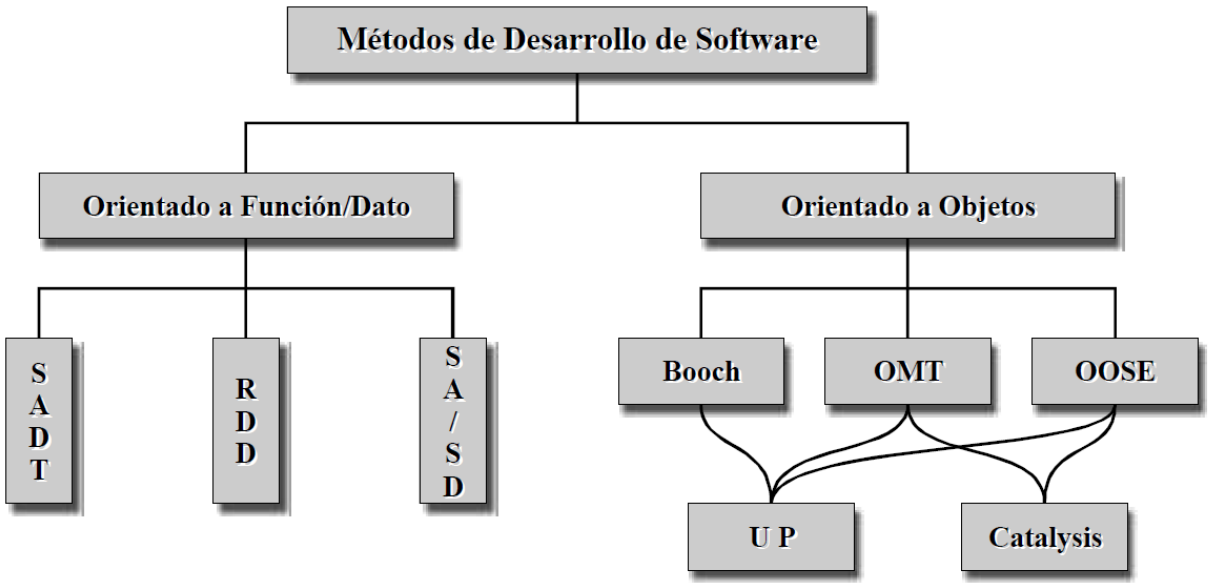


Figura 4.1.5 Métodos de desarrollo de software grupos.

Según establecen los críticos de los nuevos métodos simples estos diagnósticos predictivos no son del todo adecuados para el desarrollo actual de productos y servicios.

Los Nuevos Métodos

Evidentemente no puede negarse los beneficios de las metodologías formales predictivas, pero algunas bases que dieron origen a estos métodos están cambiando.

En las metodologías actuales se construye mientras aparecen nuevos requisitos, los modelos se adaptan a la velocidad del sector del negocio y por tanto no permite conocer a detalle cómo será el resultado final.

Lo que existe ahora son productos en mejora y evolución continuos

Algunas de las nuevas consideraciones que plantean estos modelos son:

- ¿La gestión de proyectos predictiva es la única posible?
- ¿Los criterios para determinar el éxito son siempre el cumplimiento de fechas y costos?
- ¿Puede haber proyectos cuya gestión no busque realizar un trabajo previamente planificado, con un presupuesto y en un tiempo previamente calculado?

Hoy existen directores de producto que no necesitan conocer cuáles van a ser las 200 funcionalidades que tendrá el producto final, ni si este estará terminado en 12 o en 16 meses.

Hay clientes que necesitan disponer de una primera versión con funcionalidades básicas en cuestión de semanas, y no un producto completo dentro de uno o dos años. Clientes cuyo interés es poner en el mercado rápidamente un concepto nuevo, y desarrollar de forma continua su valor.

Hay proyectos que no necesitan gestionar el seguimiento de un plan, y que fracasan por haber empleado un modelo de gestión inapropiado.

La mayoría de fracasos se producen por aplicar ingeniería secuencial y gestión predictiva tanto en el proceso de adquisición (requisitos, contratación, seguimiento y entrega) como en la gestión del proyecto, en productos que no necesitan tanto garantías de previsibilidad en la ejecución, como respuesta rápida y flexibilidad para funcionar en entornos de negocio que cambian y evolucionan rápidamente

Las nuevas metodologías basadas en procesos surgieron como alternativa a los métodos formales que se consideran excesivamente pesados

Hasta 2005, entre los defensores de los modelos de procesos y los de modelos ágiles fueron frecuentes las posturas radicales, más ocupadas en descalificar al otro, que en estudiar sus métodos y conocerlos para mejorar los propios.

El manifiesto de los “Métodos Agiles” muestra mejor estos conceptos.

Manifiesto Ágil

Estamos poniendo al descubierto mejores métodos para desarrollar software, haciéndolo y ayudando a otros a que lo hagan.

Con este trabajo hemos llegado a valorar:

- *A los individuos y su interacción, por encima de los procesos y las herramientas.*
- *El software que funciona, por encima de la documentación exhaustiva.*
- *La colaboración con el cliente, por encima de la negociación contractual.*
- *La respuesta al cambio, por encima del seguimiento de un plan.*

Aunque hay valor en los elementos de la derecha, valoramos más los de la izquierda.

Figura 4.1.6 Manifiesto Ágil.

En forma específica:

Valoramos más a los individuos y su interacción que a los procesos y las herramientas.

Este es el valor más importante del manifiesto.

Por supuesto que los procesos ayudan al trabajo. Son una guía de operación. Las herramientas mejoran la eficiencia, pero hay tareas que requieren talento y necesitan personas que lo aporten y trabajen con una actitud adecuada.

La producción basada en procesos persigue que la calidad del resultado sea consecuencia del know-how "explicitado" en los procesos, más que en el conocimiento aportado por las personas que los ejecutan.

Sin embargo en desarrollo ágil los procesos son una ayuda. Un soporte para guiar el trabajo. La defensa a ultranza de los procesos lleva a afirmar que con ellos se pueden conseguir resultados extraordinarios con personas mediocres, y lo cierto es que este principio no es cierto cuando se necesita creatividad e innovación.

Valoramos más el software que funciona que la documentación exhaustiva.

Poder anticipar cómo será el funcionamiento del producto final, observando prototipos previos, o partes ya elaboradas ofrece un "feedback" estimulante y enriquecedor, que genera ideas imposibles de concebir en un primer momento, y difícilmente se podrían incluir al redactar un documento de requisitos detallado en el comienzo del proyecto.

El manifiesto ágil no da por inútil la documentación, sólo la de la documentación innecesaria. Los documentos son soporte de hechos, permiten la transferencia del conocimiento, registran información histórica, y en muchas cuestiones legales o normativas son obligatorios, pero su relevancia debe ser mucho menor que el producto final.

La comunicación a través de documentos no ofrece la riqueza y generación de valor que logra la comunicación directa entre las personas, y a través de la interacción con prototipos del producto.

Por eso, siempre que sea posible debe preferirse reducir al mínimo indispensable el uso de documentación, que requiere trabajo sin aportar un valor directo al producto.

Si la organización y los equipos se comunican a través de documentos, además de ocultar la riqueza de la interacción con el producto, forman barreras de burocracia entre departamentos o entre personas.

Valoramos más la colaboración con el cliente que la negociación contractual.

Las prácticas ágiles están indicadas para productos cuyo detalle resulta difícil prever al principio del proyecto; y si se detallara al comenzar, el resultado final tendría menos valor que si se mejoran y precisan con retroinformación continua durante el.

También son apropiadas cuando se prevén requisitos inestables por la velocidad de cambio en el entorno de negocio del cliente.

El objetivo de un proyecto ágil no es controlar la ejecución conforme a procesos y cumplimiento de planes, sino proporcionar el mayor valor posible al producto.

Resulta por tanto más adecuada una relación de implicación y colaboración continua con el cliente, más que una contractual de delimitación de responsabilidades.

Valoramos más la respuesta al cambio que el seguimiento de un plan

Para desarrollar productos de requisitos inestables, que tienen como factor inherente el cambio y la evolución rápida y continua, resulta mucho más valiosa la capacidad de respuesta que la de seguimiento y aseguramiento de planes. Los principales valores de la gestión ágil son la anticipación y la adaptación, diferentes a los de la gestión de proyectos ortodoxa: planificación y control que evite desviaciones del plan.

Los 12 principios del "Manifiesto Ágil"

El manifiesto ágil, tras los postulados de estos cuatro valores en los que se fundamenta, establece estos 12 principios:

1. Nuestra principal prioridad es satisfacer al cliente a través de la entrega temprana y continua de software de valor.
2. Son bienvenidos los requisitos cambiantes, incluso si llegan tarde al desarrollo. Los procesos ágiles se doblegan al cambio como ventaja competitiva para el cliente.
3. Entregar con frecuencia software que funcione, en periodos de un par de semanas hasta un par de meses, con preferencia en los periodos breves.
4. Las personas del negocio y los desarrolladores deben trabajar juntos de forma cotidiana a través del proyecto.
5. Construcción de proyectos en torno a individuos motivados, dándoles la oportunidad y el respaldo que necesitan y procurándoles confianza para que realicen la tarea.
6. La forma más eficiente y efectiva de comunicar información de ida y vuelta dentro de un equipo de desarrollo es mediante la conversación cara a cara.
7. El software que funciona es la principal medida del progreso.
8. Los procesos ágiles promueven el desarrollo sostenido. Los patrocinadores, desarrolladores y usuarios deben mantener un ritmo constante de forma indefinida.
9. La atención continua a la excelencia técnica enaltece la agilidad.
10. La simplicidad como arte de maximizar la cantidad de trabajo que se hace, es esencial.
11. Las mejores arquitecturas, requisitos y diseños emergen de equipos que se auto organizan.
12. En intervalos regulares, el equipo reflexiona sobre la forma de ser más efectivo y ajusta su conducta en consecuencia.

Los métodos como **Scrum AM**, *eXtreme Programming* **XP**, *Rational Unified Process* **RUP**, entre otros, son los nuevos métodos y herramientas que se utilizan para el desarrollo de aplicaciones en las industrias especializadas.

Una breve definición de los diferentes métodos, entornos y herramientas se pueden ver en el anexo Desarrollo de software.

Yourdon

La metodología elegida es la propuesta por Edward Yourdon en 1988 debido a su amplia difusión y conocimiento y a que presenta una amplia documentación para el entendimiento del sistema y los beneficios anteriormente expuestos sobre las metodologías formales, pretendiendo con esto que el sistema sea comprendido y explicado en estos términos.

La metodología de Yourdon permite que los modelos tengan las siguientes cualidades:

- **Gráfica**, para su comprensión.
- **Concisa**, breve
- **Top-Down**, desarrollo secuencial de arriba abajo.
- **Particionada**, Piezas independientes y pequeñas.
- **No redundante**, cada pieza de información será referida y definida una vez.
- **Esencial**, que debe hacer el sistema.

Estructuralmente la metodología se puede describir como en la figura 4.1.7

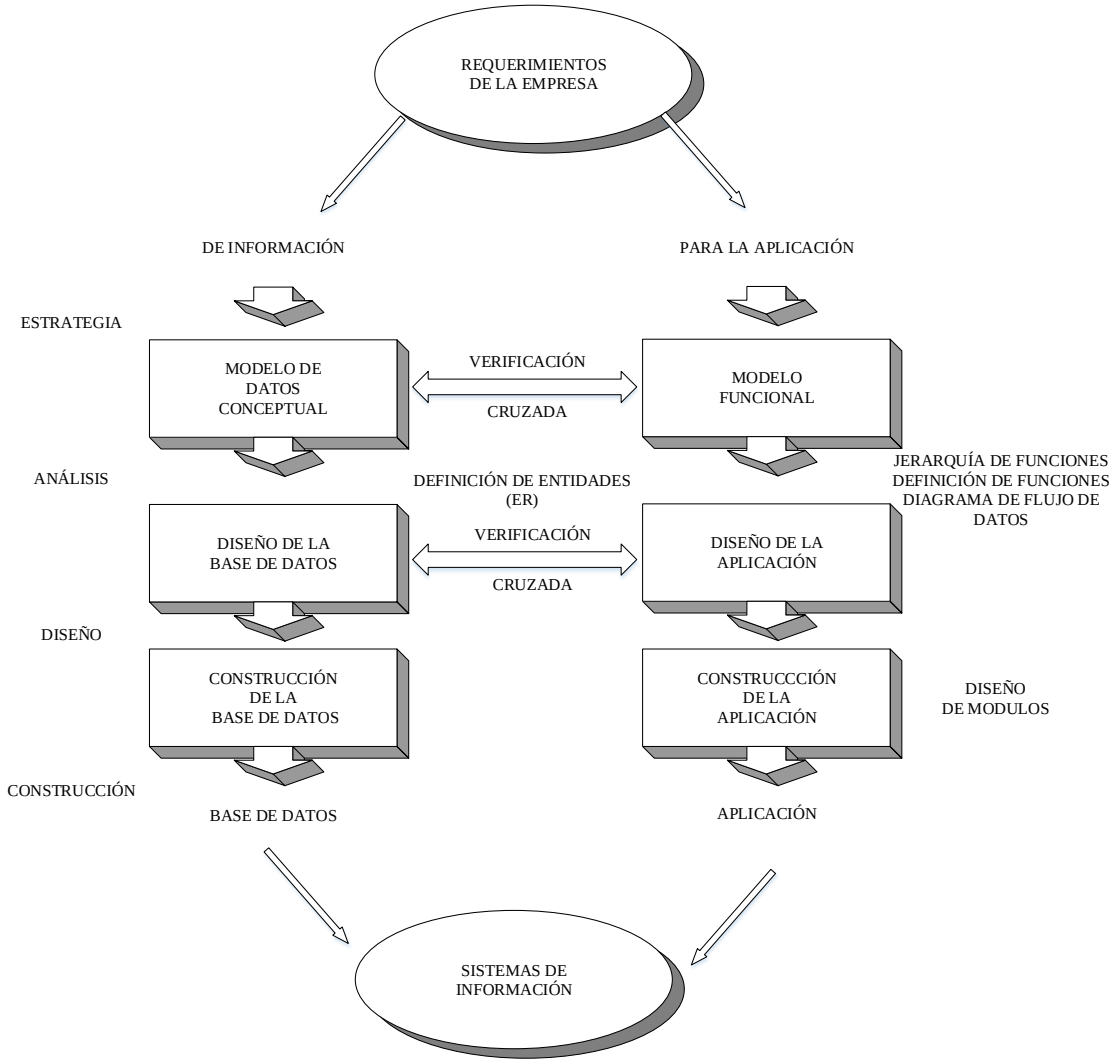


Figura 4.1.7 Diseño Estructural Yourdon

Donde se puede observar los trabajos de un proyecto:

De acuerdo a la información, en forma general, el modelo de datos conceptual, el diseño de la base de datos y la construcción de la base de datos
De acuerdo con la aplicación el modelo funcional, el diseño de la aplicación y la construcción de la aplicación
Mostrando su relación en las etapas de Estrategia, Análisis, Diseño y Construcción y mostrando el uso de algunas herramientas como jerarquía de funciones definición de funciones, diagrama de flujo, entre otros..

Yourdon utiliza las siguientes herramientas para la construcción de un sistema:

- Diagrama de Contexto
- Diagrama de Flujo
- Diccionario de Datos
- Diagrama Entidad Relación

Como se muestra en la figura 4.1.8

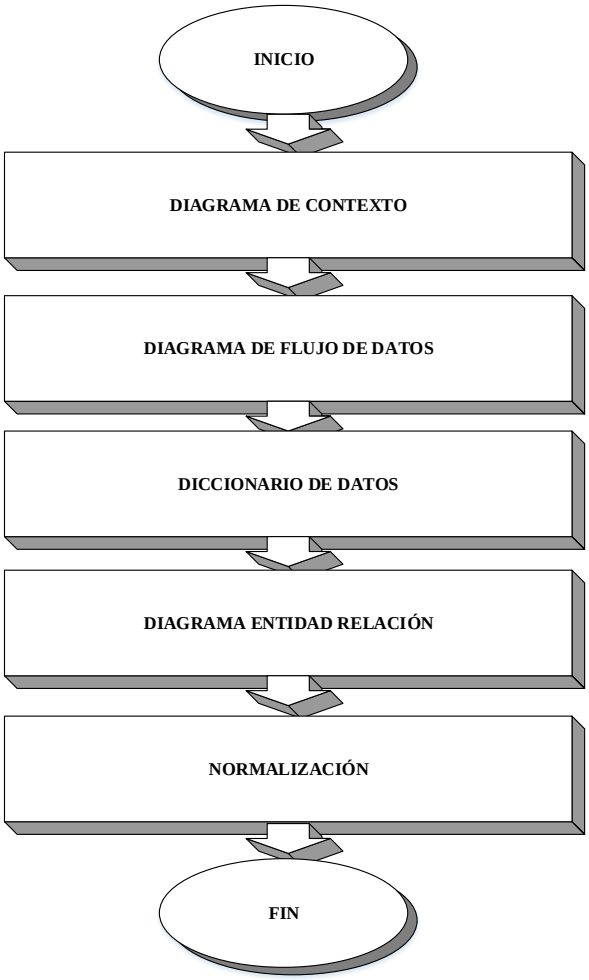


Figura 4.1.8 Herramientas Yourdon

2. Diagrama de Contexto

En el diagrama de la figura 4.2.1 se muestran los elementos de esta metodología.

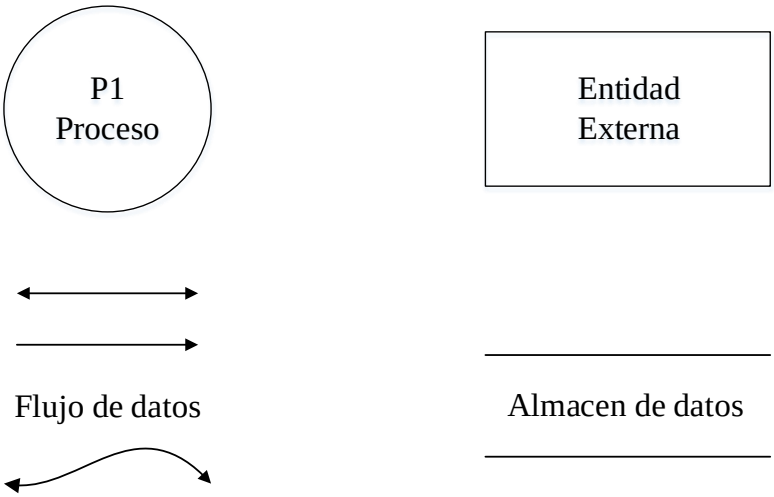


Figura 4.2.1 Elementos de diagramas

A continuación se presentaran los diagramas de contexto de los principales procesos de la aplicación.

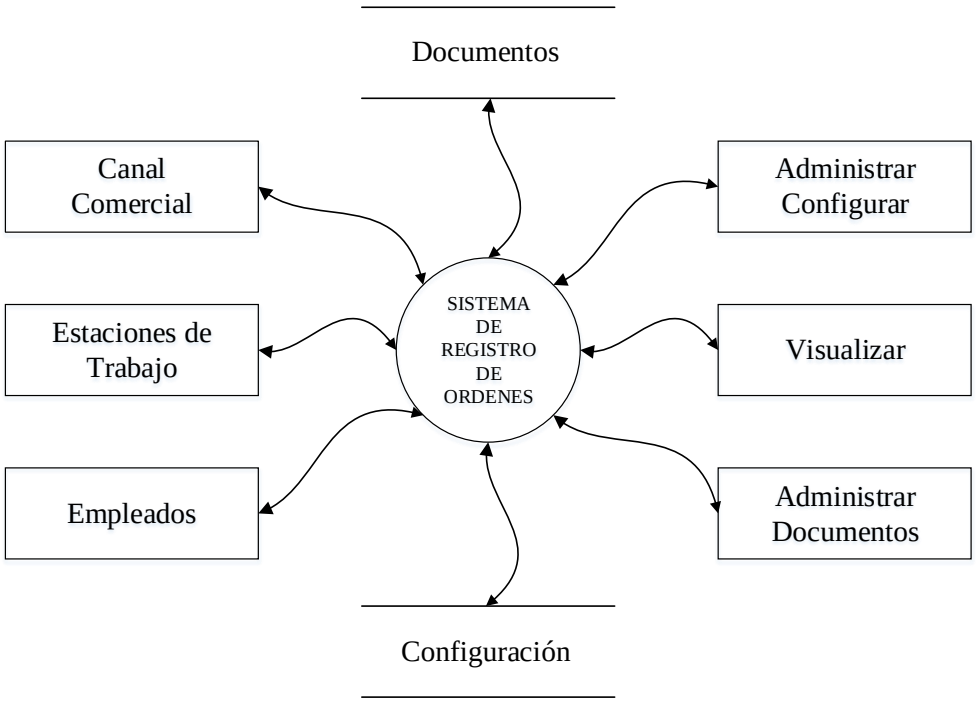


Figura 4.2.2 Diagrama de proceso

Diagrama de contexto principal
Se muestra el proceso general del sistema actores y las funciones principales de la aplicación para su funcionamiento donde: (figura4.2.2)

Canal Comercial
Se refiere a las empresas o empleados de ellas, que consultan información del sistema. Están clasificados para permitir el acceso restringido al mismo.

Empleados
Se refiere al personal de la empresa empresas, que manejan la información del sistema. Están clasificados para permitir el acceso restringido al mismo.

Estaciones de trabajo
Son estaciones lógicas de trabajo que agrupan funciones para definir roles de trabajo.

Administrar y configurar
Son acciones que se llevan a cabo para dar de alta los diferentes catálogos de configuración para el control de toda la aplicación.

Visualizar
Se refiere a las acciones necesarias para visualizar los datos o consultas.

Administrar Documentos
Son acciones que se llevan a cabo para dar de alta los diferentes documentos generados para el control de las ordenes de servicio y controlar como son tratados los datos y estatus de los mismos.

Documentos
Base de datos donde se almacenan los distintos documentos.

Configuración
Base de datos donde se almacenan los parámetros y procesos de la aplicación.

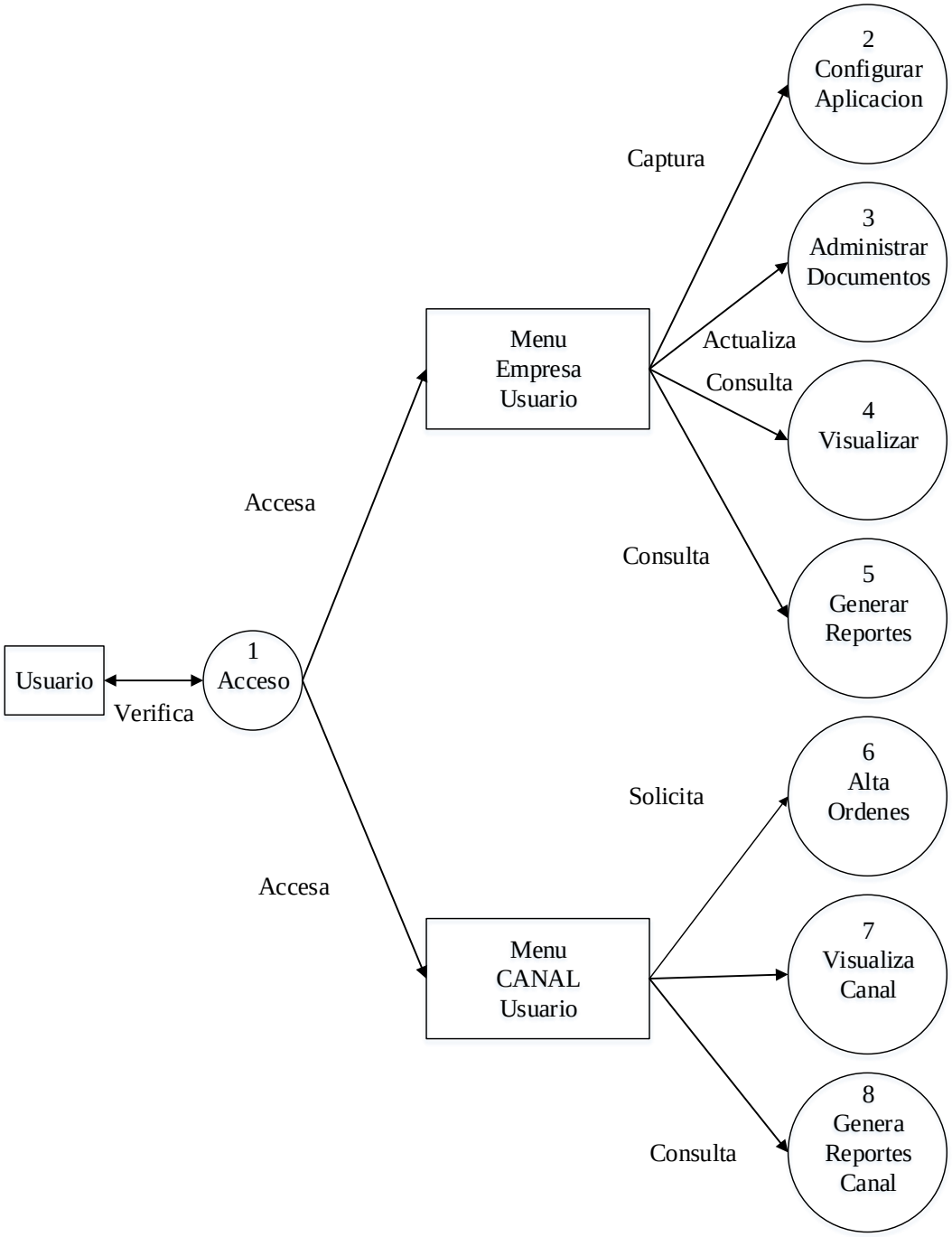


Figura 4.2.3 Diagrama de proceso nivel 1

Diagrama de contexto principal nivel 1
Se describen las funciones que componen el proceso general del sistema. Figura 4.2.3

Se observa los dos tipos de usuario y las funciones que puede realizar cada uno, de acuerdo al rol establecido.

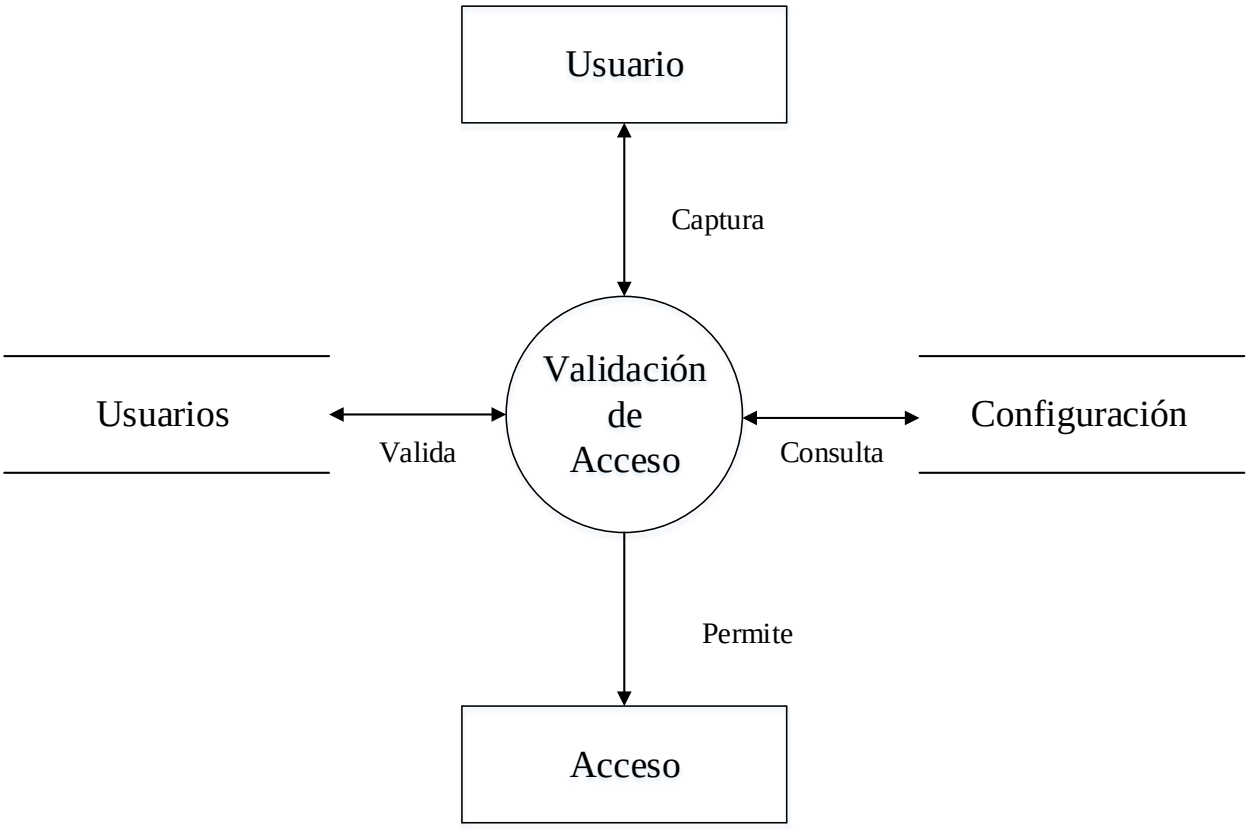


Figura 4.2.4 1 Acceso

1 Acceso

En la figura 4.2.4. Se observa la función de validación de acceso que permite verificar con la base de datos, usuarios y configuración, los roles y las claves de acceso y de acuerdo con esto permite el acceso al sistema

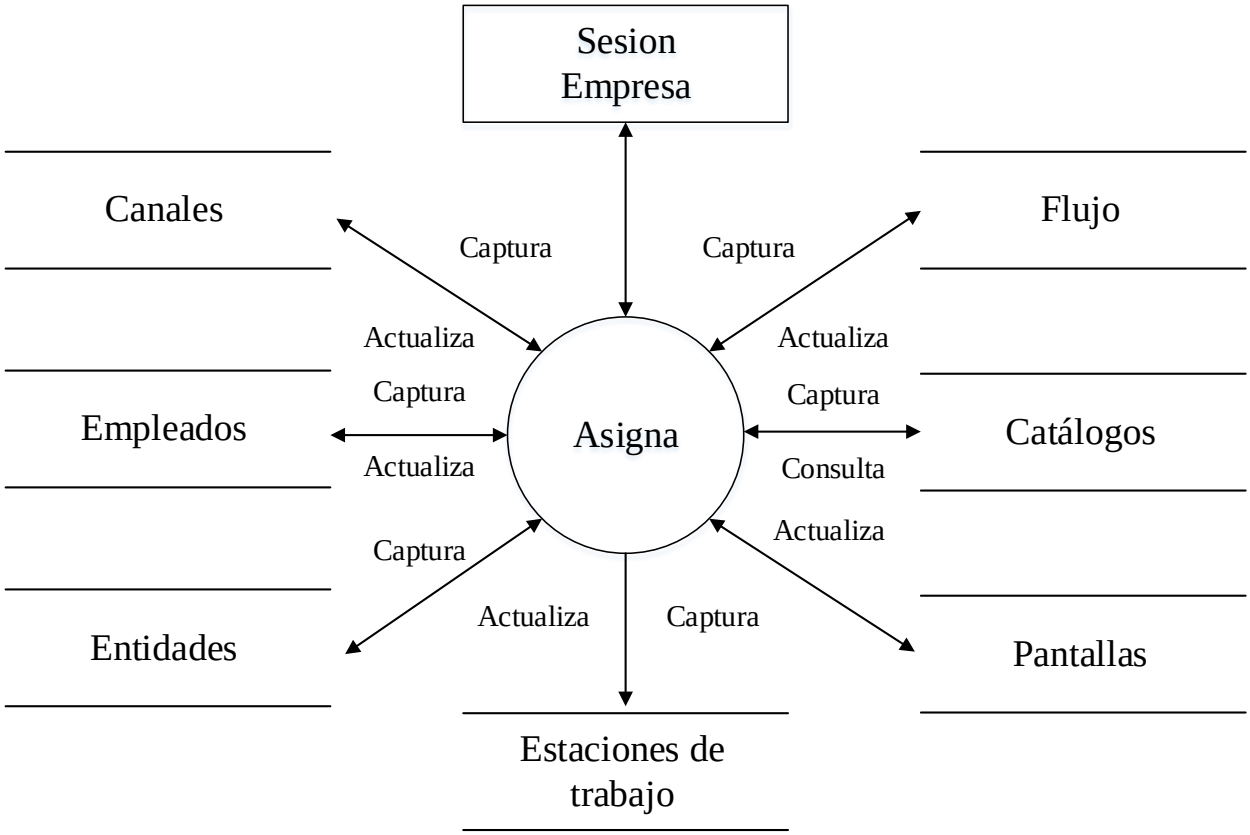


Figura 4.2.5 2 Configurar Administrar nivel 2

2 Configurar Administrar nivel 2

En la figura 4.2.5. Se observa la función de Administrar y configuración que permite establecer los parámetros iniciales del sistema o modificarlos para su funcionamiento inicial considerando los requerimientos iniciales, esta función solo la puede realizar el usuario con derechos de administración de la empresa.

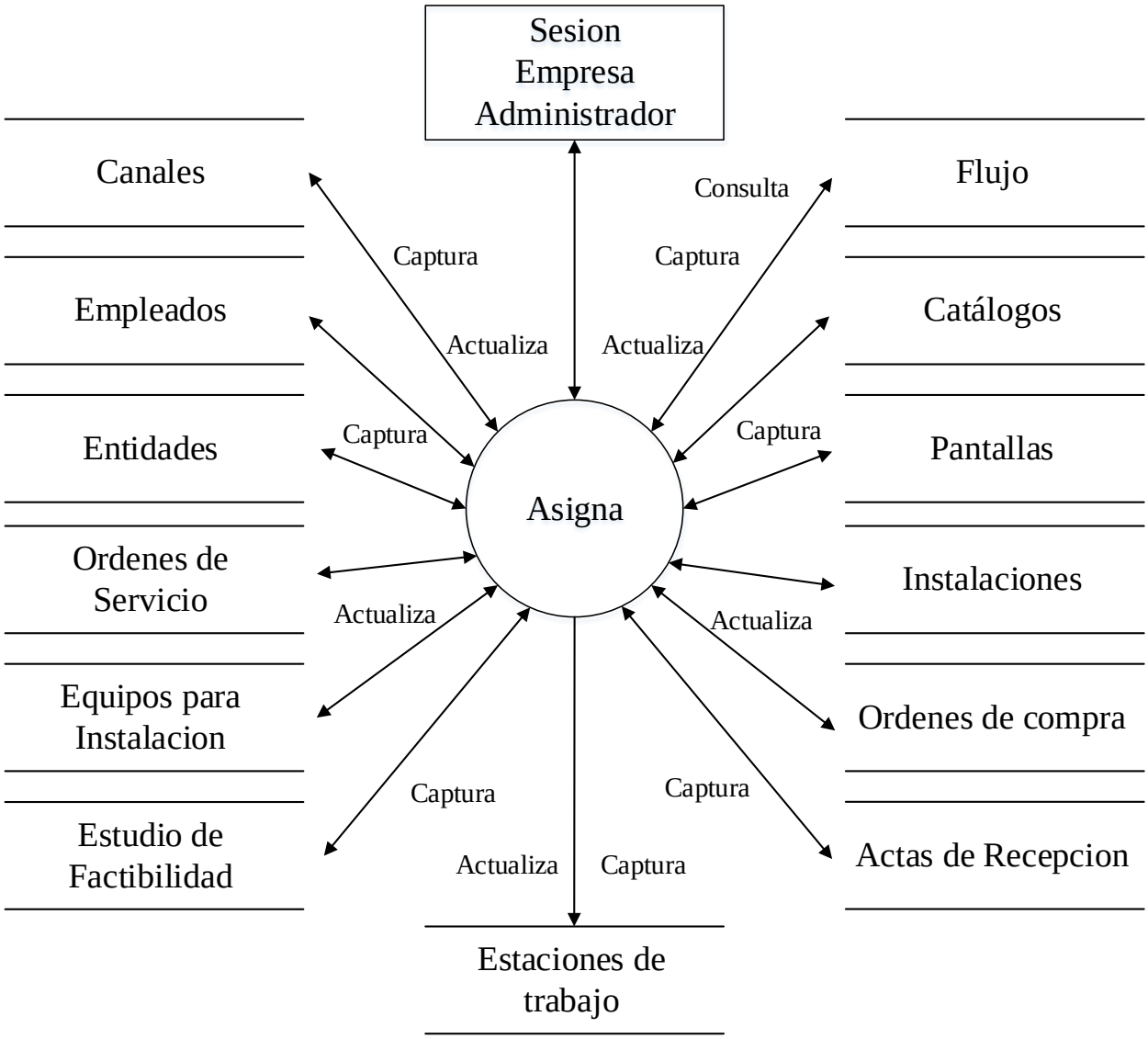


Figura 4.2.6 2 Configurar Administrar nivel 3

2 Configurar Administrar nivel 3

En la figura 4.2.6. Se observa la función de Administrar y configuración que permite establecer los parámetros iniciales del sistema o modificarlos para su funcionamiento inicial es necesario configurar cada uno de los documentos que se ocuparan, el flujo principal, catálogos, pantallas.

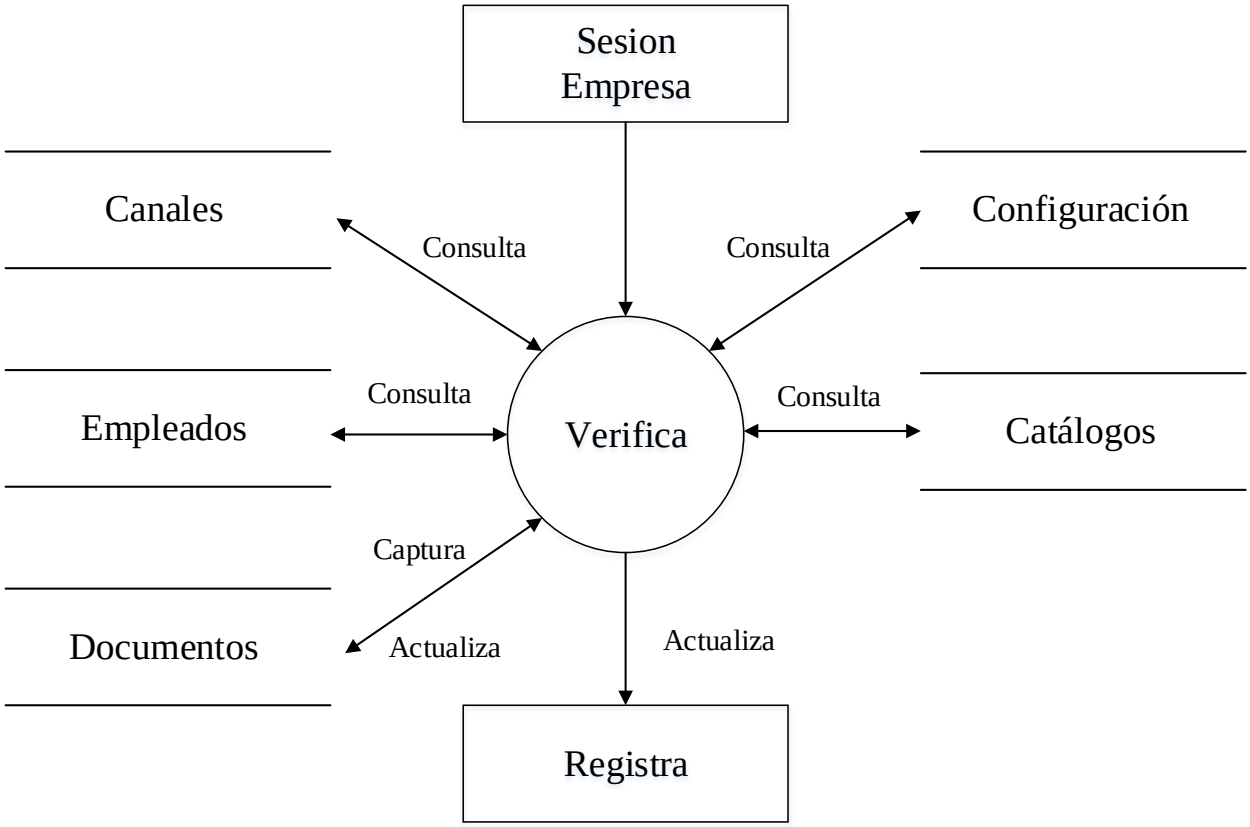


Figura 4.2.7 3 Administración de Documentos 2

3 Administración de Documentos nivel 2

En la figura 4.2.7. Se observa la función de Administrar documentos que permite modificar, dar de alta documentos y estatus o nuevos empleados de acuerdo a los parámetros establecidos. Esta función solo la puede realizar el usuario con derechos de la empresa.

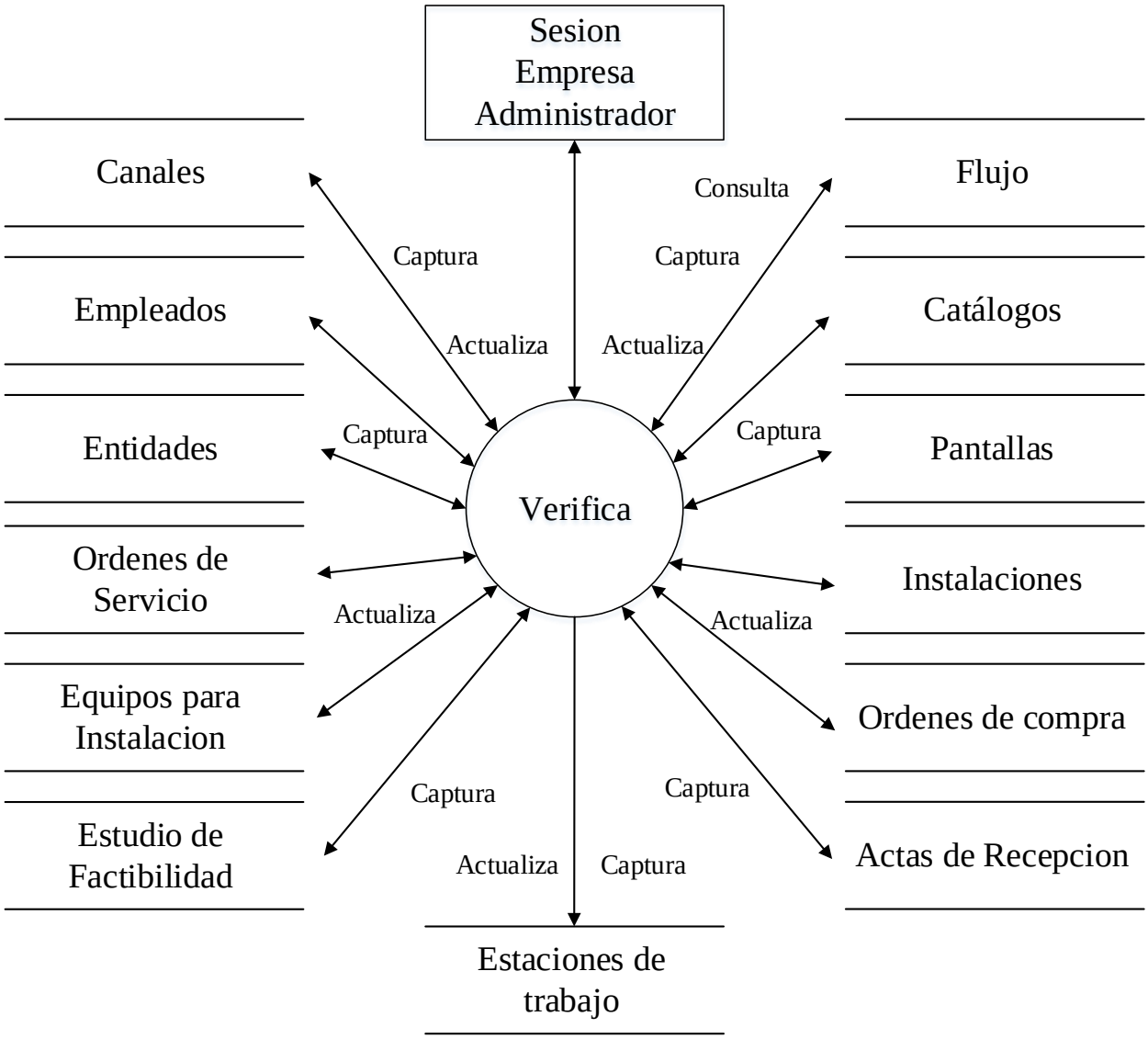


Figura 4.2.8 3 Administración de Documentos nivel 3

3 Administración de Documentos nivel 3

En la figura 4.2.8. Se observa la función de Administrar documentos que permite modificar, dar de alta Órdenes de Servicio, Equipos de Instalación, Actas de Recepción y modificar su estatus para mostrar el avance de los procesos.

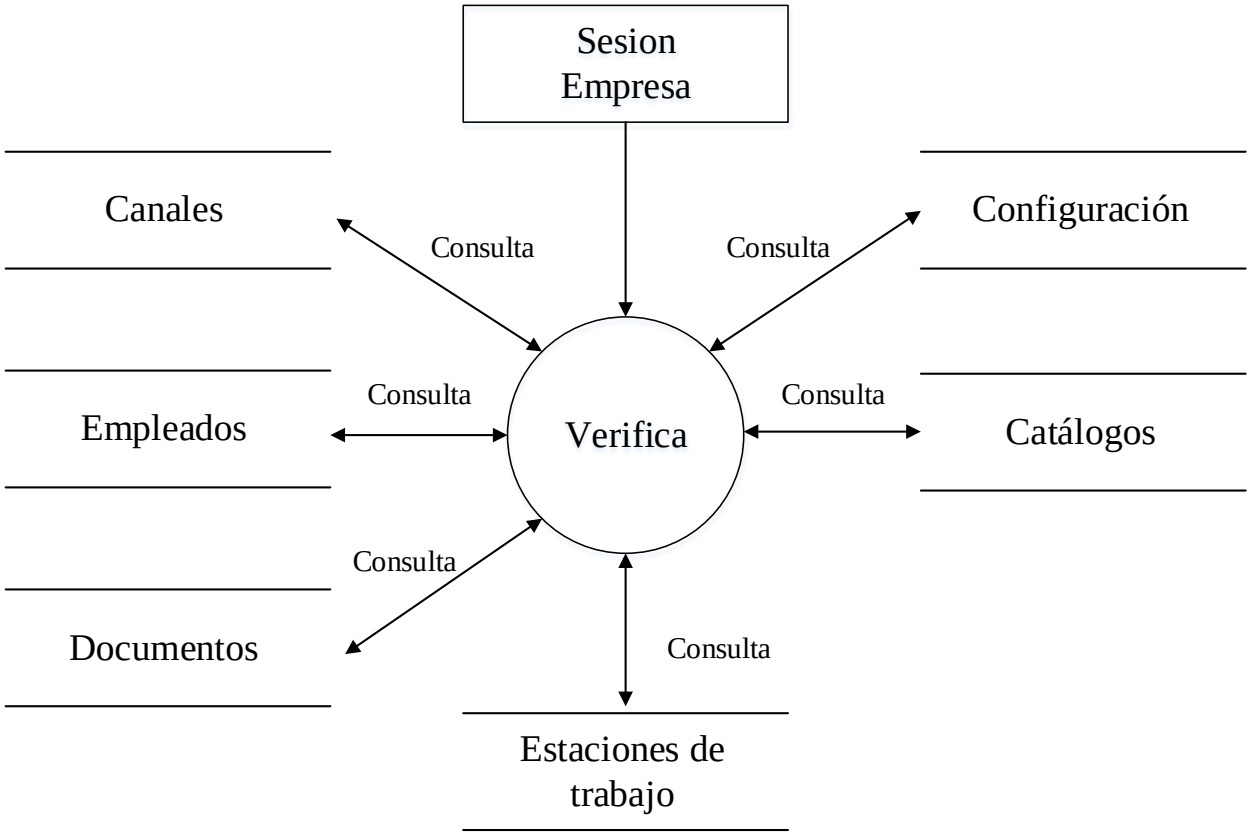


Figura 4.2.9 4 Visualiza nivel 2

4 Visualiza nivel 2

En la figura 4.2.9. Se observa la función de Visualización que permite la consulta de los documentos y catálogos del sistema. Esta función solo la puede realizar el usuario con derechos de la empresa.

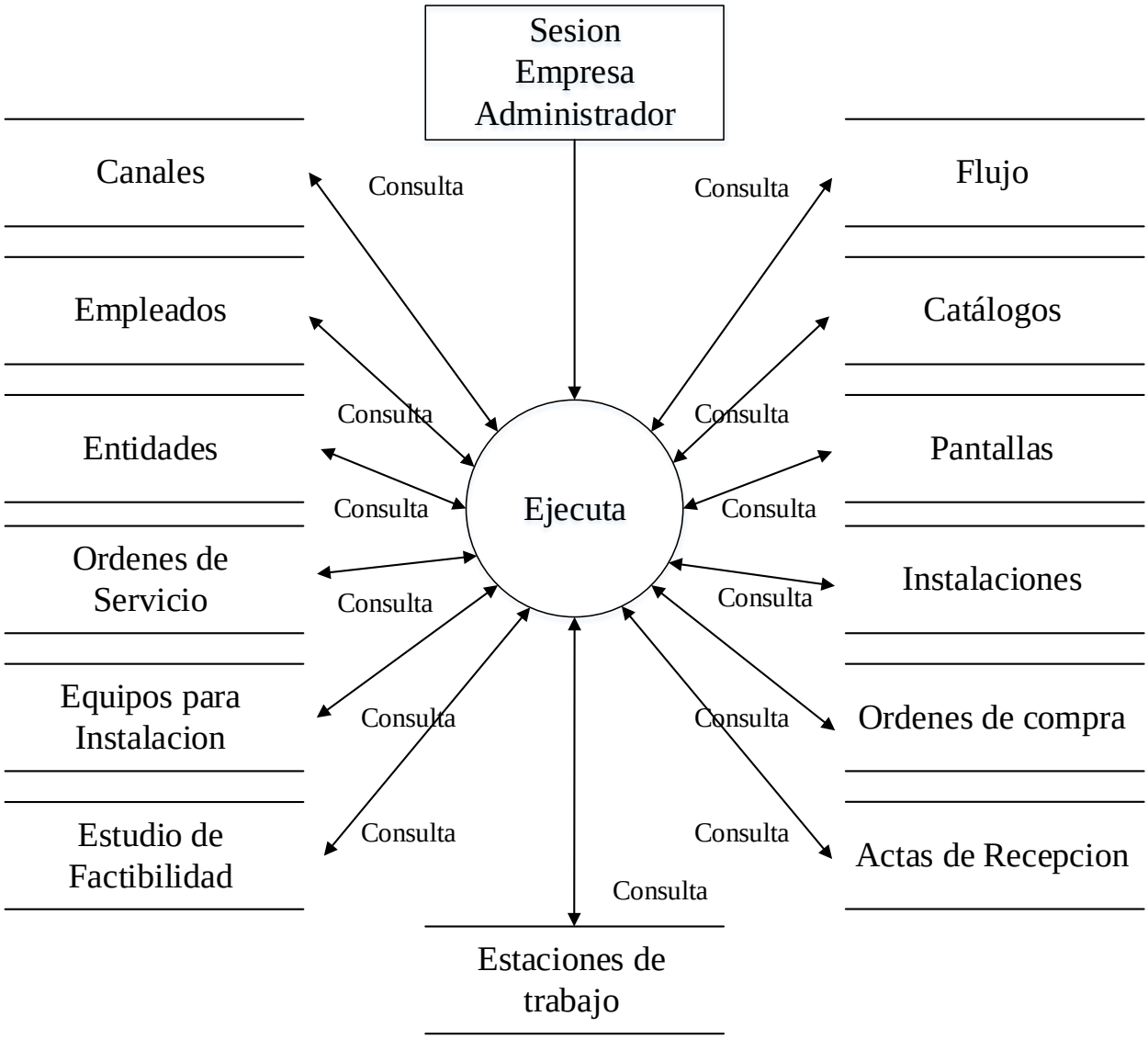


Figura 4.2.10 4 Visualiza nivel 3

4 Visualiza nivel 3

En la figura 4.2.10. Se observa la función de Visualización que permite la consulta de los documentos de por ejemplo Órdenes de servicio, Equipos de instalación y catálogos como Empleado y entidades del sistema. Esta función es la que permite la búsqueda y consulta. La función solo la puede realizar el usuario con derechos de la empresa de acuerdo a esto podrá o no tener acceso a cada uno de los documentos.

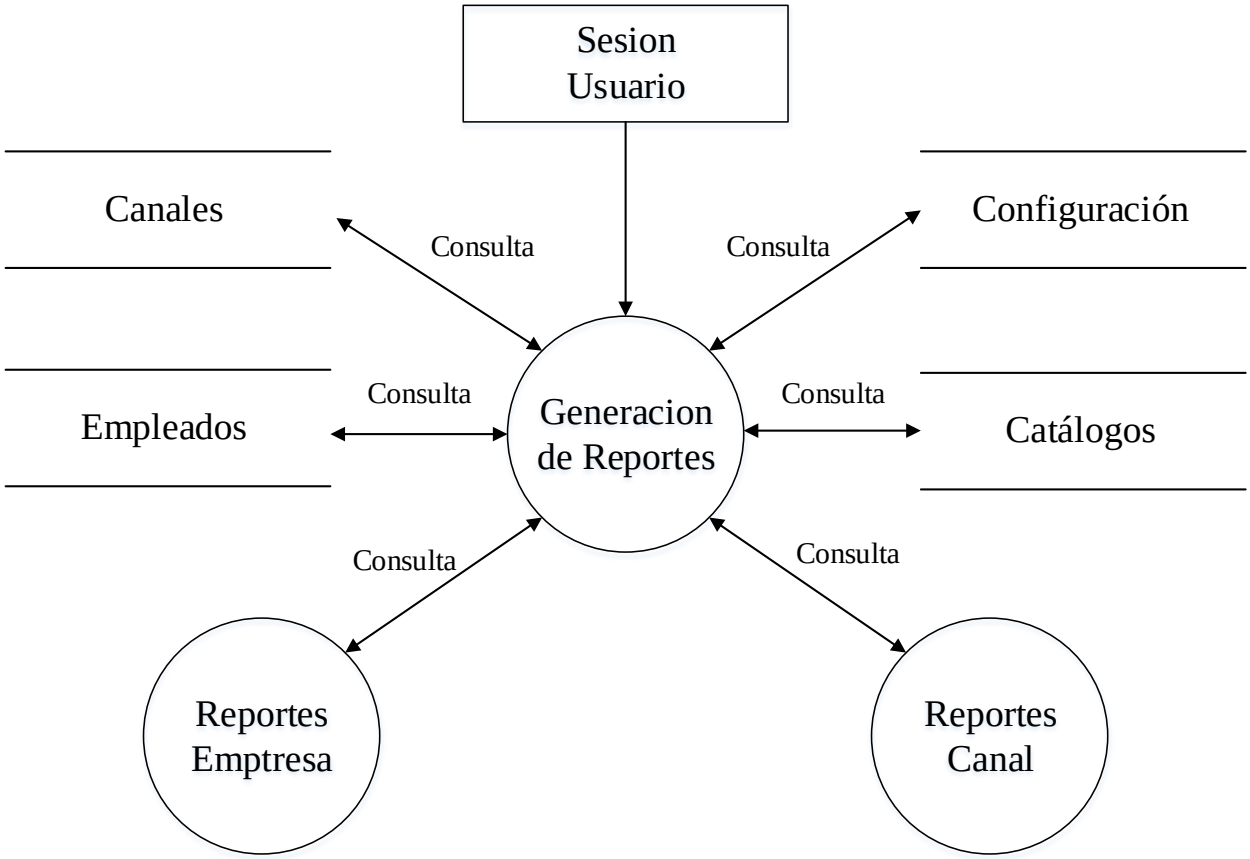


Figura 4.2.11 5 Genera Reportes nivel 2

5 Genera Reportes nivel 2

En la figura 4.2.11. Se observa la función de genera reportes que permite la consulta mediante reportes de los documentos y catálogos del sistema. Esta función solo la puede realizar el usuario con derechos de la empresa.

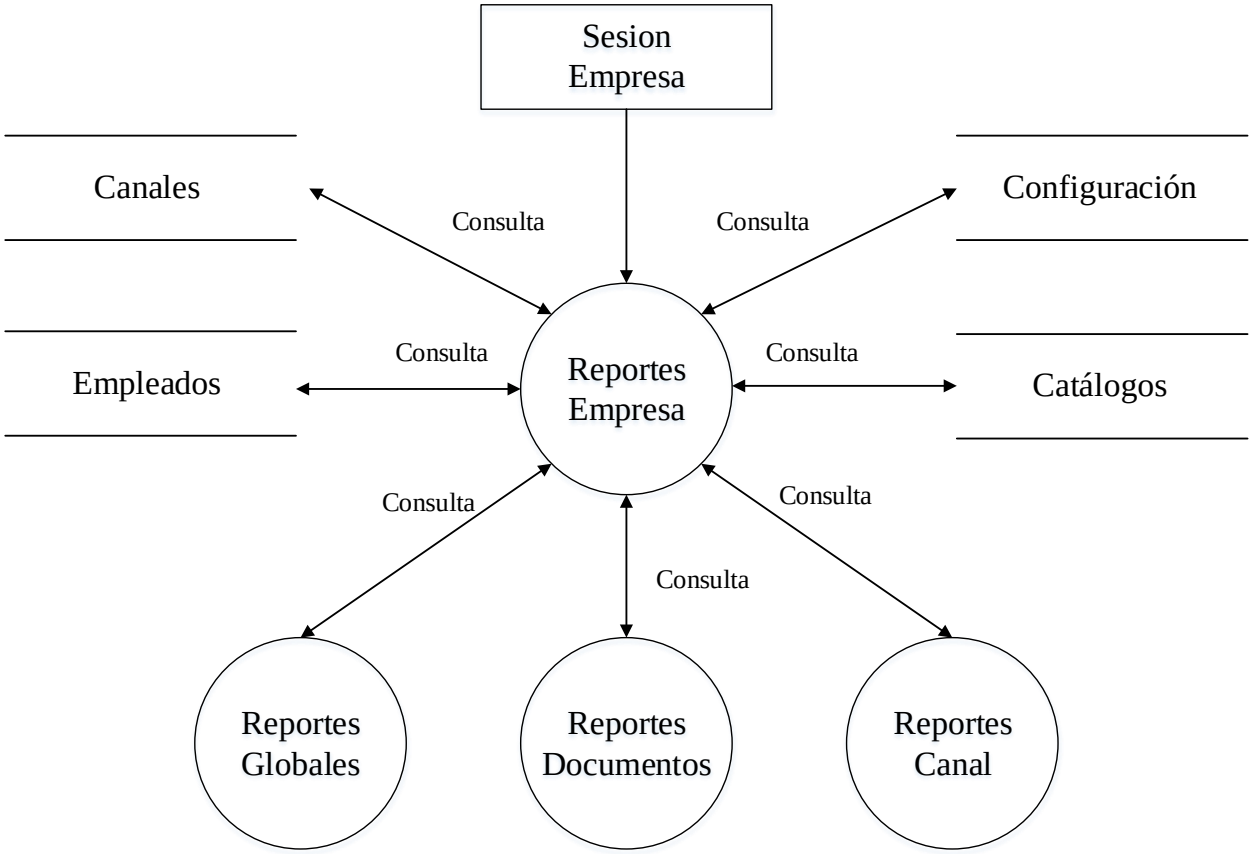


Figura 4.2.12 5 Genera Reportes nivel 3

5 Genera Reportes nivel 3

En la figura 4.2.12. Se observa la función Genera Reportes que permite la consulta de los reportes de los documentos y de acuerdo a los privilegios se subdivide Reportes globales, que incluyen reportes de todas las compañías, Reportes de documentos que permite la obtención de reportes de documentos en especial, Reportes de canal que permite la obtención de reportes de órdenes de servicio del canal en cuestión. Esta función depende del usuario que la accesa.

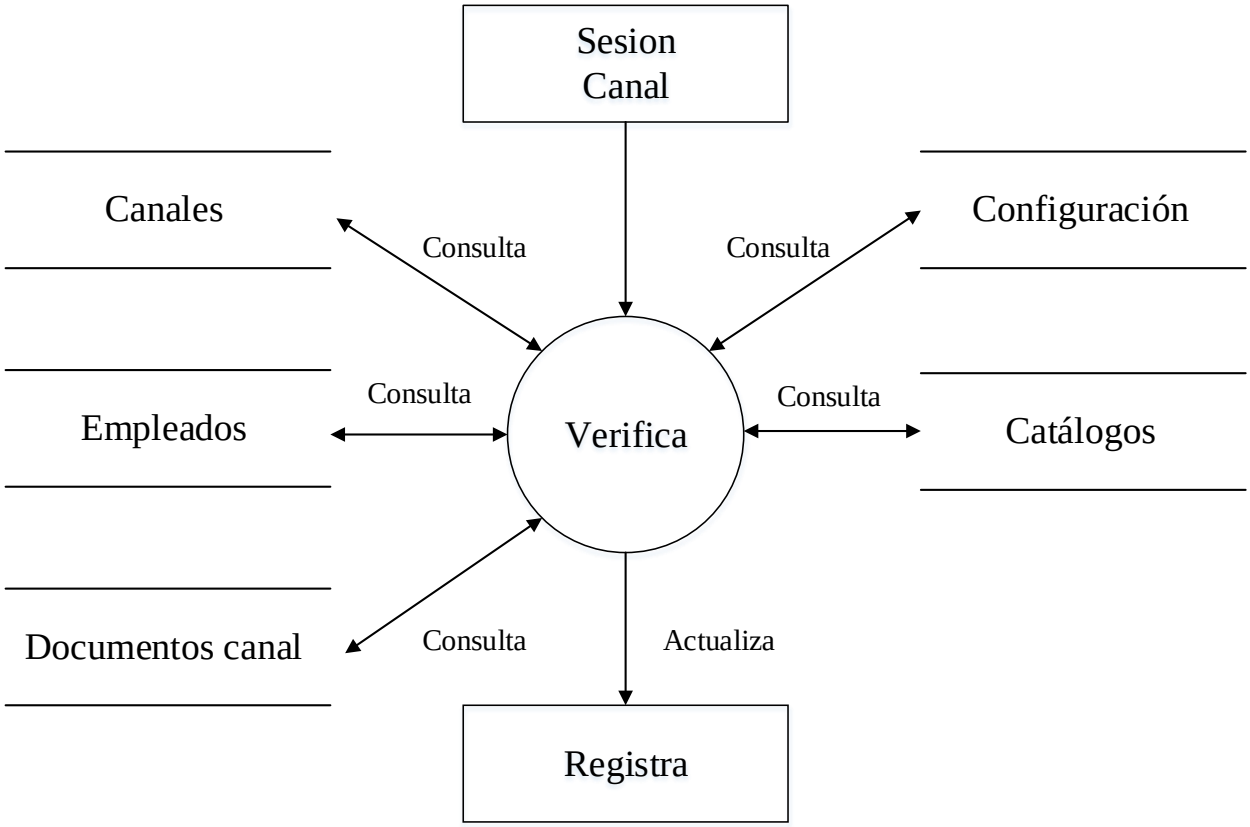


Figura 4.2.13 6 Administración de Documentos nivel 2 Canal

6 Administración de Documentos nivel 2 Canal

En la figura 4.2.13. Se observa la función de Administrar documentos que permite modificar, dar de alta Órdenes de Servicio o nuevos empleados de acuerdo a los parámetros establecidos. Esta función la puede realizar el usuario con derechos de la empresa y o el canal.

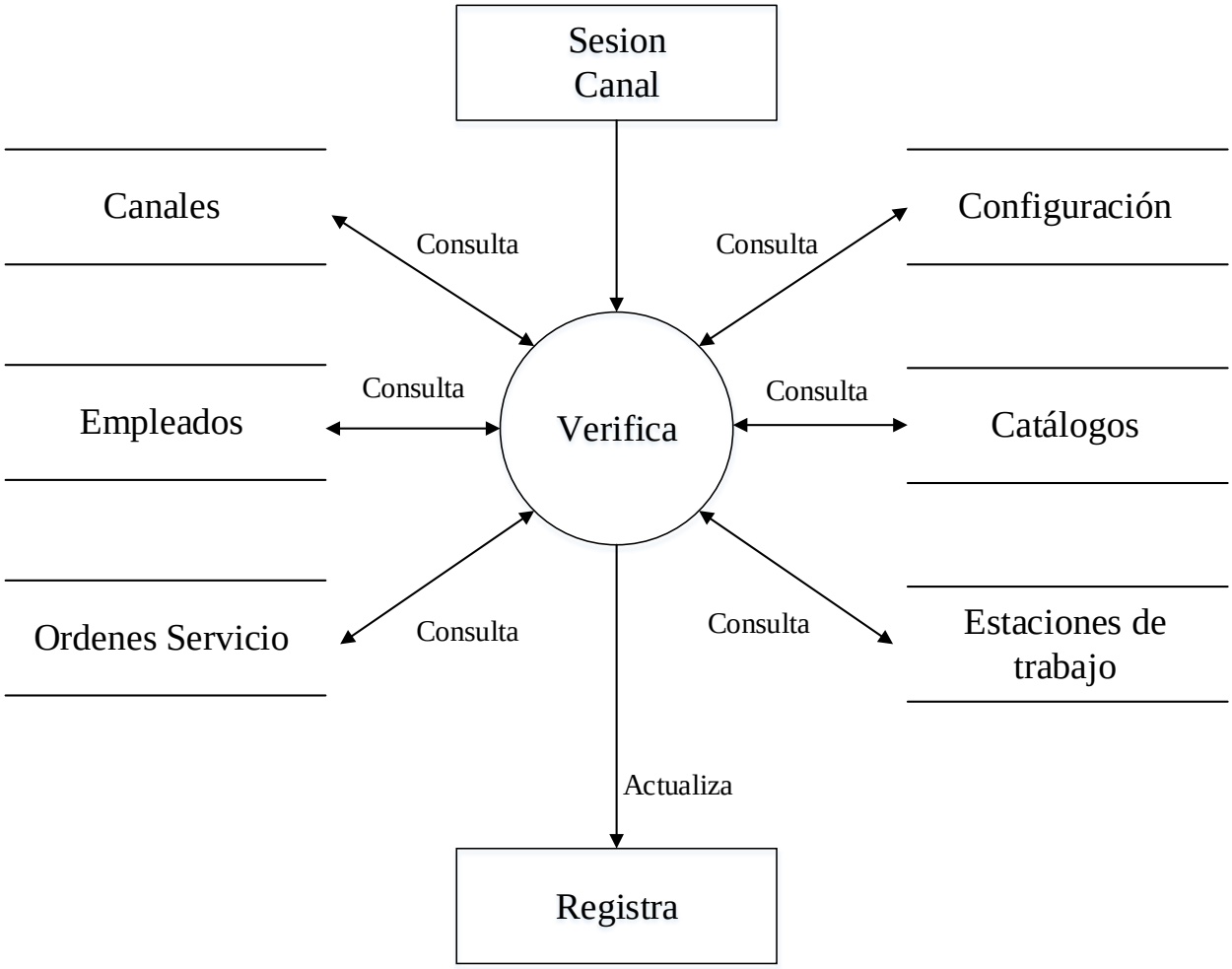


Figura 4.2.14 6 Administración de Documentos nivel 3 Canal

6 Administración de Documentos nivel 3 Canal
En la figura 4.2.14. Se observa la función de Administrar documentos que permite modificar, dar de alta Órdenes de servicio, y ver su estatus para mostrar el avance de los procesos.

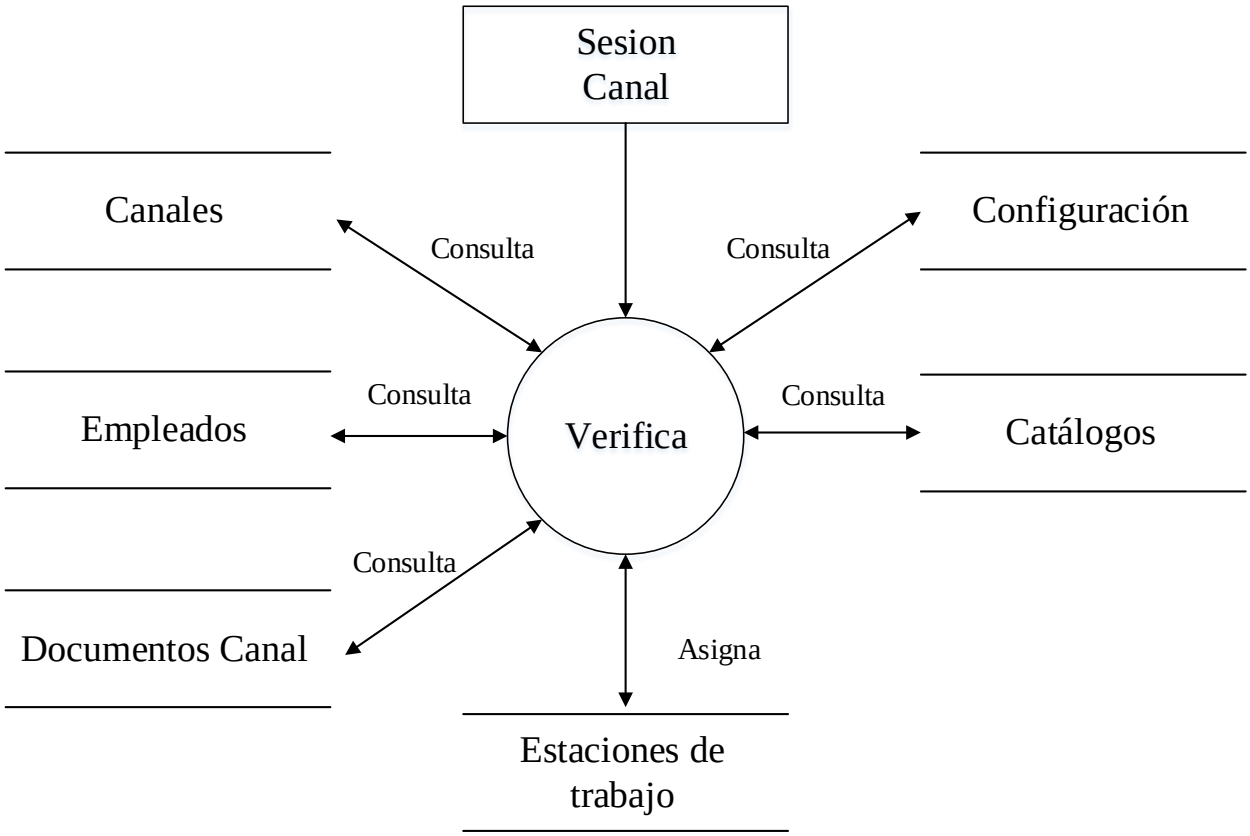


Figura 4.2.15 7 Visualiza Canal nivel 2

7 Visualiza Canal nivel 2

En la figura 4.2.15. Se observa la función de Visualización que permite la consulta de los documentos del canal. Esta función la puede realizar el usuario con derechos de la empresa y el canal.

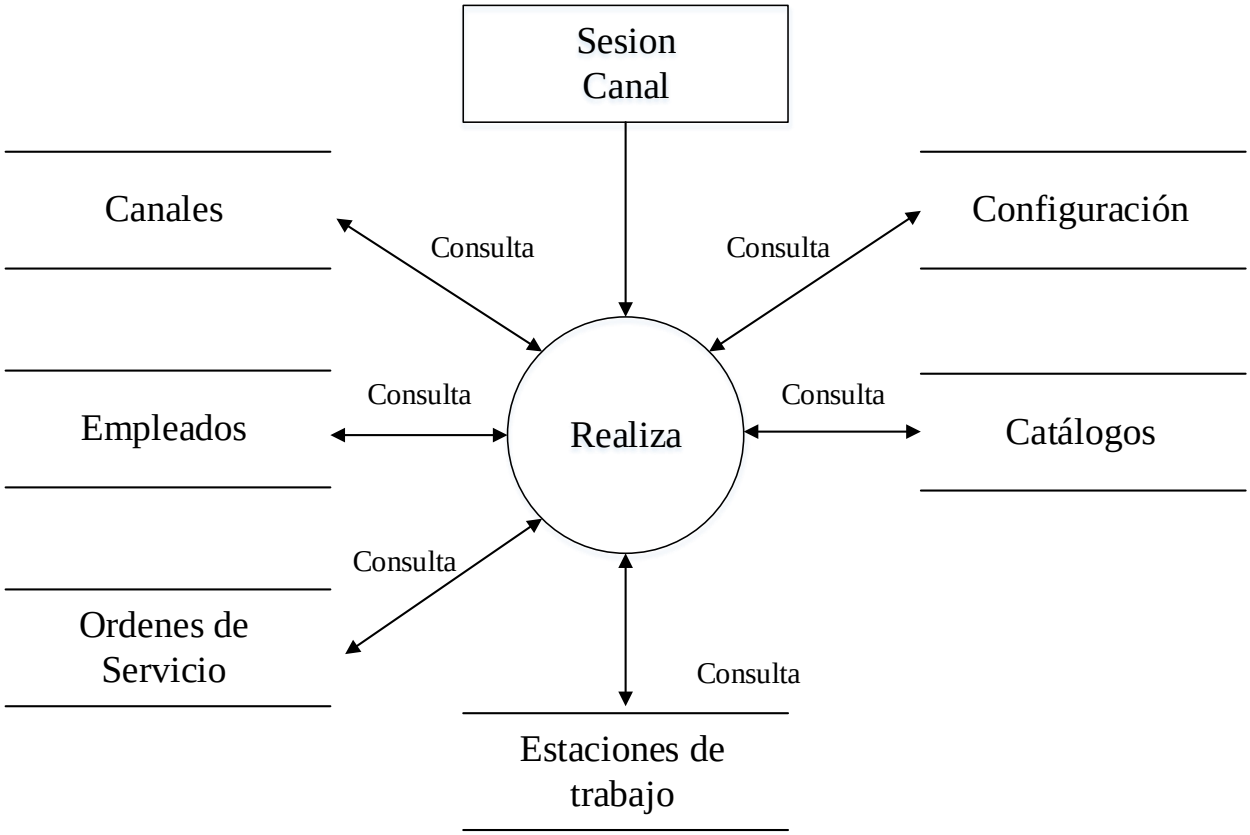


Figura 4.2.16 7 Visualiza Canal nivel 3

7 Visualiza Canal nivel 3

En la figura 4.2.16. Se observa la función de Visualización que permite la consulta de Órdenes de servicio. Esta función es la que permite la búsqueda y consulta de las órdenes de servicio de los canales. La función la puede realizar el usuario con derechos de la empresa y o canal.

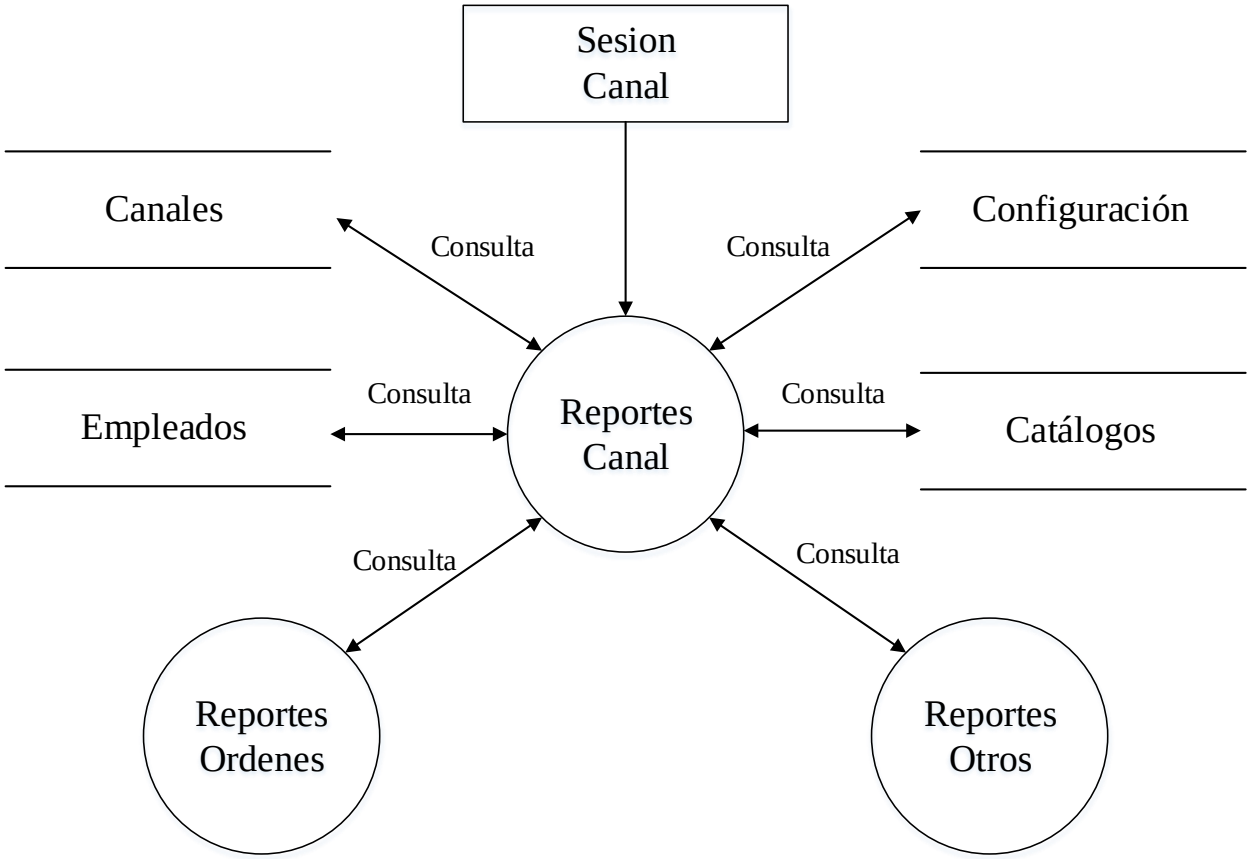


Figura 4.2.17 8 Genera Reportes Canal nivel 2

8 Genera Reportes Canal nivel 2

En la figura 4.2.17. Se observa la función de genera reportes de canal que permite la generación de reportes de los documentos del canal. Esta función la puede realizar el usuario con derechos de la empresa o el canal.

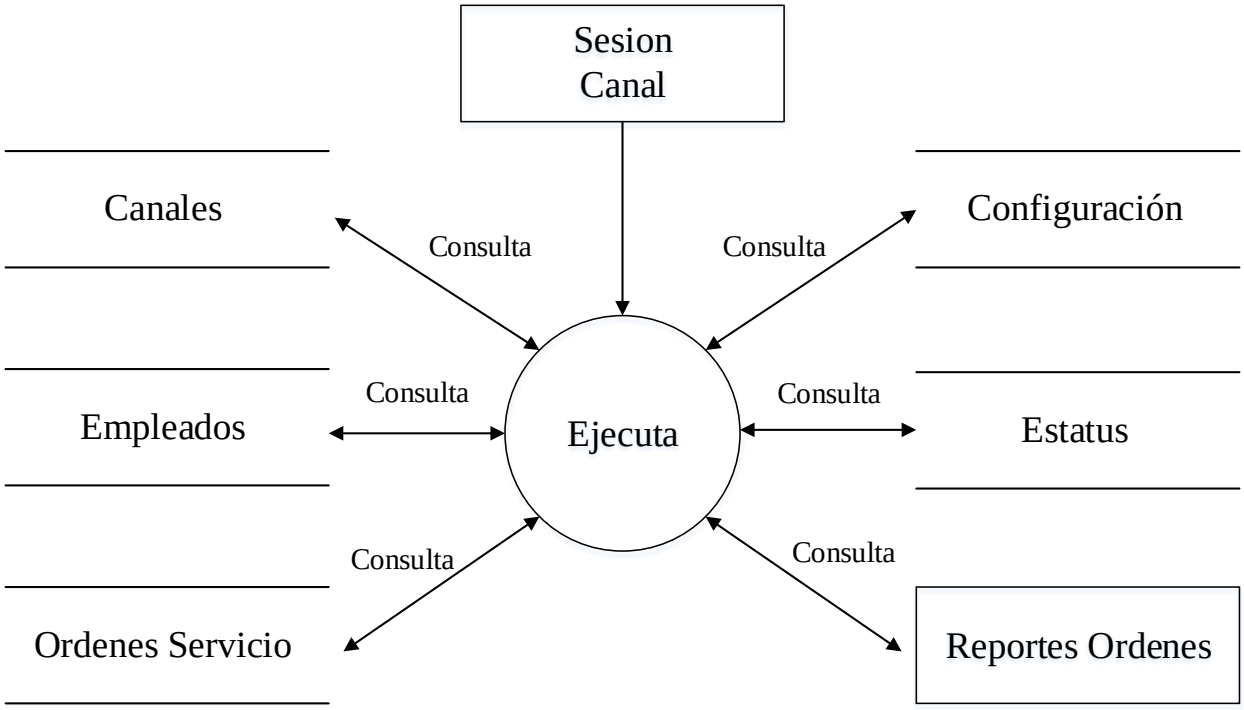


Figura 4.2.18 8 Genera Reportes Canal nivel 3

8 Genera Reportes Canal nivel 3

En la figura 4.2.18. Se observa la función Genera Reportes de Canal que permite la consulta de los reportes de Ordenes de servicio y de acuerdo a los parámetros. Esta función depende del usuario que ingrese.

3. Diagrama de Flujo de Datos

En el diagrama de la figura 4.3.1 se muestran los elementos de esta herramienta.

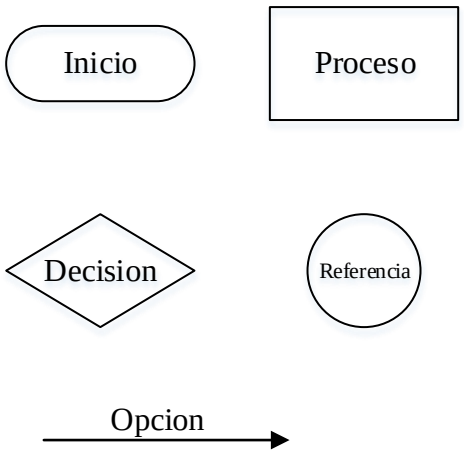


Figura 4.3.1 Elementos de diagramas de Flujo

A continuación se muestran los diagramas de flujo de datos de los procesos presentados anteriormente. En la figura 4.3.2 se muestra el diagrama de flujo de datos general.

Flujo Proceso General

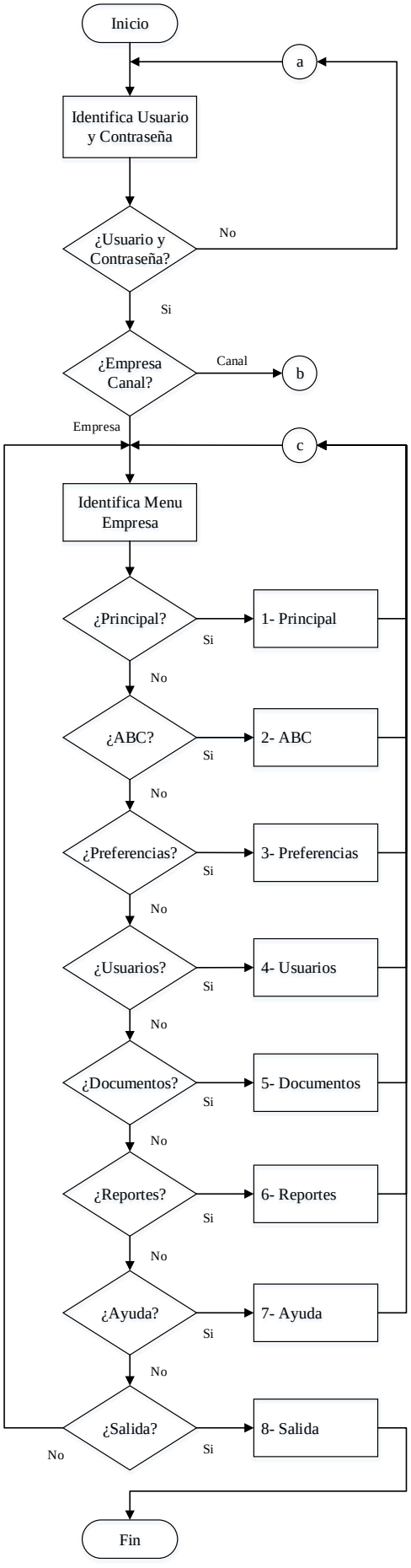


Figura 4.3.2 Proceso general

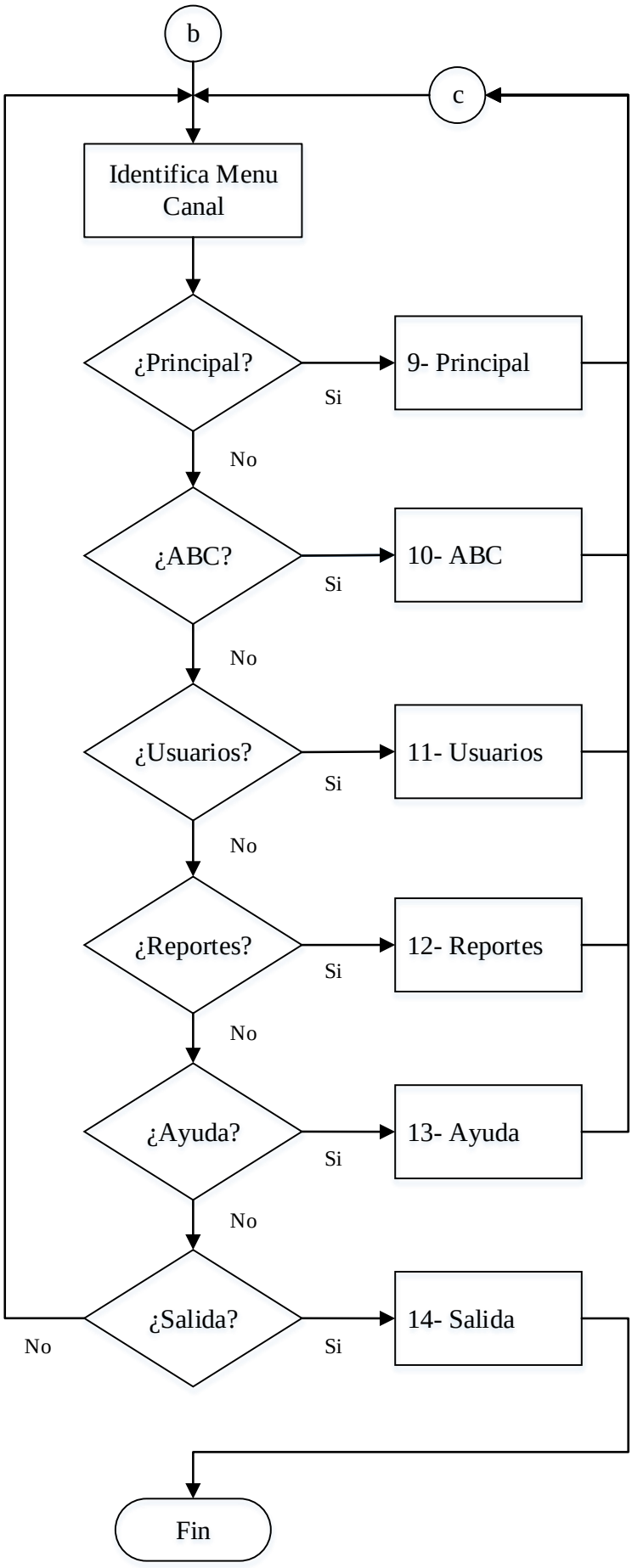


Figura 4.3.3 Proceso general b

Diagrama Principal

Una vez que se define el tipo de usuario que ingreso al sistema se tiene acceso a las opciones correspondientes. En la opción principal se encapsulan funciones de regreso a la pantalla principal, se eligen los documentos con los que se quiere trabajar desde la estación de trabajo ya determinada y se puede observar la presentación de la compañía, también se tiene acceso a la salida del sistema. Figura 4.3.4

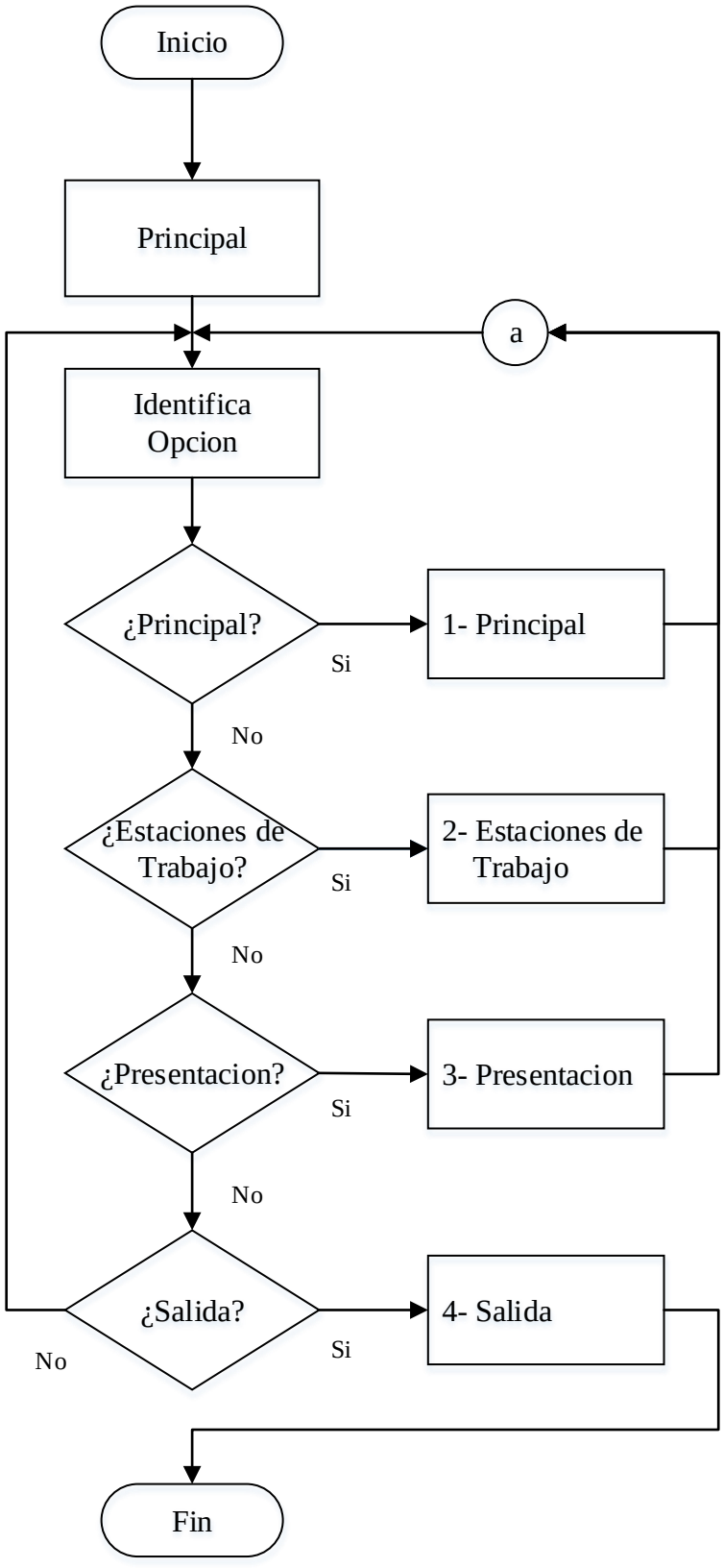


Figura 4.3.4 Flujo Principal

Diagrama ABC

Se agrupan funciones de alta de Órdenes de servicio y consulta de las misas y de catálogos. Es la función principal del seguimiento del estatus de órdenes y la que inicia el ciclo principal. A la función de alta de órdenes solo tienen acceso los usuarios con privilegios de la empresa.
La función de consulta de catálogos se utiliza para revisión de los diferentes elementos necesarios para dar de alta una orden. . Figura 4.3.5

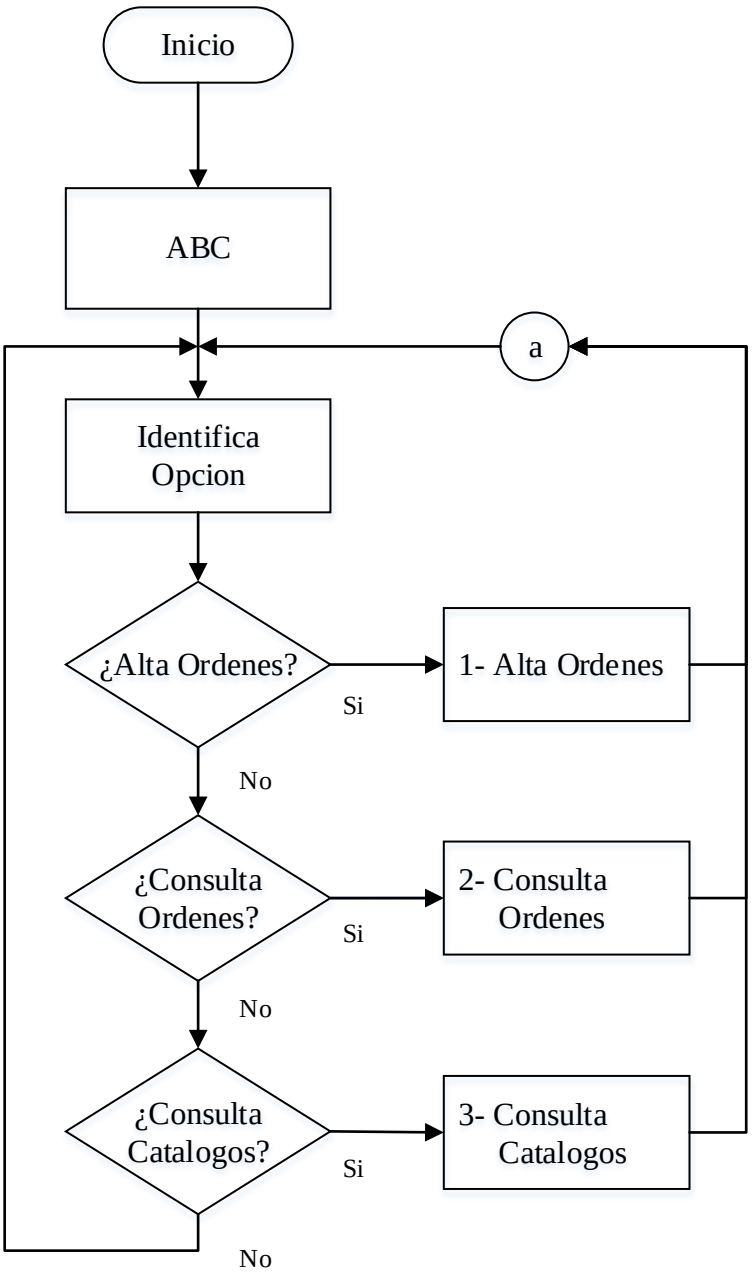


Figura 4.3.5 Flujo ABC

Flujo Usuarios

En esta función se pueden dar de alta a los usuarios del sistema. Solo los usuarios con privilegios tienen acceso a esta opción. En el caso de empleados de los canales de distribución solo pueden solicitar las claves de acceso y no darlas de alta, así como también solicitar modificaciones al estatus de alguna orden o datos relacionados con su información. Figura 4.3.6

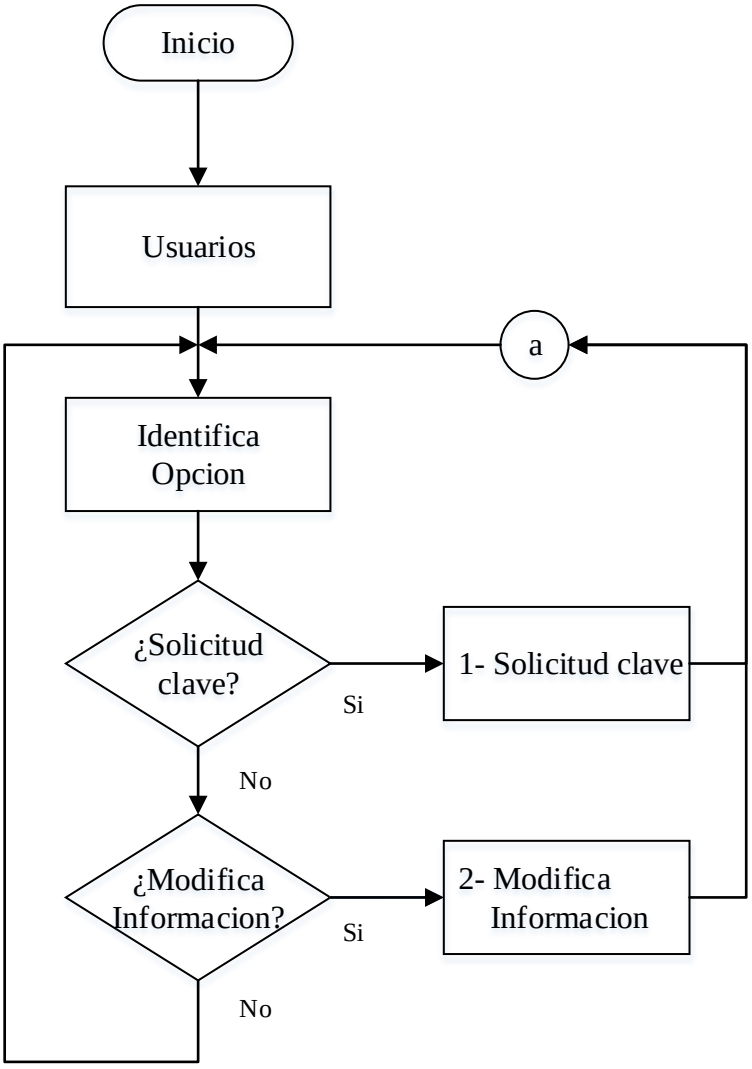


Figura 4.3.6 Usuarios

Flujo de Preferencias

Una vez configurado los parámetros principales del sistema, se pueden realizar configuraciones que permitan un mayor entendimiento de la información presentada, es el caso de la configuración de campos que permite entre otras cosas cambio de nombre de columnas, ordenamiento, visibilidad. El flujo de los documentos y los estatus de estos también puede ser modificado mediante la configuración de flujo. Estas son funciones solo para administradores de la aplicación. Figura 4.3.7

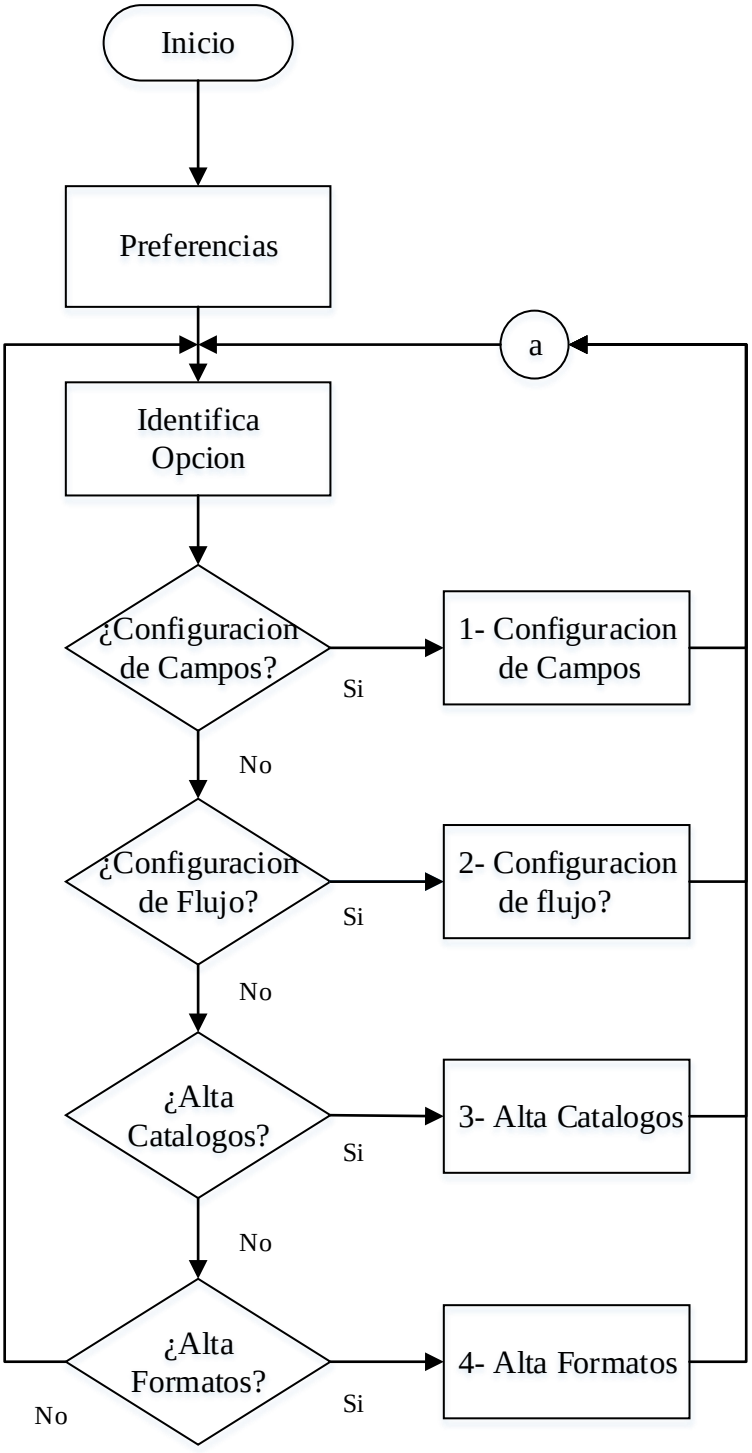


Figura 4.3.7 Preferencias

Flujo de Formatos

Estas funciones son únicamente de consulta y solo muestran formatos base para los vendedores de los formularios, contratos, cotizaciones y precios. Figura 4.3.8

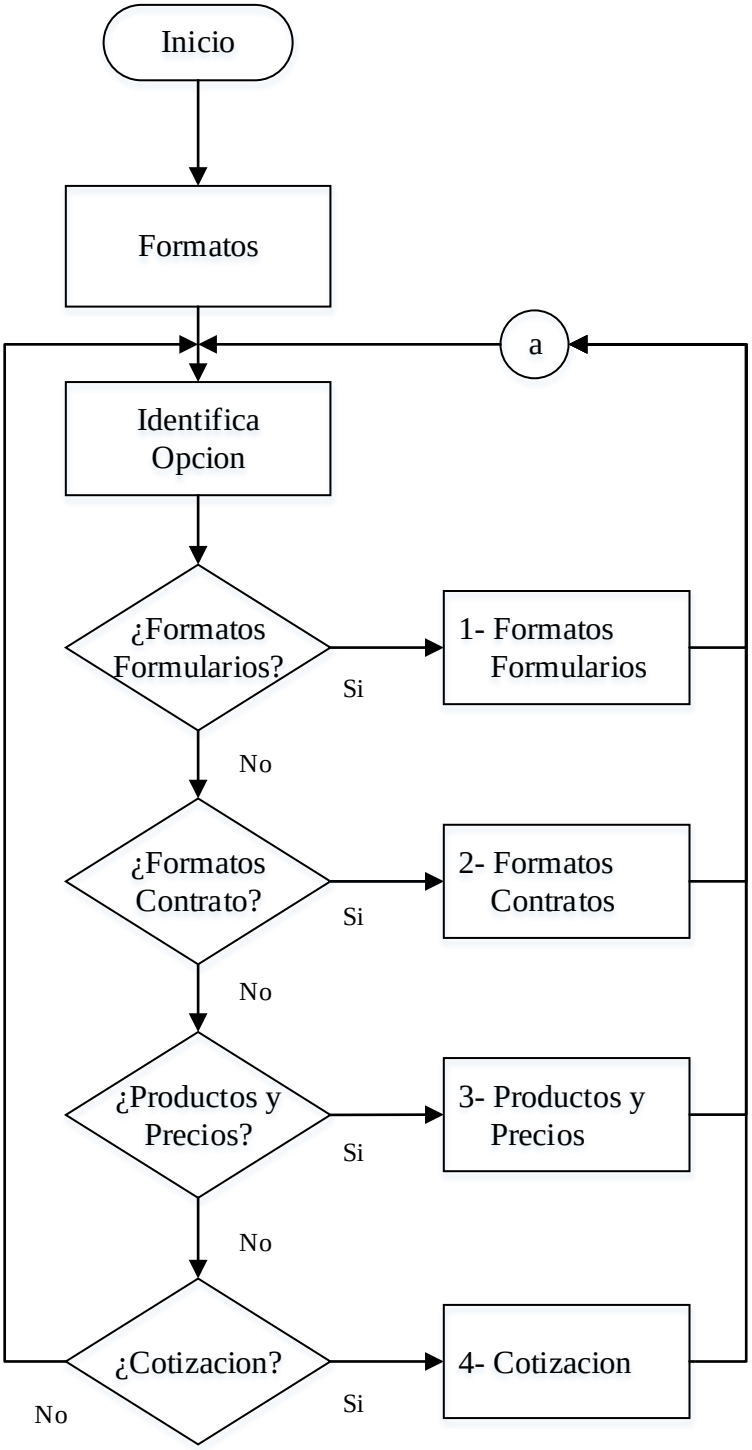


Figura 4.3.8 Formatos

Flujo de Reportes

Las funciones de reportes una vez que se tiene acceso a ellos comienza por la decisión de que es lo que se desea reportar y si se requiere de los datos específicos para la obtención del reporte.
Una vez seleccionado el documento se puede acceder a los distintos parámetros con los que se puede obtener un reporte. Figura 4.3.9

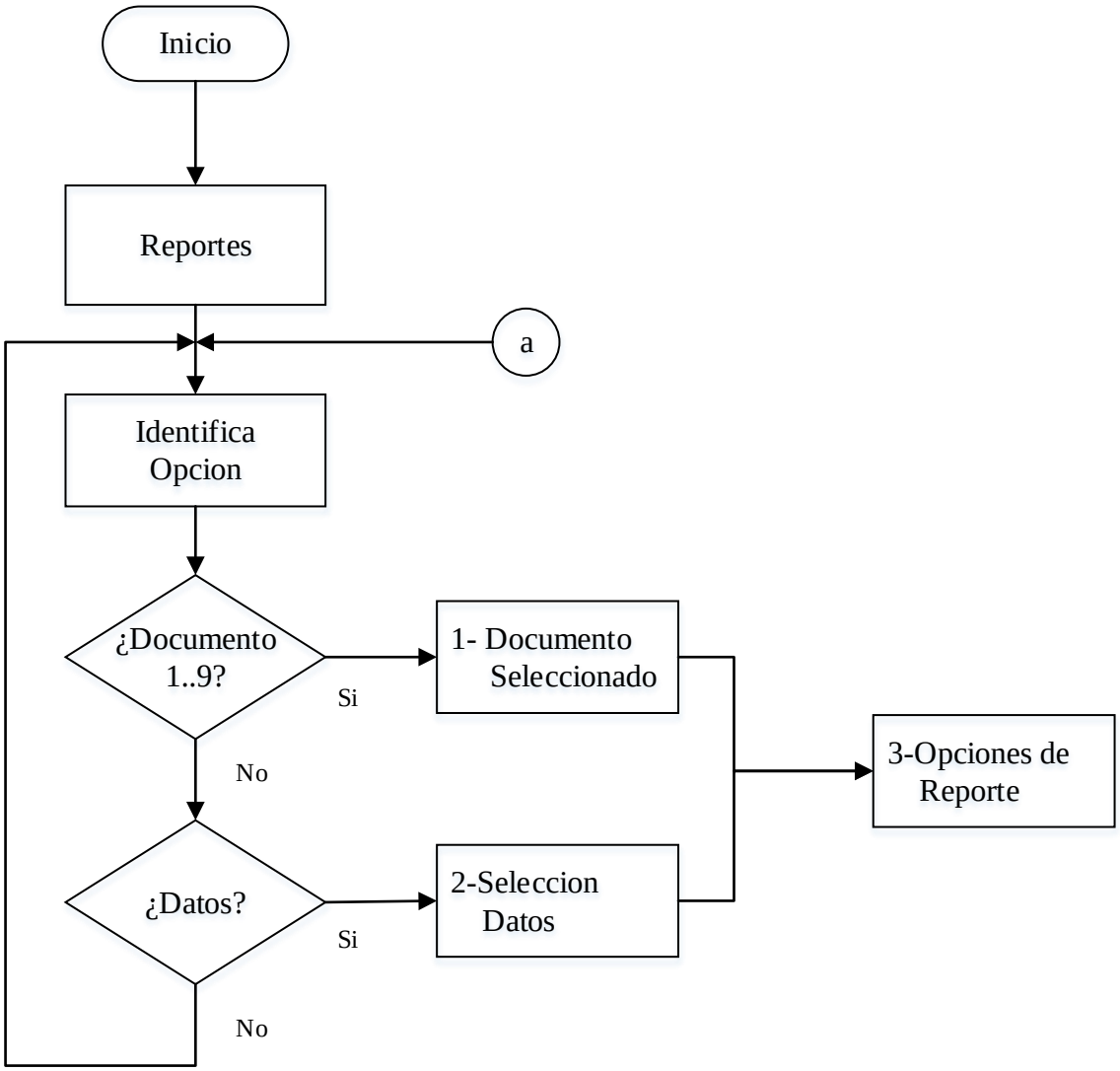


Figura 4.3.9 Reportes

Flujo Opciones de reportes

Una vez seleccionado el documento se eligen el tipo de gráfico, los parámetros que se pueden seleccionar por documento como son la división del reporte o columna clave, el periodo en el que se efectuará el reporte y los filtros que se desea establecer. Figura 4.3.10

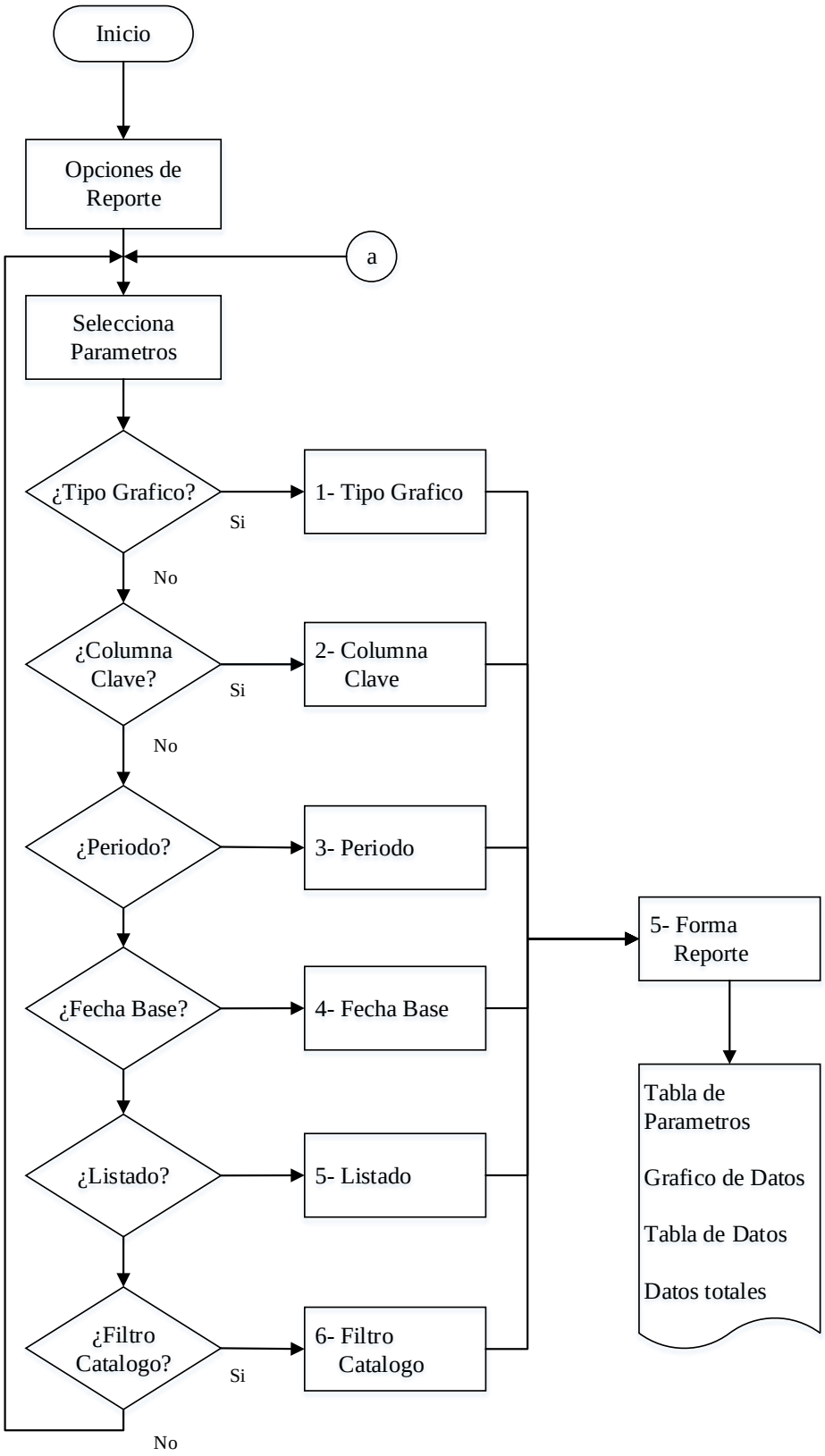


Figura 4.3.10 Opciones de Reportes

Flujo de Ayuda

Presenta un documento accesible a todos sobre los conceptos principales en el sistema como documentos, estaciones de trabajo, reportes y pantallas. Figura 4.3.11

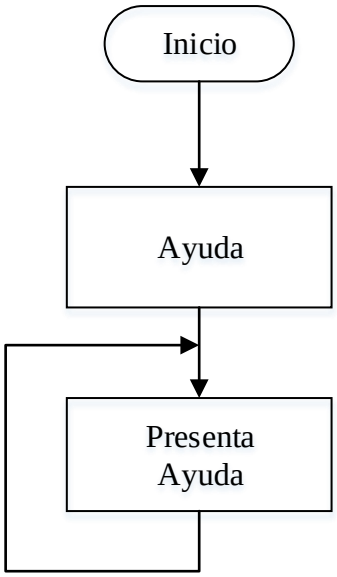


Figura 4.3.11 Ayuda

Flujo de Salida

Establece la salida del sistema y en su caso elimina parámetros del sesión y conexión. Figura 4.3.12

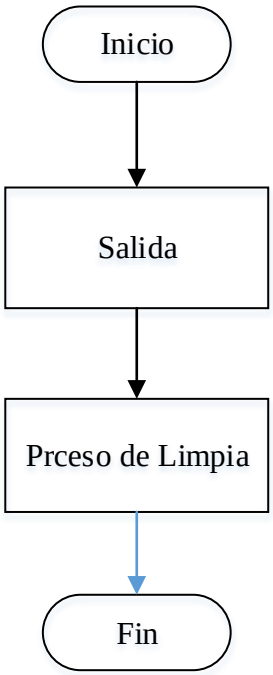


Figura 4.3.12 Salida

4. Diccionario de datos DD

Un diccionario de datos DD es un conjunto de metadatos que contiene las características lógicas y puntuales de los datos que se van a utilizar en el sistema, incluyendo nombre, descripción, alias, contenido y organización.

En un diccionario de datos se encuentra la lista de todos los elementos que forman parte del flujo de datos de todo el sistema.

Los datos elementales son aquellos para los cuales no hay una descomposición significativa. Y deben ser introducidos en el DD y proveer una breve descripción que describa el significado del dato. En el caso de que el dato tenga un nombre significativo, se puede omitir la descripción, sin embargo; es importante especificar las unidades de medida que el dato puede tomar.

Con el propósito de mostrar lo más importante del sistema en la tabla 4.4.1 se muestran los documentos más importantes de la aplicación con el nombre físico y la llave primaria.

Los documentos mostrados en la tabla 4.4.1 se crean a lo largo de los procesos de la aplicación, comenzando con la *Orden de Servicio* misma que da origen a *Estudios de factibilidad, Estudio AIR, Equipos para instalación, Órdenes de compra, Instalaciones, Material adicional, Servicios instalados, Actas de Recepción*.

Descripción	**Tabla**	**Pkey**
Ordenes de Servicio	DatosOS	consecutivo
Estudio de Factibilidad	DatosFactibilidad	CLVFACT
Estudio AIR	DatosAir	CLVAIR
Equipos para Instalación	DatosEqInst	CLVQINT
Órdenes de compra	DatosOC_PP	CLVOCPP
Instalaciones	DatosInst	CLVINST
Material Adicional de Instalaciones	DatosEquiAdic	CLVADIC
Servicios Instalados	DatosServInst	CLVSIA
Actas de Recepción	DatosAR	CLVAR

Figura 4.4.1 Tablas principales

En la figura 4.4.2 se muestran las columnas, el tipo de datos, las tablas relacionadas y la descripción de las tablas principales de la aplicación.

Tabla	**Nombre**	**Columna**	**Tipo de Dato**	**PK**	**Tablas Relacionadas**	**Descripción**
DatosOS	Ordenes de Servicio	consecutivo	Integer	PK		
DatosOS	Ordenes de Servicio	ANTCLVOS	Varchar	10		Clave Anterior
DatosOS	Ordenes de Servicio	CLVCanal	Integer		CatCanal	Canal Comercial
DatosOS	Ordenes de Servicio	RSEmpresa	Varchar	250		Razón Social
DatosOS	Ordenes de Servicio	Fechaelab	Date			Elaboración
DatosOS	Ordenes de Servicio	Direccion	Varchar	250		Dirección
DatosOS	Ordenes de Servicio	Colonia	Varchar	50		Colonia
DatosOS	Ordenes de	CP	Varchar	5		Código Postal

	Servicio					
DatosOS	Ordenes de Servicio	CLVDEMU	Integer		CatDemu	Municipios
DatosOS	Ordenes de Servicio	CLVEDO	Integer		CatEdo	Estados
DatosOS	Ordenes de Servicio	RFC	Varchar	13		Registro Federal de Contribuyentes
DatosOS	Ordenes de Servicio	Contacto1	Varchar	50		Contacto con el Cliente
DatosOS	Ordenes de Servicio	Telefono2	Varchar	15		Teléfono Contacto
DatosOS	Ordenes de Servicio	Fax	Varchar	15		Fax
DatosOS	Ordenes de Servicio	CLVBW	Integer		CatBW	Ancho de banda
DatosOS	Ordenes de Servicio	NumIP	Integer			Numero de Direcciones IP
DatosOS	Ordenes de Servicio	PlazoContrato	Integer			Plazo de contrato
DatosOS	Ordenes de Servicio	Calleprox1	Varchar	50		Calle Próxima
DatosOS	Ordenes de Servicio	Calleprox2	Varchar	50		Calle Próxima
DatosOS	Ordenes de Servicio	NomSolicitante	Varchar	50		Solicitante de la OS
DatosOS	Ordenes de Servicio	FechaSol	Date			Fecha de Solicitud
DatosOS	Ordenes de Servicio	EjeCta	Varchar	50		Ejecutivo de Cuenta
DatosOS	Ordenes de Servicio	CLVSERV	Integer		CatTipoServicio	Tipo Servicio
DatosOS	Ordenes de Servicio	CLVTSS	Integer		CatTipoServSol	Tipo de Solución Servicio
DatosOS	Ordenes de Servicio	Comentarios	Memo	page		Comentarios
DatosOS	Ordenes de Servicio	CLVSTATUS	Integer		CatStatus	Estatus
DatosOS	Ordenes de Servicio	CLVSEGUIM	Integer		CatSeguim	Seguimiento
DatosEqInst	Equipos para Instalación	CLVQINT	Integer	PK		
DatosEqInst	Equipos para Instalación	Cant	Integer			Cantidad
DatosEqInst	Equipos para Instalación	descrpeq	Varchar	20		Descripción por equipo
DatosEqInst	Equipos para Instalación	Noserie	Varchar	40		Número de Serie
DatosEqInst	Equipos para Instalación	capacidad	Varchar	50		Capacidad
DatosEqInst	Equipos para Instalación	instalador	Varchar	50		Nombre del Instalador
DatosEqInst	Equipos para Instalación	comentarios	Memo	page		comentarios
DatosEqInst	Equipos para Instalación	CLVCanal	Integer		CatCanal	Canal Comercial
DatosEqInst	Equipos para Instalación	consecutivo	Integer		DatosOS	Ordenes de Servicio
DatosEqInst	Equipos para Instalación	CLVSTATUS	Integer		CatStatus	Estatus
DatosEqInst	Equipos para Instalación	fecha_elab	Date			Elaboración
DatosFactibilidad	Estudio de Factibilidad	CLVFACT	Integer	PK		
DatosFactibilidad	Estudio de Factibilidad	Recibe	Varchar	50		Nombre quien Recibe EF
DatosFactibilidad	Estudio de Factibilidad	Fecharecepcion	Date			Recepción

DatosFactibilidad	Estudio de Factibilidad	Fechalimiteentrega	Date			Limite de entrega
DatosFactibilidad	Estudio de Factibilidad	Fechaprogramacion	Date			Fecha programada
DatosFactibilidad	Estudio de Factibilidad	CLVResultado	Integer		CatResultadofact	Resultado Factibilidad
DatosFactibilidad	Estudio de Factibilidad	CLVTFAC	Integer		CatTipofact	Tipo Factibilidad
DatosFactibilidad	Estudio de Factibilidad	Comentarios	Memo	page		Comentarios
DatosFactibilidad	Estudio de Factibilidad	GPSN	Varchar	10		Dirección GPS Norte
DatosFactibilidad	Estudio de Factibilidad	GPSW	Varchar	10		Dirección GPS Oeste
DatosFactibilidad	Estudio de Factibilidad	Altura	Varchar	10		Altura
DatosFactibilidad	Estudio de Factibilidad	Enlaceconrepetidora	Varchar	50		Estación Repetidora
DatosFactibilidad	Estudio de Factibilidad	Distancia	Integer			Distancia a la Estación
DatosFactibilidad	Estudio de Factibilidad	EsfHr_Hom	Integer			Número de horas de instalación
DatosFactibilidad	Estudio de Factibilidad	Fechaentrega	Date			Fecha entregada
DatosFactibilidad	Estudio de Factibilidad	NomrealizoEF	Varchar	50		Nombre quien realiza EF
DatosFactibilidad	Estudio de Factibilidad	ObsCliente	Memo	page		Observaciones del Cliente
DatosFactibilidad	Estudio de Factibilidad	Recibio	Varchar	50		Nombre de quien Recibe
DatosFactibilidad	Estudio de Factibilidad	CLVSTATUS	Integer		CatStatus	Estatus
DatosFactibilidad	Estudio de Factibilidad	CLVSEGUIM	Integer		CatSeguim	Seguimiento
DatosFactibilidad	Estudio de Factibilidad	consecutivo	Integer		DatosOS	Ordenes de Servicio
DatosFactibilidad	Estudio de Factibilidad	Ubicacion	Varchar	OLE		Foto Edificio
DatosFactibilidad	Estudio de Factibilidad	Azotea	Varchar	OLE		Foto Azotea
DatosFactibilidad	Estudio de Factibilidad	CComputo	Varchar	OLE		Foto Centro de computo
DatosFactibilidad	Estudio de Factibilidad	Trayecto	Varchar	OLE		Imagen Trayectoria cable
DatosFactibilidad	Estudio de Factibilidad	CLVCanal	Integer		CatCanal	Canal Comercial
DatosInst	Instalaciones	CLVINST	Integer	PK		
DatosInst	Instalaciones	consecutivo	Integer		DatosOS	Ordenes de Servicio
DatosInst	Instalaciones	CLVEDO	Integer		CatEdo	Estados
DatosInst	Instalaciones	FechaRInst	Date			Fecha de realización
DatosInst	Instalaciones	DescripcionInst	Varchar	250		Descripción de instalación
DatosInst	Instalaciones	NomRespInst	Varchar	50		Responsable de Instalación
DatosInst	Instalaciones	NomCapdatos	Varchar	50		Nombre Capturo Datos
DatosInst	Instalaciones	Fechaproginst	Date			Fecha Programada
DatosInst	Instalaciones	PathAC	Varchar	10		Directorio de Acceso
DatosInst	Instalaciones	DescripcionAC	Varchar	250		Descripción del Documento
DatosInst	Instalaciones	Fechaentregainst	Date			Fecha de entrega
DatosInst	Instalaciones	FechaEntAC	Date			Fecha de entrega acta
DatosInst	Instalaciones	FecharDocOR	Date			Fecha de elaboración del documento
DatosInst	Instalaciones	DocOrig	Varchar	250		Nombre del documento
DatosInst	Instalaciones	Comentarios	Memo	page		Comentarios
DatosInst	Instalaciones	Razones	Memo	page		Razones
DatosInst	Instalaciones	CLVSTATUS	Integer		CatStatus	Estatus
DatosInst	Instalaciones	CLVCanal	Integer		CatCanal	Canal Comercial

DatosEquiAdic	Material Adicional de Instalaciones	CLVADIC	Integer	PK		
DatosEquiAdic	Material Adicional de Instalaciones	descripcion	Memo	page		Descripción del Material
DatosEquiAdic	Material Adicional de Instalaciones	costo	Double			Costo del Material
DatosEquiAdic	Material Adicional de Instalaciones	CLVOCPP	Integer		DatosOC_PP	Órdenes de Compra
DatosEquiAdic	Material Adicional de Instalaciones	CLVCanal	Integer		CatCanal	Canal Comercial
DatosEquiAdic	Material Adicional de Instalaciones	fecha_elab	Date			Fecha de Registro del material
DatosOC_PP	Órdenes de compra	CLVOCPP	Integer	PK		
DatosOC_PP	Órdenes de compra	consecutivo	Integer		DatosOS	Ordenes de Servicio
DatosOC_PP	Órdenes de compra	CLVCanal	Integer		CatCanal	Canal Comercial
DatosOC_PP	Órdenes de compra	DuracionServ	Varchar	50		Periodo de contratación
DatosOC_PP	Órdenes de compra	CLVEDO	Integer		CatEdo	Estados
DatosOC_PP	Órdenes de compra	CLVBW	Integer		CatBW	Ancho de banda
DatosOC_PP	Órdenes de compra	CostAct	Double			Costo de activación
DatosOC_PP	Órdenes de compra	DescActiv	Double			Descripción de activación
DatosOC_PP	Órdenes de compra	Periocidad	Varchar	50		Períodos de pago
DatosOC_PP	Órdenes de compra	CostoRent	Double			Costo de Renta
DatosOC_PP	Órdenes de compra	DescRent	Double			Descuento en Renta
DatosOC_PP	Órdenes de compra	PorcDesc	Double			Porcentaje de Descuento
DatosOC_PP	Órdenes de compra	FechaelabOCPP	Date			Fecha de elaboración
DatosOC_PP	Órdenes de compra	RepLega	Varchar	50		Representante legal del Cliente
DatosOC_PP	Órdenes de compra	CLVTFAC	Integer		CatTipofact	Tipo Factibilidad
DatosOC_PP	Órdenes de compra	CLVSTATUS	Integer		CatStatus	Estatus
DatosOC_PP	Órdenes de compra	NomElaocpp	Varchar	50		Nombre de quien Elaboro
DatosOC_PP	Órdenes de compra	FechaenvioOCPP	Date			Fecha de envío
DatosOC_PP	Órdenes de compra	Comentarios	Memo	page		Comentarios
DatosOC_PP	Órdenes de compra	RespuestaCliente	Memo			Respuesta del Cliente
DatosOC_PP	Órdenes de compra	CLVSEGUIM	Integer		CatSeguim	Seguimiento
DatosServInst	Servicios Instalados	CLVSIA	Integer	PK		
DatosServInst	Servicios Instalados	cantidad	Integer			Número de servicios
DatosServInst	Servicios Instalados	descripcion	Varchar	250		Descripción del Servicio
DatosServIns	Servicios	fechainnst	Date			Fecha de instalación de Servicios

t	Instalados					
DatosServInst	Servicios Instalados	CLVCanal	Integer		CatCanal	Canal Comercial
DatosServInst	Servicios Instalados	consecutivo	Integer		DatosOS	Ordenes de Servicio
DatosAR	Actas de Recepción	CLVAR	Integer	PK		
DatosAR	Actas de Recepción	consecutivo	Integer		DatosOS	Ordenes de Servicio
DatosAR	Actas de Recepción	fechaar	Date			Fecha de elaboración
DatosAR	Actas de Recepción	CLVEDO	Integer		CatEdo	Estados
DatosAR	Actas de Recepción	subred	Varchar	50		Dirección IP de la Subred
DatosAR	Actas de Recepción	mask	Varchar	50		Mascara de la red
DatosAR	Actas de Recepción	CLVBW	Integer		CatBW	Ancho de banda
DatosAR	Actas de Recepción	distancia	Varchar	50		Distancia a la Torre
DatosAR	Actas de Recepción	coordpuntaAN	Varchar	50		Dirección GPS Norte Punta A
DatosAR	Actas de Recepción	coordpuntaAW	Varchar	50		Dirección GPS Oeste Punta A
DatosAR	Actas de Recepción	coordpuntaBN	Varchar	50		Dirección GPS Norte Punta B
DatosAR	Actas de Recepción	coordpuntaBW	Varchar	50		Dirección GPS Oeste Punta B
DatosAR	Actas de Recepción	Base	Varchar	50		Torre base
DatosAR	Actas de Recepción	puntaA	Varchar	50		Torre base
DatosAR	Actas de Recepción	puntaB	Varchar	50		Torre Cliente
DatosAR	Actas de Recepción	frecuencia	Varchar	50		Frecuencia programada
DatosAR	Actas de Recepción	netmonlogin	Varchar	50		Usuario de Red
DatosAR	Actas de Recepción	netmonpass	Varchar	50		Clave de usuario
DatosAR	Actas de Recepción	nomdircom	Varchar	50		Nombre Director Comercial
DatosAR	Actas de Recepción	nomdiradm	Varchar	50		Nombre Director Administración
DatosAR	Actas de Recepción	nomdirtec	Varchar	50		Nombre Director Tecnología
DatosAR	Actas de Recepción	nomrepcl	Varchar	50		Nombre Representante Cliente
DatosAR	Actas de Recepción	CLVSTATUS	Integer		CatStatus	Estatus
DatosAR	Actas de Recepción	CLVSEGUIM	Integer		CatSeguim	Seguimiento
DatosAR	Actas de Recepción	CLVCanal	Integer		CatCanal	Canal Comercial
DatosAir	Estudio AIR	CLVAIR	Integer	PK		
DatosAir	Estudio AIR	consecutivo	Integer		DatosOS	Ordenes de Servicio
DatosAir	Estudio AIR	fechaair	Date			Fecha de elaboración
DatosAir	Estudio AIR	CLVEDO	Integer		CatEdo	Estados
DatosAir	Estudio AIR	subred	Varchar	16		Dirección IP de la Subred
DatosAir	Estudio AIR	mask	Varchar	16		Mascara de la red
DatosAir	Estudio AIR	CLVBW	Integer		CatBW	Ancho de banda
DatosAir	Estudio AIR	gateway	Varchar	50		Dirección IP de la Marcara
DatosAir	Estudio AIR	Base	Varchar	50		Torre base
DatosAir	Estudio AIR	IP	Varchar	50		Dirección IP
DatosAir	Estudio AIR	puntaA	Varchar	50		Torre base
DatosAir	Estudio AIR	puntaB	Varchar	50		Torre del Cliente

DatosAir	Estudio AIR	frecuencia	Varchar	50		Frecuencia de Funcionamiento
DatosAir	Estudio AIR	CBQ	Varchar	50		CBQ
DatosAir	Estudio AIR	netmonlogin	Varchar	50		Usuario de Red
DatosAir	Estudio AIR	netmonpass	Varchar	50		Clave de usuario
DatosAir	Estudio AIR	IpMonitor	Varchar	50		Dirección IP de Monitoreo
DatosAir	Estudio AIR	VisualPulse	Varchar	50		Nemónico de Seguimiento
DatosAir	Estudio AIR	CLVSTATUS	Integer		CatStatus	Estatus
DatosAir	Estudio AIR	CLVSEGUIM	Integer		CatSeguim	Seguimiento
DatosAir	Estudio AIR	CLVCanal	Integer		CatCanal	Canal Comercial
DatosAir	Estudio AIR	fecha_term	Date			Fecha de terminación del estudio

Figura 4.4.2 Diccionario de datos Tablas Principales

Las tablas relevantes que no se describirán por cuestiones de espacio son:

Empleados, Canales, Estados, Municipios, Estatus de los diferentes documentos y otros catálogos y las tablas de configuración de la aplicación.

5. Diagrama Entidad Relación

El diagrama ER nos permite conocer en detalle la información que hay en cada entidad de datos y la relación que existe entre ellas. Tiene dos componentes principales:

- Tipos de objetos: que se representan por medio de rectángulos y representan una colección o grupo de objetos.
- Relaciones: Se representan por medio de conexiones entre los tipos de objetos por medio de flechas.

En las siguientes figuras de la 4.5.2 a la 4.5.7 se presentan algunos de los documentos principales y las relaciones que tienen entre ellos y las demás tablas del sistema de manera simplificada. El Diagrama ER se presenta en la figura 4.5.1.
Se agregan unas tablas de configuración.

Diagrama Entidad Relación

Figura 4.5.1

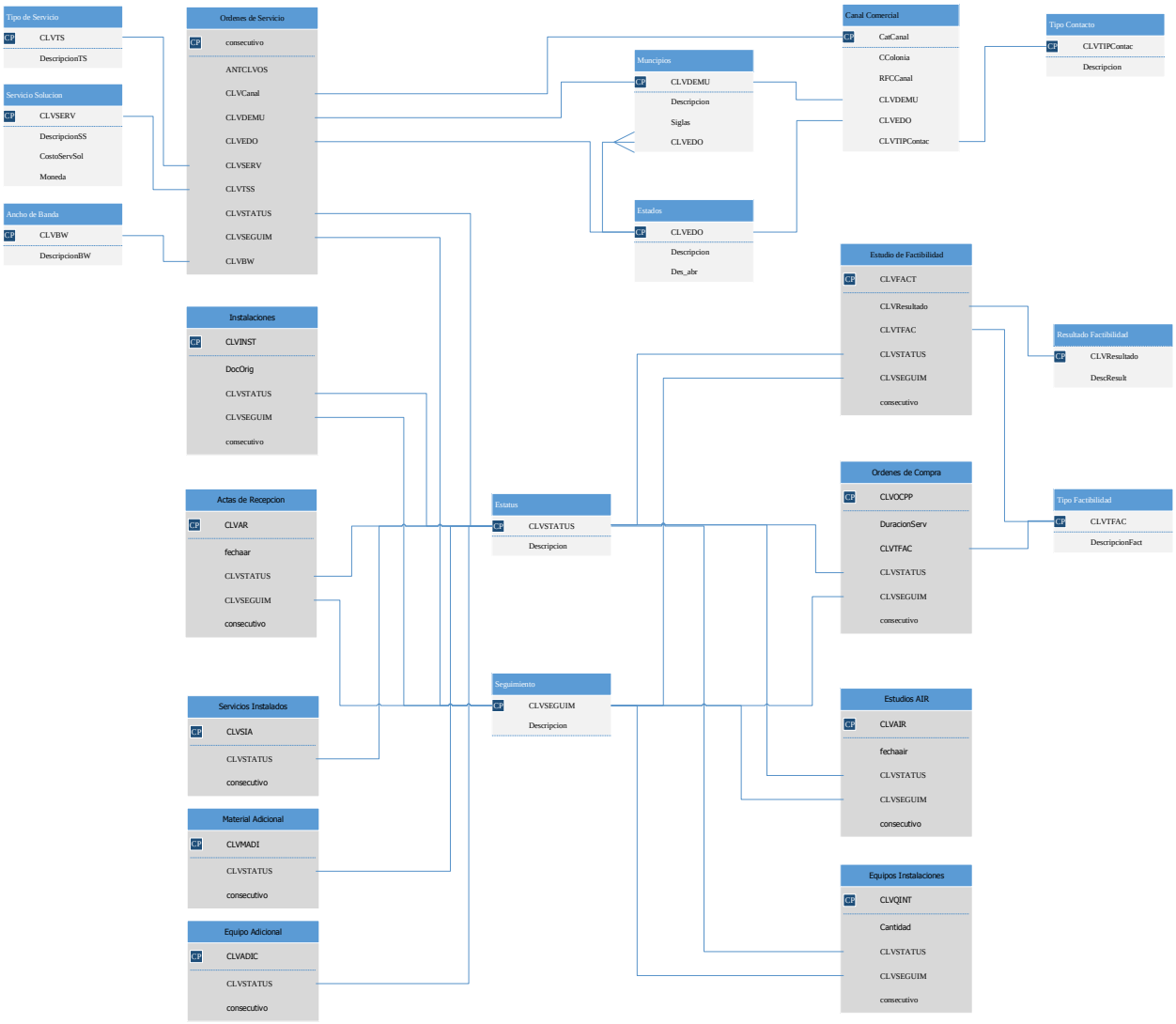


Figura 4.5.1 Diagrama ER

Ordenes de Servicio

Figura 4.5.2

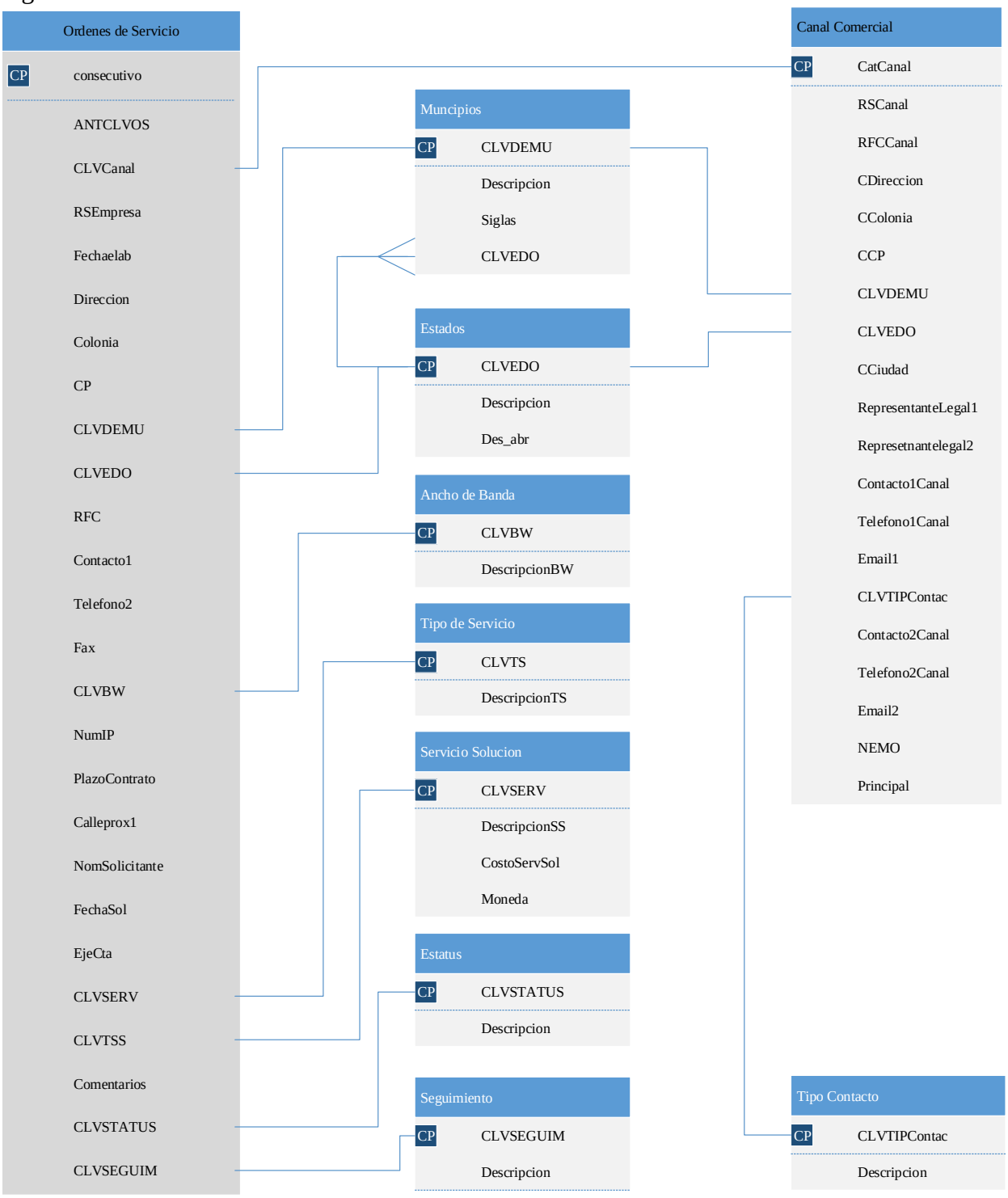


Figura 4.5.2 Ordenes de Servicio

Estudio Factibilidad

Figura 4.5.3

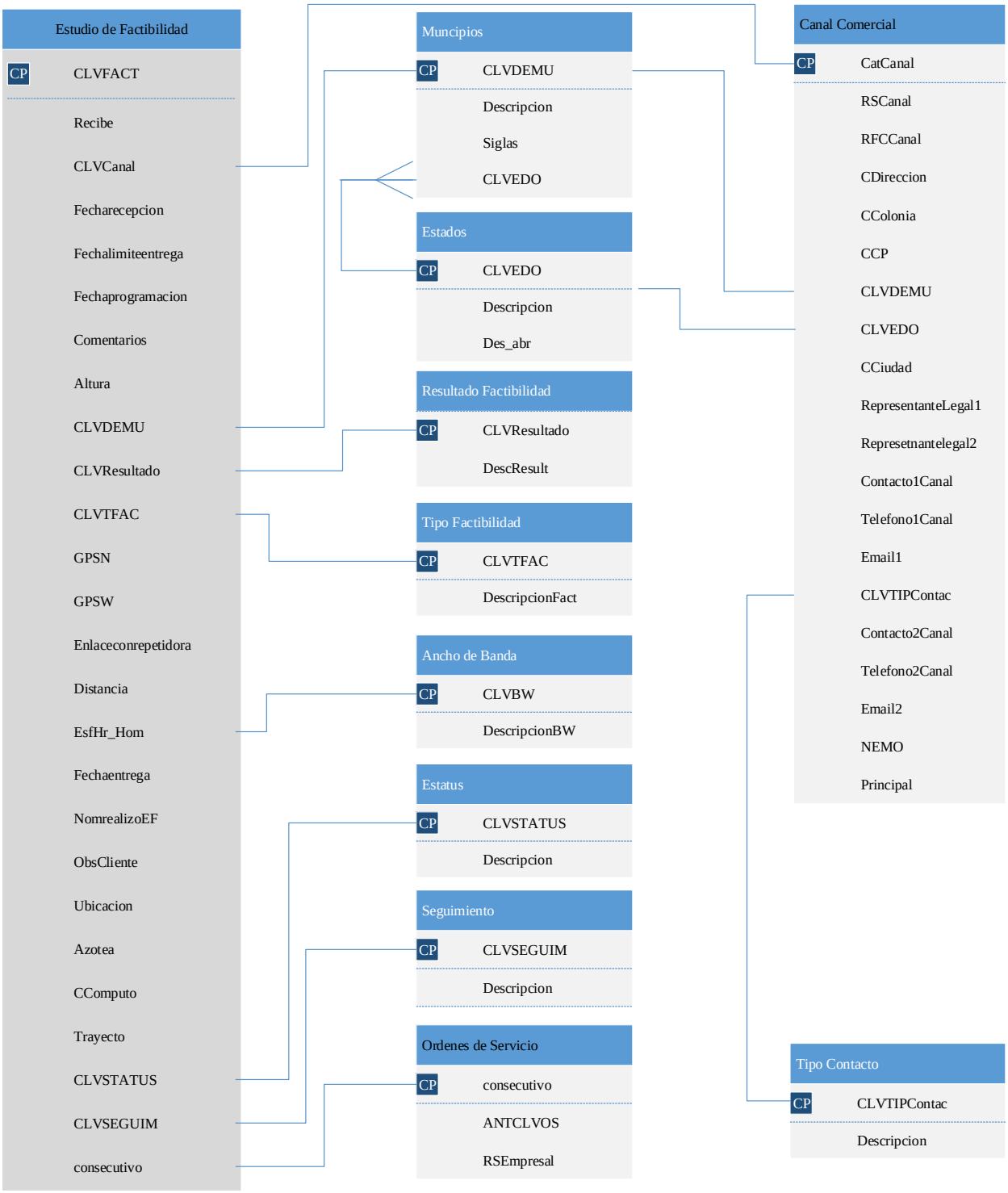


Figura 4.5.3 Estudio Factibilidad

Órdenes de Compra

Figura 4.5.4

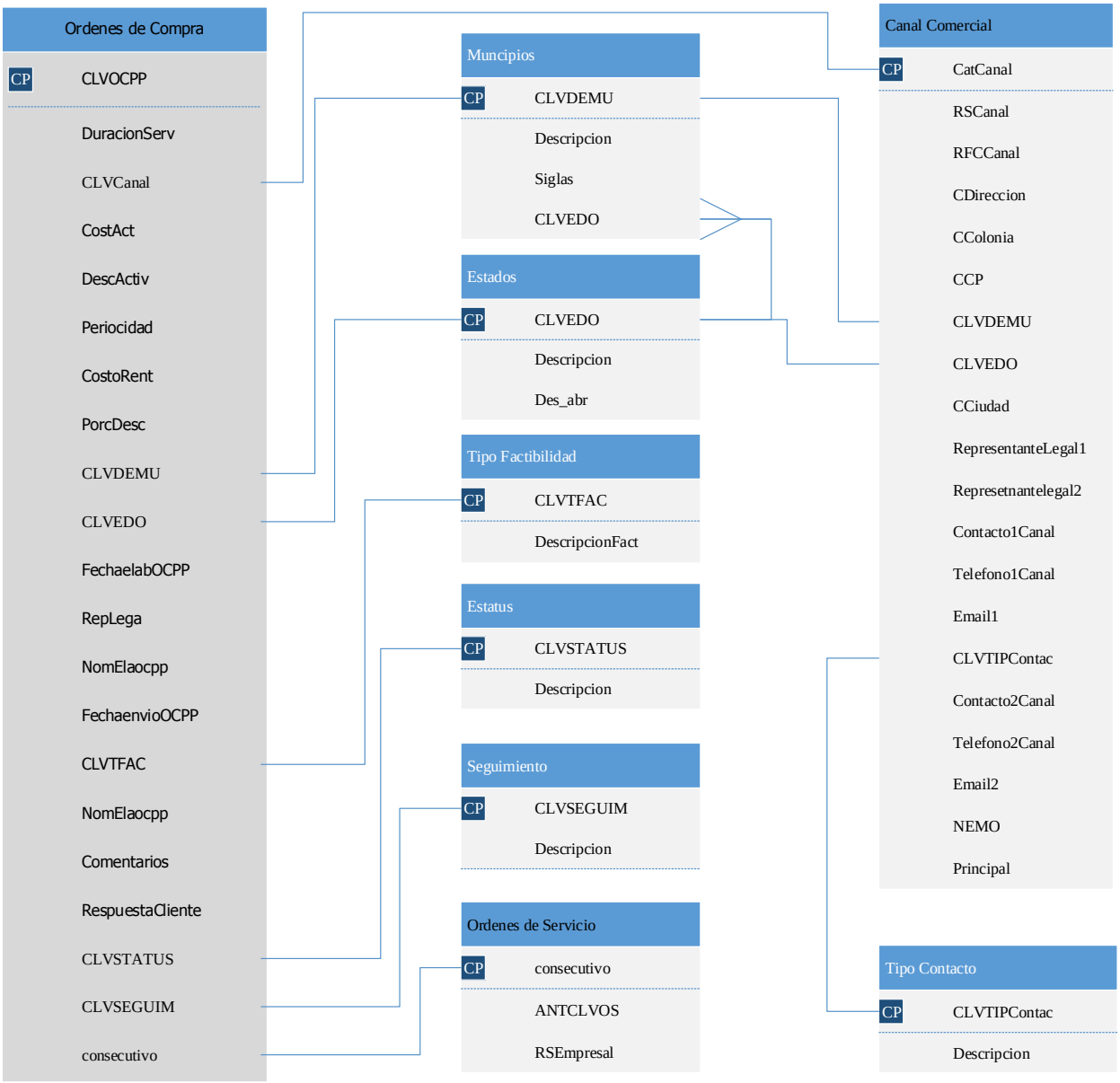


Figura 4.5.4 Órdenes de Compra

Instalaciones

Figura 4.5.5

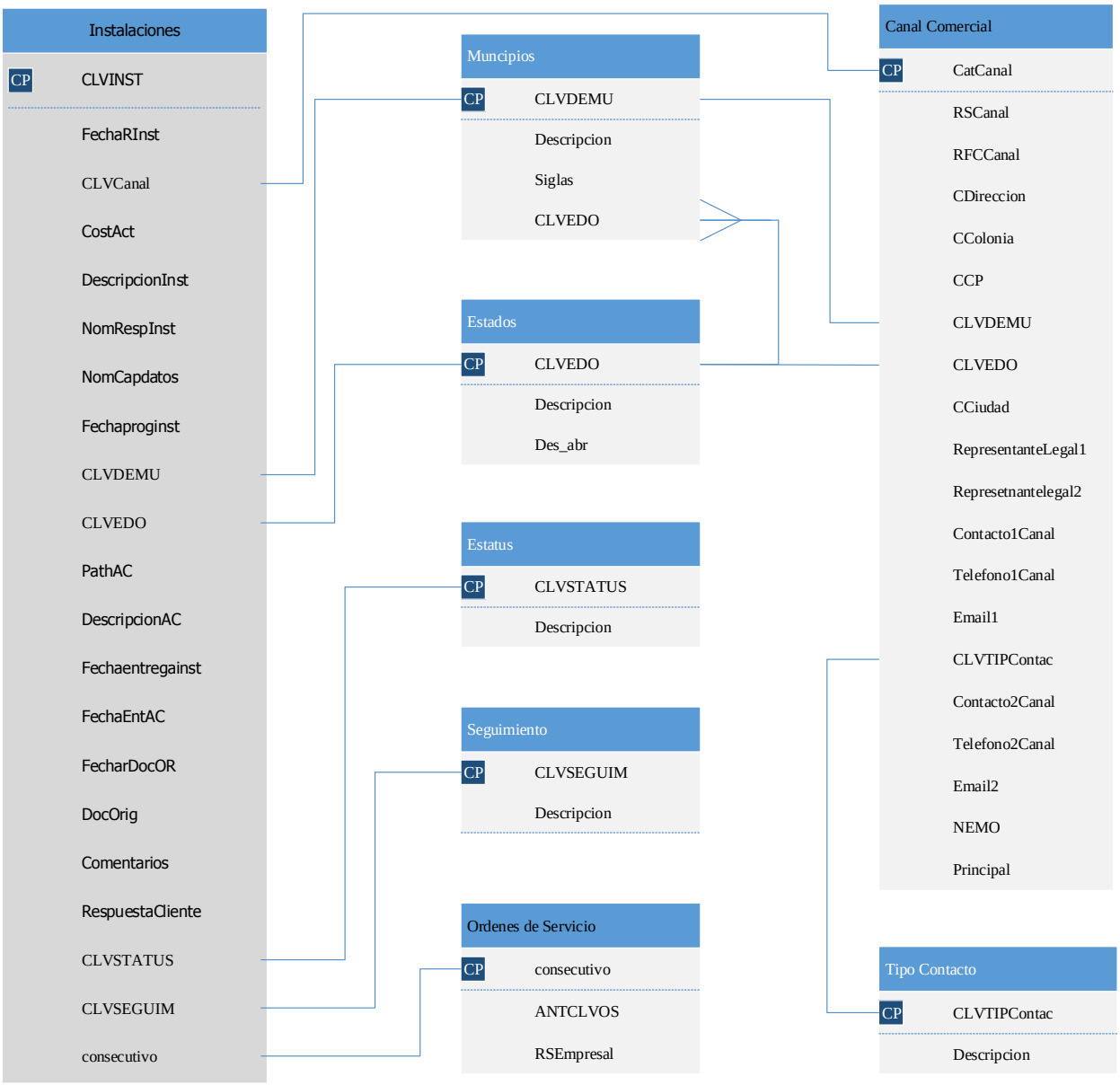


Figura 4.5.5 Instalaciones

Actas de Recepción

Figura 4.5.6

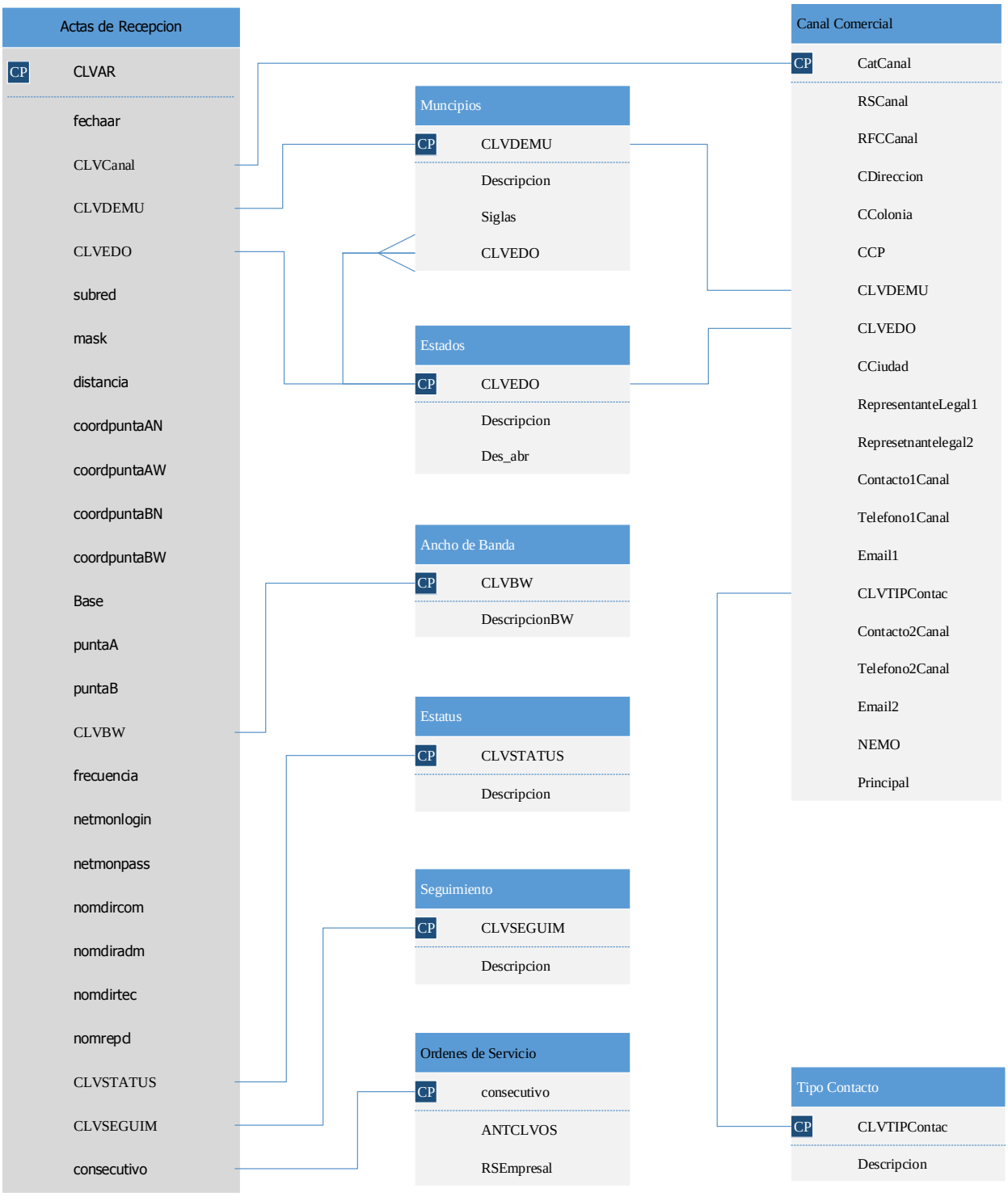


Figura 4.5.6 Actas de Recepción

Configuración

Figura 4.5.7

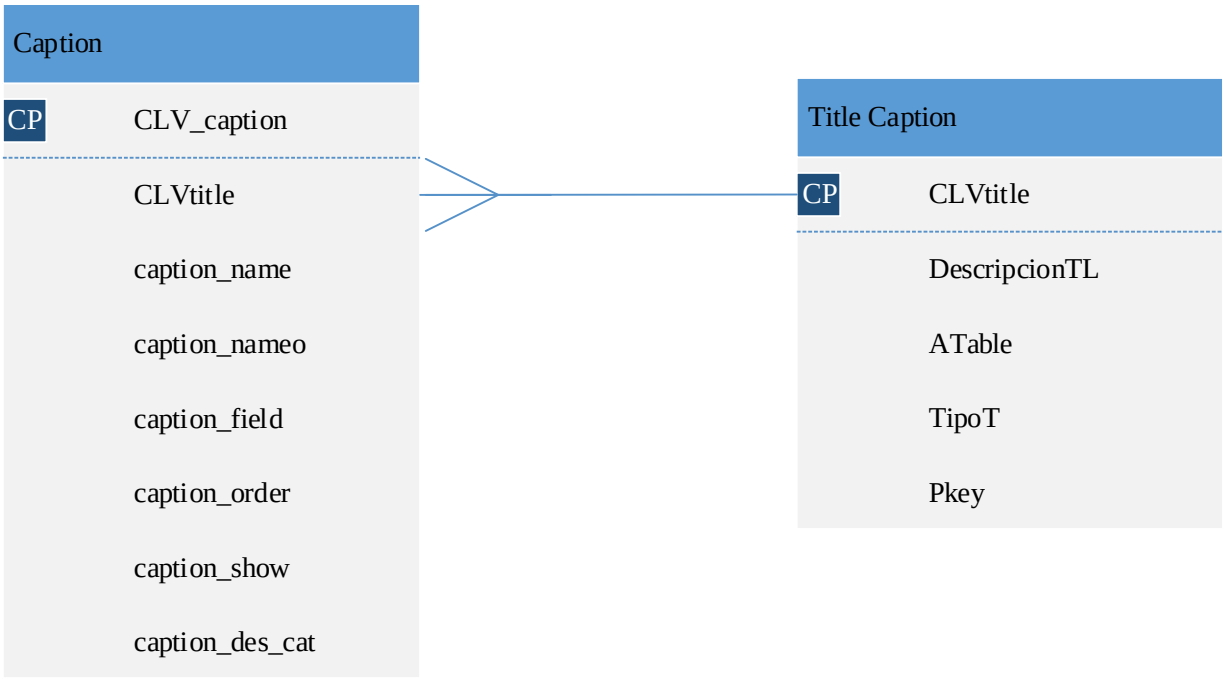


Figura 4.5.7 Configuración

6. Normalización

Como se mencionó anteriormente en las bases teóricas una normalización consiste en un proceso mediante el cual se transforman datos complejos a un conjunto de estructuras de datos más pequeñas; que además son más simples y más estables y por tanto más fáciles de mantener.

A manera de resumen

Forma Normal	Descripción
Primera	Incluye la eliminación de todos los grupos repetidos.
Segunda	Asegura que todas las columnas que no son llave sean completamente dependientes de la llave primaria.
Tercera	Elimina cualquier dependencia transitiva. Una dependencia transitiva es aquella en la cual las columnas que no son llave son dependientes de otras columnas que tampoco son llave

Figura 4.6.1 Formas Normales

Los pasos a seguir para la normalización son:

Primera Forma normal

- Eliminar los grupos repetidos.
- Crear una nueva tabla con la llave primaria de la tabla base y el grupo repetido

Segunda Forma Normal

- Determinar cuáles columnas que no son llave no dependen de la llave primaria.
- Eliminar esas columnas de la tabla.
- Crear una segunda tabla con esas columnas y la columna de la llave primaria de la cual dependen.

Tercera forma normal

- Determinar las columnas que son dependientes de otra columna no llave
- Eliminar esas columnas de la tabla.
- Crear una segunda tabla con esas columnas y con la columna no llave de la cual son dependientes.

Como ejemplo de las reglas de normalización Normalizaremos una Orden de servicio. Cabe aclarar que previo a este sistema algunos documentos ahora identificados, estaban mezclados con la orden misma y no permitían la separación de los procesos.

Normalización de una Orden de servicio

ANTCLVOS	RSCanal	RSEmpresa	Fechaelab	RFC
ADEOCT02	ADETEL	SALLES SAINZ GRANT THORTON	01-Jan-02	SSG980506-U65
ADEOCT01	ADETEL	GLOBAL NT S.A DE C.V	02-Oct-02	GONT980125
ADEOCT01	ADETEL	GLOBAL NT S.A DE C.V	02-Oct-02	GONT980125
VNMOCT01	VIA NETWORKS MEXICO	CENTRO EDUCATIVO CELTIC A.S.	10-Apr-02	CEEC850114-H25
ABAOCT01	Comunicacion S.A. COMSA	Constructora ATCO S.A. de C.V.	10-Apr-02	COAT900510-K25
ABAOCT01	Comunicacion S.A. COMSA	Constructora ATCO S.A. de C.V.	10-Apr-02	COAT900510-K25
QNESEP01	QUICKNET	AVANTE INGENIEROS S.A. de C.V. (Edo. De México)	09-Jun-02	AAIN790501-P74
VNMSEP01	VIA NETWORKS MEXICO	Grupo Industrial del Parque	22-Aug-02	GUIP960701-V11
VNMSEP04	VIA NETWORKS MEXICO	Johann A. Krause S.A. DE C.V.	26-Aug-02	JKM990609JI7
VNMSEP04	VIA NETWORKS MEXICO	Johann A. Krause S.A. DE C.V.	26-Aug-02	JKM990609JI7
VNMSEP03	VIA NETWORKS MEXICO	Grupo Industrial del Parque	20-Sep-02	GUIP960701-V11
VNMSEP03	VIA NETWORKS MEXICO	Grupo Industrial del Parque	20-Sep-02	GUIP960701-V11

Contacto1	Telefono1	BW	NumIP	Plazo	NomSolicitante
JOSÉ CARLOS TORRES	54246500	0	2	un año	Marcos A. Robles H.
RESALIÓ VERA	55789196	64 linea	1	un año	Roberto Morales
RESALIÓ VERA	55789196	64 linea	1	un año	Roberto Morales
LIC. PARIS OYOLA	228 52 96 - 98	0	1	un año	Maricela Aguilar Tel. 5629 8102
Arq. Fernando Arcos	9	512	1	un año	José Flores (55) 52728813 ext 650
Arq. Fernando Arcos	9	512	1	un año	José Flores (55) 52728813 ext 650
Ing. Carlos Galindo (Gte. Informática)	55 65 38 55	0	1	un año	Georgina Alonso
Ing. Maximino González	01427- 272 7922	64 linea	1	un año	Maricela Aguilar Illán Tel. 56298102
ING. ARTURO DÍAZ	01442 2 215 77 47	128	1	un año	Maricela Aguilar 56298102 Nextel 1839-49
ING. ARTURO DÍAZ	01442 2 215 77 47	128	1	un año	Maricela Aguilar 56298102 Nextel 1839-49
Ing. Maximino González	01427 2727922	0	1	un año	Maricela Aguilar Illán Tel. 56298102
Ing. Maximino González	01427 2727922	0	1	un año	Maricela Aguilar Illán Tel. 56298102

FechaSol	SERVICIO	TSS	Comentarios	Status	Seguim
01-Jan-03	NORMAL	Factibilidad de línea de vista	NA	Solicitado	ACEPTADO
01-Jan-03	NORMAL	Desmontar enlace	NA	Solicitado	ACEPTADO
01-Jan-03	NORMAL	Desmontar enlace	NA	Solicitado	ACEPTADO
04-Oct-02	NORMAL	Factibilidad de línea de vista	NA	Solicitado	ACEPTADO
04-Oct-02	NORMAL	Factibilidad de línea de vista	NA	Solicitado	ACEPTADO
04-Oct-02	NORMAL	Factibilidad de línea de vista	NA	Solicitado	ACEPTADO
01-Jan-03	NORMAL	Factibilidad de línea de vista	NA	Solicitado	ACEPTADO
05-Sep-02	NORMAL	Factibilidad de línea de vista	NA	Solicitado	ACEPTADO
26-Sep-02	NORMAL	Factibilidad de línea de vista	condicionado a la terminación de las instalaciones allá en el PIBQ, este enlace es especial	Solicitado	ACEPTADO
26-Sep-02	NORMAL	Factibilidad de línea de vista	condicionado a la terminación de las instalaciones allá en el PIBQ, este enlace es especial	Solicitado	ACEPTADO
20-Sep-02	NORMAL	Factibilidad de línea de vista	NA	Solicitado	ACEPTADO
20-Sep-02	NORMAL	Factibilidad de línea de vista	NA	Solicitado	ACEPTADO

Estado	Municipio	fecha_elab	Cant	descrpeq	No serie
DISTRITO FEDERAL	CUAUHTÉMOC	01-Jan-03	1	radio SM	SM-123459826-482
DISTRITO FEDERAL	CUAUHTÉMOC	01-Jan-03	1	radio SM	SM-123459826-407
DISTRITO FEDERAL	CUAUHTÉMOC	01-Jan-03	1	plato Reflector	123459826-407
QUERÉTARO ARTEAGA	QUERÉTARO	04-Oct-02	1	radio SM	SM-123459826-487
QUERÉTARO ARTEAGA	QUERÉTARO	04-Oct-02	1	radio SM	SM-123459826-448
QUERÉTARO ARTEAGA	QUERÉTARO	04-Oct-02	1	plato Reflector	123459826-448
MÉXICO	TLALNEPANTLA DE BAZ	01-Jan-03	1	radio SM	SM-123459826-485
QUERÉTARO ARTEAGA	SAN JUAN DEL RIO	05-Sep-02	1	radio SM	SM-123459826-462
QUERÉTARO ARTEAGA	QUERÉTARO	26-Sep-02	1	radio BH	BH-123459826-400
QUERÉTARO ARTEAGA	QUERÉTARO	26-Sep-02	1	plato Reflector	123459826-400

QUERÉTARO ARTEAGA	QUERÉTARO	20-Sep-02	1	radio AP	AP-123459826-417
QUERÉTARO ARTEAGA	QUERÉTARO	20-Sep-02	1	plato Reflector	123459826-417

Capacidad	Instaladores	Comentarios	Estatus inst.
2	MAAN	NP	Solicitado
2	MAAN	NP	Solicitado
0	Raul Ceballos	NP	Solicitado
2	MAAN	NP	Solicitado
2	MAAN	NP	Solicitado
0	MAAN	NP	Solicitado
2	MAAN	NP	Solicitado
2	MAAN	NP	Solicitado
2	MAAN	Por capacidad	Solicitado
0	Raul Ceballos	Por capacidad	Solicitado
2	Raul Ceballos	NP	Solicitado
0	Raul Ceballos	NP	Solicitado

Figura 4.6.2 Datos Orden de Servicio Des normalizada

Primera Forma normal

- Eliminar los grupos repetidos.
- Crear una nueva tabla con la llave primaria de la tabla base y el grupo repetido

De donde salen dos grupos
El de orden de servicio y equipo de instalaciones

Orden de servicio

ANTCLVOS	RSCanal	RSEmpresa	Fechaelab	RFC
ADEOCT02	ADETEL	SALLES SAINZ GRANT THORTON	01-Jan-02	SSG980506-U65
ADEOCT01	ADETEL	GLOBAL NT S.A DE C.V	02-Oct-02	GONT980125
VNMOCT01	VIA NETWORKS MÉXICO	CENTRO EDUCATIVO CELTIC A.S.	10-Apr-02	CEEC850114-H25
ABAOCT01	Comunicacion S.A. COMSA	Constructora ATCO S.A. de C.V.	10-Apr-02	COAT900510-K25
QNESEP01	QUICKNET	AVANTE INGENIEROS S.A. de C.V. (Edo. De México)	09-Jun-02	AAIN790501-P74
VNMSEP01	VIA NETWORKS MÉXICO	Grupo Industrial del Parque	22-Aug-02	GUIP960701-V11
VNMSEP04	VIA NETWORKS MÉXICO	Johann A. Krause S.A. DE C.V.	26-Aug-02	JKM990609JI7
VNMSEP03	VIA NETWORKS MÉXICO	Grupo Industrial del Parque	20-Sep-02	GUIP960701-V11

Contacto1	Telefono1	BW	NumIP	Plazo	NomSolicitante
JOSÉ CARLOS TORRES	54246500	0	2	un año	Marcos A. Robles H.
ROSALIO VERA	55789196	64 linea	1	un año	Roberto Morales
LIC. PARIS OYOLA	228 52 96 - 98	0	1	un año	Maricela Aguilar Tel. 5629 8102
Arq. Fernando Arcos	9	512	1	un año	José Flores (55) 52728813 ext 650
Ing. Carlos Galindo (Gte. Informática)	55 65 38 55	0	1	un año	Georgina Alonso
Ing. Maximino González	01427- 272 7922	64 linea	1	un año	Maricela Aguilar Illán Tel. 56298102
ING. ARTURO DIAZ	01442 2 215 77 47	128	1	un año	Maricela Aguilar 56298102 Nextel 1839-49
Ing. Maximino Gonzalez	01427 2727922	0	1	un año	Maricela Aguilar Illán Tel. 56298102

FechaSol	SERVICIO	TSS	Comentarios	Status	Seguim
01-Jan-03	NORMAL	Factibilidad de línea de vista	NA	Solicitado	ACEPTADO

01-Jan-03	NORMAL	Desmontar enlace	NA	Solicitado	ACEPTADO
04-Oct-02	NORMAL	Factibilidad de línea de vista	NA	Solicitado	ACEPTADO
04-Oct-02	NORMAL	Factibilidad de línea de vista	NA	Solicitado	ACEPTADO
01-Jan-03	NORMAL	Factibilidad de línea de vista	NA	Solicitado	ACEPTADO
05-Sep-02	NORMAL	Factibilidad de línea de vista	NA	Solicitado	ACEPTADO
26-Sep-02	NORMAL	Factibilidad de línea de vista	condicionado a la terminación de las instalaciones allá en el PIBQ, este enlace es especial	Solicitado	ACEPTADO
20-Sep-02	NORMAL	Factibilidad de línea de vista	NA	Solicitado	ACEPTADO

Estado	Municipio
DISTRITO FEDERAL	CUAUHTÉMOC
DISTRITO FEDERAL	CUAUHTÉMOC
QUERÉTARO ARTEAGA	QUERÉTARO
QUERÉTARO ARTEAGA	QUERÉTARO
MÉXICO	TLALNEPANTLA DE BAZ
QUERÉTARO ARTEAGA	SAN JUAN DEL RIO
QUERÉTARO ARTEAGA	QUERÉTARO
QUERÉTARO ARTEAGA	QUERÉTARO

Figura 4.6.3 Datos Orden de Servicio pre normalizada

Equipo de Instalaciones

ANTCLVOS	fecha_elab	Cant	descrpeq	No serie	Capacidad	Instalores
ADEOCT02	01-Jan-03	1	radio SM	SM-123459826-482	2	MAAN
ADEOCT01	01-Jan-03	1	radio SM	SM-123459826-407	2	MAAN
ADEOCT01	01-Jan-03	1	plato Reflector	123459826-407	0	Raul Ceballos
VNMOCT01	04-Oct-02	1	radio SM	SM-123459826-487	2	MAAN
ABAOCT01	04-Oct-02	1	radio SM	SM-123459826-448	2	MAAN
ABAOCT01	04-Oct-02	1	plato Reflector	123459826-448	0	MAAN
QNESEP01	01-Jan-03	1	radio SM	SM-123459826-485	2	MAAN
VNMSEP01	05-Sep-02	1	radio SM	SM-123459826-462	2	MAAN
VNMSEP04	26-Sep-02	1	radio BH	BH-123459826-400	2	MAAN
VNMSEP04	26-Sep-02	1	plato Reflector	123459826-400	0	Raul Ceballos
VNMSEP03	20-Sep-02	1	radio AP	AP-123459826-417	2	Raul Ceballos
VNMSEP03	20-Sep-02	1	plato Reflector	123459826-417	0	Raul Ceballos

Comentarios	Estatus inst.
NP	Solicitado
NP	Solicitado
NP	Solicitado
NP	Solicitado
NP	Solicitado
NP	Solicitado
NP	Solicitado
NP	Solicitado
Por capacidad	Solicitado
Por capacidad	Solicitado
NP	Solicitado
NP	Solicitado

Figura 4.6.4 Datos Equipos de Instalaciones pres normalizada

Segunda Forma Normal

- Determinar cuáles columnas que no son llave no dependen de la llave primaria.
- Eliminar esas columnas de la tabla.
- Crear una segunda tabla con esas columnas y la columna de la llave primaria de la cual dependen.

Tenemos para la Orden de servicio

PK	ANTCLVOS	CLVCanal	RSEmpresa	Fechaelab	RFC
50	ADEOCT02	1	SALLES SAINZ GRANT THORTON	01-Jan-02	SSG980506-U65
51	ADEOCT01	1	GLOBAL NT S.A DE C.V	02-Oct-02	GONT980125
52	VNMOCT01	9	CENTRO EDUCATIVO CELTIC A.S.	10-Apr-02	CEEC850114-H25
53	ABAOCT01	4	Constructora ATCO S.A. de C.V.	10-Apr-02	COAT900510-K25
55	QNESEP01	8	AVANTE INGENIEROS S.A. de C.V. (Edo. De México)	09-Jun-02	AAIN790501-P74
58	VNMSEP01	9	Grupo Industrial del Parque	22-Aug-02	GUIP960701-V11
59	VNMSEP04	9	Johann A. Krause S.A. DE C.V.	26-Aug-02	JKM990609JI7
60	SAMJUL14	2	GRAFICA VILLALBA S.A. DE C.V.	07-Oct-02	GAVI860921-W78

Contacto1	Telefono1	clvBW	NumIP	PlazoContrato	NomSolicitante
JOSÉ CARLOS TORRES	54246500	1	2	un año	Marcos A. Robles H.
ROSALIO VERA	55789196	2	1	un año	Roberto Morales
LIC. PARIS OYOLA	228 52 96 - 98	1	1	un año	Maricela Aguilar
Arq. Fernando Arcos	9999999	5	1	un año	José Flores
Ing. Carlos Galindo (Gte. Informática)	55 65 38 55	1	1	un año	Georgina Alonso
Ing. Maximino González	01427- 272 7922	2	1	un año	Maricela Aguilar Illán Tel. 56298102
ING. ARTURO DIAZ	01442 2 215 77 47	3	1	un año	Maricela Aguilar 56298102 Nextel 1839-49
PABLO FIGUEROA	58 15 77 77	3	1	un año	Manuel Alatorre

FechaSol	clvser	CLVTSS	Comentarios	CLVStatus	CLVSeguim
01-Jan-03	1	1	NA	1	1
01-Jan-03	1	5	NA	2	1
04-Oct-02	1	1	NA	1	1
04-Oct-02	1	1	NA	1	1
01-Jan-03	1	1	NA	1	1
05-Sep-02	1	1	NA	1	1
26-Sep-02	3	1	es factible condicionado terminación l PIBQ,	1	1
01-Jan-03	1	1	NA	1	1

CLVEstado	CLVMunicipio
9	9015
9	9015
22	22014
22	22014
15	15104
22	22016
22	22014
9	9004

Figura 4.6.5 Datos Orden de Servicio normalizada

Para la cual se identificaron RSCanal, BW, Plazo contrato, Nombre del solicitante, Servicio, Tipo de servicio, Estatus, Seguimiento, Estado, Municipio y se crearon:

Para el Canal Comercial

Clave	Razón S.	RFC	Estado	CLVMUN	NEMO
1	ADETEL	ADT970311 7PA	D.F.	Miguel Hidalgo	ADE
2	Servicios Administrados MEXIS	SAM 870311 C20	D.F.	Col. Granjas de San Antonio	SAM
4	Comunicacion S.A. COMSA	COM 920701 QXA	D.F.	Del Valle	ABA
5	Diveo INTERNET DE MEXICO	DIM 911024 SP8	D.F.	Doctores	DIV
6	GLOBAL	GLO 920313DF9	D.F.	San Miguel Chapultepec	GBT
7	METRORED	MTR 861219 6R5	D.F.	Col Roma	MTR
8	QUICKNET	QIT8401118A6	D.F.	Industrial	QNE

Figura 4.6.6 Datos Canal Comercial Des normalizada

Nuevamente para estado y Municipio del canal.
Tercera forma normal

- Determinar las columnas que son dependientes de otra columna no llave
- Eliminar esas columnas de la tabla.
- Crear una segunda tabla con esas columnas y con la columna no llave de la cual son dependientes.

En este caso Estado y municipio

Clave	Razón S.	RFC	CLVEstado	CLVMUN	NEMO
1	ADETEL	ADT970311 7PA	9	9016	ADE
2	Servicios Administrados MEXIS	SAM 870311 C20	9	9016	SAM
4	Comunicacion S.A. COMSA	COM 920701 QXA	9	9014	ABA
5	Diveo INTERNET DE MÉXICO	DIM 911024 SP8	9	9015	DIV
6	GLOBAL	GLO 920313DF9	9	9016	GBT
7	METRORED	MTR 861219 6R5	9	9015	MTR
8	QUICKNET	QIT8401118A6	9	9005	QNE

Figura 4.6.7 Datos Canal Comercial normalizada

Para el Ancho de banda

Clave	Descripción
1	0
2	64 linea
3	128
4	256
5	512
6	1024
7	2048

Figura 4.6.8 Datos Ancho de Banda

Para tipos de servicio urgencia

Clave	Descripción
1	NORMAL
2	REGULAR
3	URGENTE

Figura 4.6.9 Datos Tipo de servicio

Para Servicio solicitado

Clave	Descripción	TiempoE
1	Factibilidad de línea de vista	1
2	ELVIS completo	3
3	Instalar enlace	2
4	Calibrar enlace	1
5	Desmontar enlace	1
1	Factibilidad de línea de vista	1

Figura 4.6.10 Datos Servicio solicitado

Para el estatus de la solicitud

Estatus	Descripción
1	Solicitado
2	Rechazado
3	Aceptada
4	Modificado
5	En proceso
6	Factible

Figura 4.6.11 Datos Estatus seguimiento

Para el estado y municipio

EDO	Descripción	Des_abr
1	AGUASCALIENTES	AGS
9	DISTRITO FEDERAL	DF
11	GUANAJUATO	GTO
13	HIDALGO	HGO
14	JALISCO	JAL
15	MÉXICO	MEX
16	MICHOACÁN DE OCAMPO	MICH
17	MORELOS	MOR
19	NUEVO LEÓN	NL
21	PUEBLA	PUE
22	QUERÉTARO ARTEAGA	QRO

Figura 4.6.12 Datos Estados

Municipio

MUN	Descripcion	Siglas	CLVEDO
1001	AGUASCALIENTES	AGES	1
1002	ASIENTOS	ASOS	1
1003	CALVILLO	CALO	1
1004	COSIO	COIO	1

Figura 4.6.13 Datos Municipios

Para el grupo de Equipos de Instalaciones:

CLVQINT	PK	fecha_elab	Cant	descrpeq	No serie	Capacidad
54	50	01-Jan-03	1	radio SM	SM-123459826-482	2
68	51	01-Jan-03	1	radio SM	SM-123459826-407	2
94	51	01-Jan-03	1	plato Reflector	123459826-407	0
59	52	04-Oct-02	1	radio SM	SM-123459826-487	2
20	53	04-Oct-02	1	radio SM	SM-123459826-448	2
95	53	04-Oct-02	1	plato Reflector	123459826-448	0
57	55	01-Jan-03	1	radio SM	SM-123459826-485	2
34	58	05-Sep-02	1	radio SM	SM-123459826-462	2
35	59	26-Sep-02	1	radio BH	BH-123459826-400	2
98	59	26-Sep-02	1	plato Reflector	123459826-400	0
60	60	20-Sep-02	1	radio AP	AP-123459826-417	2
99	60	20-Sep-02	1	plato Reflector	123459826-417	0

Instalores	Comentarios	CLVStatus
MAAN	NP	1
MAAN	NP	1
Raul Ceballos	NP	1
MAAN	NP	1
MAAN	NP	1
MAAN	NP	1
MAAN	NP	1
MAAN	NP	1
MAAN	Por capacidad	1
Raul Ceballos	Por capacidad	1
Raul Ceballos	NP	1
Raul Ceballos	NP	1

Figura 4.6.14 Datos Equipos de Instalaciones

Y para la tercera forma normal obtenemos cuatro tablas tablas

- Datos Instalaciones
- Equipos instalados
- Tipos de equipos
- Instaladores

Datos Instalaciones

CLVQINT	PK	fecha_elab	Cant	CLVEQ	Capacidad	CLVEMP	Comentarios	CLVStatus
54	50	01-Jan-03	1	1001	2	100	NP	1
68	51	01-Jan-03	1	1002	2	100	NP	1
94	51	01-Jan-03	1	1003	0	200	NP	1
59	52	04-Oct-02	1	1004	2	100	NP	1
20	53	04-Oct-02	1	1005	2	100	NP	1
95	53	04-Oct-02	1	1006	0	100	NP	1
57	55	01-Jan-03	1	1007	2	100	NP	1
34	58	05-Sep-02	1	1008	2	100	NP	1
35	59	26-Sep-02	1	1009	2	100	capacidad	1
98	59	26-Sep-02	1	1010	0	200	capacidad	1
60	60	20-Sep-02	1	1011	2	200	NP	1
99	60	20-Sep-02	1	1012	0	200	NP	1

Figura 4.6.15 Datos de Instalaciones

Equipos Instalados

CLVEQ	No serie	CLVTEQ
1001	SM-123459826-482	1
1002	SM-123459826-407	1
1003	123459826-407	5
1004	SM-123459826-487	1
1005	SM-123459826-448	1
1006	123459826-448	5
1007	SM-123459826-485	1
1008	SM-123459826-462	1
1009	BH-123459826-400	3
1010	123459826-400	5
1011	AP-123459826-417	2
1012	123459826-417	5

Figura 4.6.16 Datos Equipos Instalados

Instaladores

CLVEMP	Nombre	Inicales	ROL
100	MANUEL ALVAREZ NATIVIDAD	MAAN	Istalador
200	RAUL CEBALLOS GASPARIN	GASP	Instalador
300	ENRIQUE ALEGRE ALVAREZ	ENAA	Instalador

Figura 4.6.17 Instaladores

Tipos de equipo

CLVTEQ	DESCRIPCION	NEMO
100	SUBSCRIBER MODULE	SM
200	ACCES POINT	AP
300	BACK HAUL	BH
500	PLATO REFLECTOR	PR

Figura 4.6.18 Tipo de Equipo

Muchas de las tablas fueron desnormalizadas por conveniencias del sistema.

La desnormalizacion puede utilizarse para lograr eficiencias en la aplicación, la base, la forma de acceso o diversos procesos. En un modelo normalizado, comunmente se encuentran consultas que se realizan en base a varias tablas ligadas que llegan ha ser muy complejas, si consideramos además que una unión compleja puede producir tablas temporales de trabajo, se necesita considerar la utilización de algún grado de desnormalizacion. Por supuesto considerando siempre el motivo de la normalización y la integridad de los datos.

7. Implementación del Back End

Para la preparación del Back End se requiere de la instalación de:

- Sistema Operativo
- El servidor web
- Componentes adicionales
- Manejador de Base de datos
- Herramientas de administración y configuración
- Los directorios y archivos de la aplicación

Para la aplicación se configuró con

Windows server 2008 r2, IIS versión 6, ASP y ASP.NET, componentes .Net, SQL server 2008, Visual Studio, MMC, y la consola de administración de base de datos SQL Server Management Studio (SSMS), Drivers de conexión OLE DB y SQL server, servicios FTP, entre otros.

Ejemplos de estas configuraciones sin detallar.

Windows server
El sistema operativo donde se instaló la aplicación es Windows server 2008 r2. Es el SO proporcionado por la empresa.

Edición
Essentials
Entorno
Entornos de pequeña empresa para servidores con dos procesadores como máximo
Interfaz más sencilla, conectividad pre configurada con servicios basados en la nube; una instancia virtual de Essentials
Modelo de licencias
Server (límite de 25 usuarios)

Instalación del web Server y ASP.net

La instalación del web server requiere la instalación del IIS (Internet Information Server) y los módulos de ASP.net, además de su configuración y los archivos de configuración de la aplicación.
Un ASP.Net web server es una extensión del IIS al que se le integra el modelo ASP.NET en el core del servidor.
De manera que al modelo de IIS se le adicionan a groso modo

- ASP.net
- .NET framework
- ISAPI

Para soportar aplicaciones de .NET 2 y .NET 3.5 se instaló .NET 3.5

Para configurar una aplicación ASP.NET es necesario configurar al menos

- **Alias**
- **Application pool**
- **Physical path**

Es decir un alias o nombre de la aplicación, un directorio lógico de la aplicación o una sección donde se encuentra la aplicación y la dirección física donde se encuentra la aplicación.

Todos los parámetros de configuración tanto IIS como ASP y la aplicación se pueden configurar mediante el uso del Microsoft Management Console (MMC).

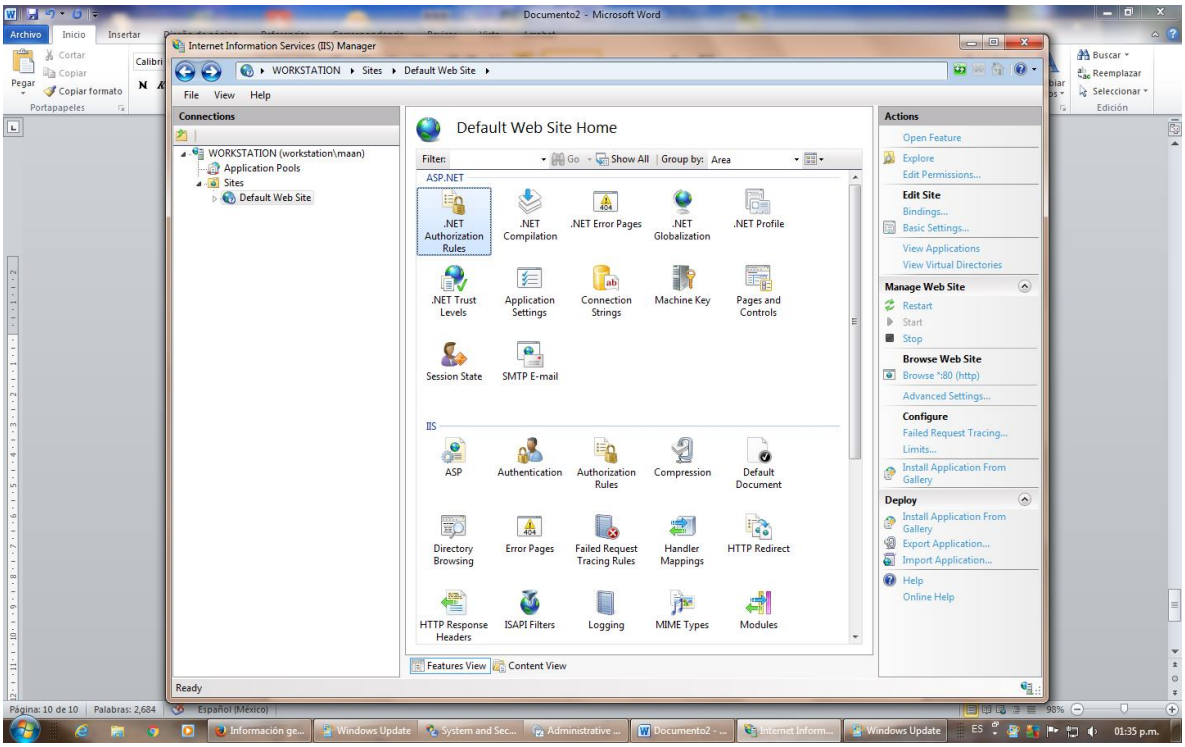


Figura 4.7.1 Consola IIS

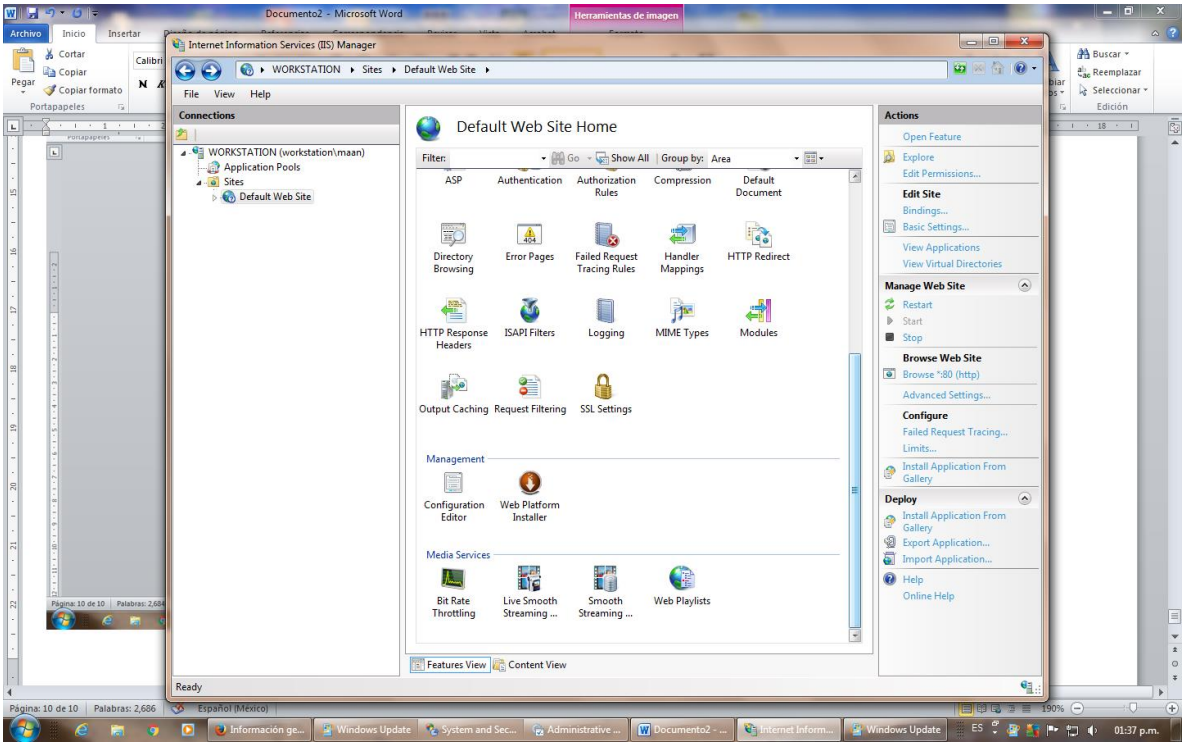


Figura 4.7.2 Consola IIS 2

Una vez configurado el servidor web configuramos el SQL server en nuestro caso:

SQL Server

Version	Año	Nombre	Alias
10.5	20010	SQL Server 2008	Kilimanjaro

Figura 4.7.3 SQL Server

Creación de la base de datos

Algunos de los datos más relevantes de la creación de la base de datos son:

- Nombre de la base de datos
- El archivo y directorio de la base de datos
- El incremento automático de la base de datos.
- La creación del usuario o usuarios de la base de datos y sus privilegios
- La creación de las tablas y vistas.

Todas estas actividades se pueden realizar usando la interface grafica del el manejador de base de datos, SQL Server Management Studio (SSMS) o el DDL.
Estos son unos ejemplos de la creación de una tabla y unas consultas en la base de datos.

Creación de tablas

Se crea la tabla DatosAir (Análisis de Infraestructura de Red (AIR)) con sus restricciones y llaves externas.

```
CREATE TABLE DatosAir
(
CLVAIR      int    IDENTITY (1, 1) PRIMARY KEY
consecutivo int NOT NULL,
fechaair    date DEFAULT GETDATE(),
fecha_term  date DEFAULT GETDATE(),
CLVEDO      int NOT NULL DEFAULT  22,
subred      varchar     (50) DEFAULT “255” ,
mask        varchar     (50) DEFAULT “255”  ,
bw          varchar     (50) DEFAULT  “64”,
gateway     varchar     (50) DEFAULT “255”  ,
Base        varchar     (50) DEFAULT “TORRE B” ,
IP          varchar     (50) DEFAULT “255” ,
puntaA      varchar     (50) DEFAULT “TORRE B”,
puntaB      varchar     (50) DEFAULT  “TORRE B”
frecuencia  int DEFAULT 1 ,
CBQ         varchar     (50) DEFAULT “DOS” ,
netmonlogin varchar     (50) DEFAULT “no access” ,
netmonpass  varchar     (50) DEFAULT “no access” ,
IpMonitor   varchar     (50) DEFAULT “255”  ,
VisualPulse varchar     (50) DEFAULT “node 0” ,
CLVSTATUS int NOT NULL DEFAULT 1,
CLVSEGUIM int NOT NULL DEFAULT 1,
CLVCanal    int NOT NULL DEFAULT 1,
CONSTRAINT DatosAirDatosOS FOREIGN KEY (consecutivo)
REFERENCES DatosOS (consecutivo),
```

CONSTRAINT ***DatosAirCatEdo*** FOREIGN KEY (CLVEDO)
REFERENCES ***CatEdo*** (CLVEDO),
CONSTRAINT ***DatosAirCatCanal*** FOREIGN KEY (CLVCanal)
REFERENCES ***CatCanal*** (CLVCanal),
CONSTRAINT ***DatosAirCatSeguim*** FOREIGN KEY (CLVSEGUIM)
REFERENCES ***CatSeguim*** (CLVSEGUIM),
CONSTRAINT ***DatosAirCatStatus*** FOREIGN KEY (CLVSTATUS)
REFERENCES ***CatStatus*** (CLVSTATUS)
)

Consultas

Consulta inicial sobre las características del usuario y la empresa a la que pertenece.
La consulta esta parametrizada para evitar la intrusión por inserción de sql

```
txtSQL ="
SELECT
        RSCanal as canal,
        CatCanal.CLVCanal as num_canal,
        CatTipoActor.DescripcionAC,
        CatEstacion.DescripcionEs,
        empleados.CLVActor,
        CatEstacion.CLVEstacion,
        empleados.sAdmin
FROM
        CatTipoActor,
        CatEstacion,
        empleados,
        CatCanal
WHERE
        CatCanal.CLVCanal = empleados.CLVCanal
AND     IDEMPLEADO=@CID,
AND     SPASSWORD=@Cpass,
AND     CatTipoActor.CLVActor = empleados.CLVActor
AND     CatEstacion.CLVEstacion = CatTipoActor.CLVEstacion"

Parametro3.Value = ckUserID
Parametro4.Value = ckSTRpassword
command2.Parameters.Add (Parametro3)
command2.Parameters.Add (Parametro4)
NMrs = command2.ExecuteReader ()
```

Consulta Orden de servicio

Esta es la consulta máxima de los elementos de la orden de servicio. Durante el uso y configuración de la aplicación se seleccionan las columnas de más interés o en el orden que se desee trabajar.

```
SELECT
        DatosOS.consecutivo,
        DatosOS.ANTCLVOS,
        DatosOS.CLVCanal,
        CatCanal.RSCanal,
        empleados.NOMBREEMPLEADO,
        empleados.IDEMPLEADO,
        empleados.SPASSWORD,
        DatosOS.Fechaelab,
        DatosOS.Direccion,
        DatosOS.Colonia,
        DatosOS.CP,
        DatosOS.RFC,
        DatosOS.Contacto1,
        DatosOS.Telefono1,
        DatosOS.Telefono2,
        DatosOS.Fax,
        CatBW.DescripcionBW,
```

```
        DatosOS.PlazoContrato,
        DatosOS.Calleprox1,
        DatosOS.Calleprox2,
        DatosOS.NomSolicitante,
        DatosOS.EjeCta,
        DatosOS.CLVSERV,
        CatTipoServicio.DescripcionTS,
        DatosOS.CLVTSS,
        CatTipoServSol.DescripcionTSS,
        DatosOS.Comentarios,
        DatosOS.CLVSTATUS,
        CatStatus.Descripcion,
        DatosOS.CLVSEGUIM,
        CatSeguim.Descripcion,
        DatosOS.CLVEDO,
        CatEdo.Descripcion,
        DatosOS.CLVDEMU,
        CatDemu.Descripcion
FROM
        DatosOS
        CatBW
        CatCanal
        CatDemu
        CatEdo
        CatSeguim
        CatStatus
        CatTipoServicio
        CatTipoServSol
        CatTipoContact
        empleados

WHERE
        DatosOS.CLVBW = CatBW.CLVBW)
AND     DatosOS.CLVCanal = CatCanal.CLVCanal
AND     DatosOS.CLVDEMU = CatDemu.CLVDEMU
AND     DatosOS.CLVEDO = CatEdo.CLVEDO
AND     CatDemu.CLVEDO = CatEdo.CLVEDO
AND     DatosOS.CLVSEGUIM = CatSeguim.CLVSEGUIM
AND     DatosOS.CLVSTATUS = CatStatus.CLVSTATUS
AND     DatosOS.CLVSERV = CatTipoServicio.CLVSERV
AND     DatosOS.CLVTSS = CatTipoServSol.CLVTSS
AND     CatCanal.CLVTIPContac = CatTipoContact.CLVTIPContac
AND     CatCanal.CLVCanal = empleados.CLVempleado
ORDER BY
        DatosOS.consecutivo;
```

Instalación de Chartdirector

La ejecución de reportes, como se planeó en un principio, debe ser lo más independiente de la aplicación y que permita la realización de nuevos reportes o consideraciones para los mismos, por lo que se optó por un módulo pre compilado llamado ”Chartdirector”, y usar la funcionalidad y no crearla.
Para la aplicación se puso en marcha el componente dentro de la aplicación se realizaron los reportes básicos y se introdujo al personal en el manejo del mismo.

Algunas de sus características principales son:

Universal
La aplicación puede ser usada en diferentes tipos de aplicaciones (desktop, web, console, batch, tiempo real) soporta diversos lenguajes de programación sistemas operativos. .NET, C#, VB, ASP, COM, Java, C++, PHP, Perl, Python, Ruby, ColdFusion

Extensos tipos de gráficos
Circular, de anillos, barras, líneas, spline, línea de paso, la regresión, ajuste de curvas, relleno entre líneas, áreas, banda, de dispersión, de burbuja, caja flotante, caja-bigote, cascada, contorno, mapa de calor, superficie, vector, finanzas, Gantt, radiales, polares, rosa, pirámide, cono, embudo medidores y contadores.

Arquitectura en capas
Múltiples tipos de gráficos se pueden combinar en capas para crear gráficos personalizados combinados. También puede utilizar capas para añadir símbolos para resaltar los puntos específicos, añadir marcadores, umbrales, zonas, límites de error.

Interactivo
Es compatible con los cursores de pista, información sobre herramientas, opción múltiple (drill-down), así como zoom y desplazamiento. Su herramienta visor permite a los usuarios navegar por los datos que pueden variar fácilmente. Soporta ratón táctil y eventos manipuladores para la personalización de las gráficas

8. Implementación del Front End

Estructura de las páginas

Figura 4.8.1 Estructura de las páginas

Nombre de la aplicación
El nombre de la aplicación aparece en la pestaña y cuando se ingresa a otra función aparece el nombre también.
Encabezados
En todas las pantantallas de la aplicación aparece el logo de la compañía y la imagen diseñada para ello en movimiento.
Títulos
Muestra estatus de las operaciones y o lugar donde se encuentra.
Menú
Muestra los minués disponibles para cada tipo de acceso.
Cuerpo
Se presentan los datos y resultado de las consultas.
Pie
Hace referencia a la empresa y la comunicación con el administrador.

Mapa de opciones

Acceso
La aplicación presenta una pantalla de acceso donde se pide el usuario y la clave de acceso, sin ningún menú activado.
Entrada
Una vez con el acceso, las opciones del menú se activan y se dividen en los siguientes opciones:

Principal

Página Principal

- Estaciones de trabajo
- Salida
- Presentación

ABC

- Alta de Ordenes
- Consulta de Ordenes
- Consulta de Catálogos

Usuarios

- Solicitud de Clave
- Solicitud de Modificación

Preferencias

- Configuración de Campos
- Configuración de Flujo
- Alta de Catálogos
- Alta de Formatos

Documentos

- Formatos de Formularios
- Formatos de Contratos
- Productos y precios
- Cotización

Reportes

- Documentos
- Impresión de Flujo

Ayuda

Las pantallas se presentan a continuación:

Acceso

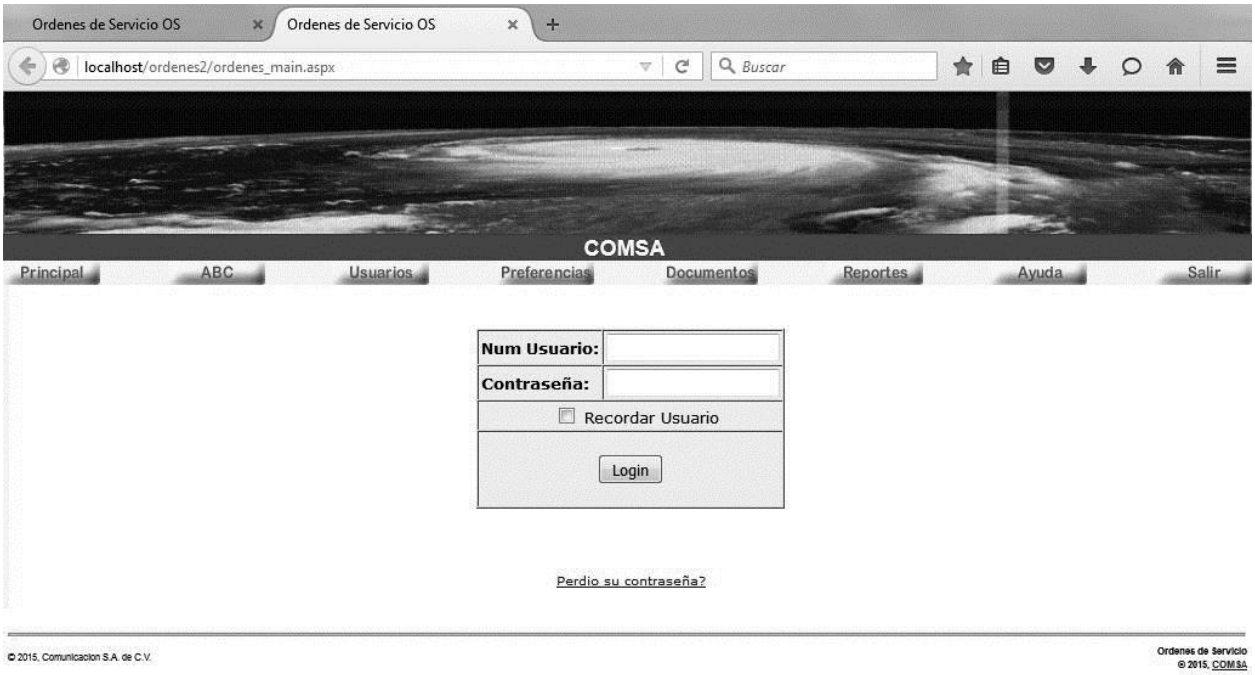


Figura 4.8.2 Acceso

Entrada

Figura 4.8.3 Página Principal

Menú Principal

Figura 4.8.4 Menú Principal

Menú ABC

Figura 4.8.5 Menú ABC

Menú de usuarios

Figura 4.8.6 Menú Usuarios

Menú de Preferencias

Figura 4.8.7 Menú Preferencias

Menú de Documentos

Figura 4.8.8 Menú Documentos

Menú de Reportes

Figura 4.8.9 Menú Reportes

Las imágenes de cada opción en el menú.

Principal

Página Principal
Estaciones de trabajo
Salida
Presentación

Página Principal

Regresa a la pantalla principal desde cualquier menú.

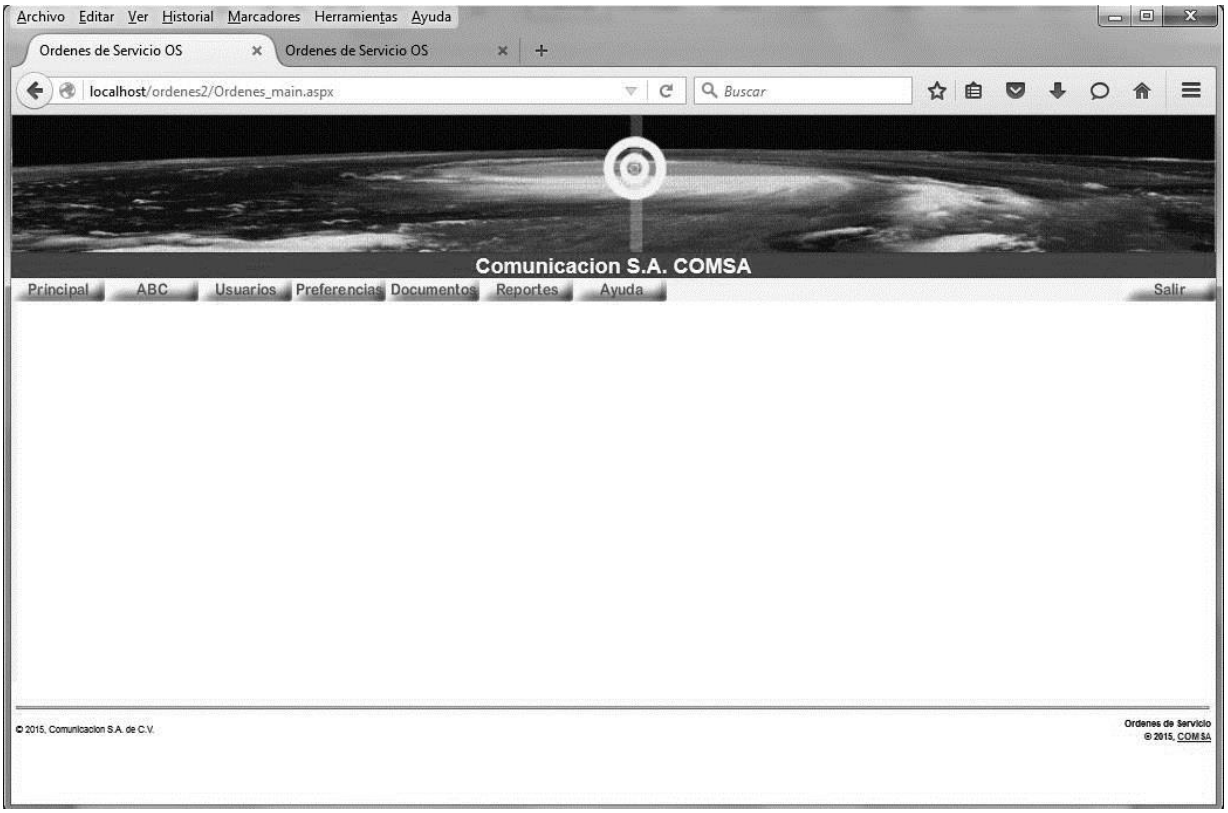


Figura 4.8.10 Página Principal

Estaciones de Trabajo

Muestra las opciones del menú de acuerdo al tipo de usuario.

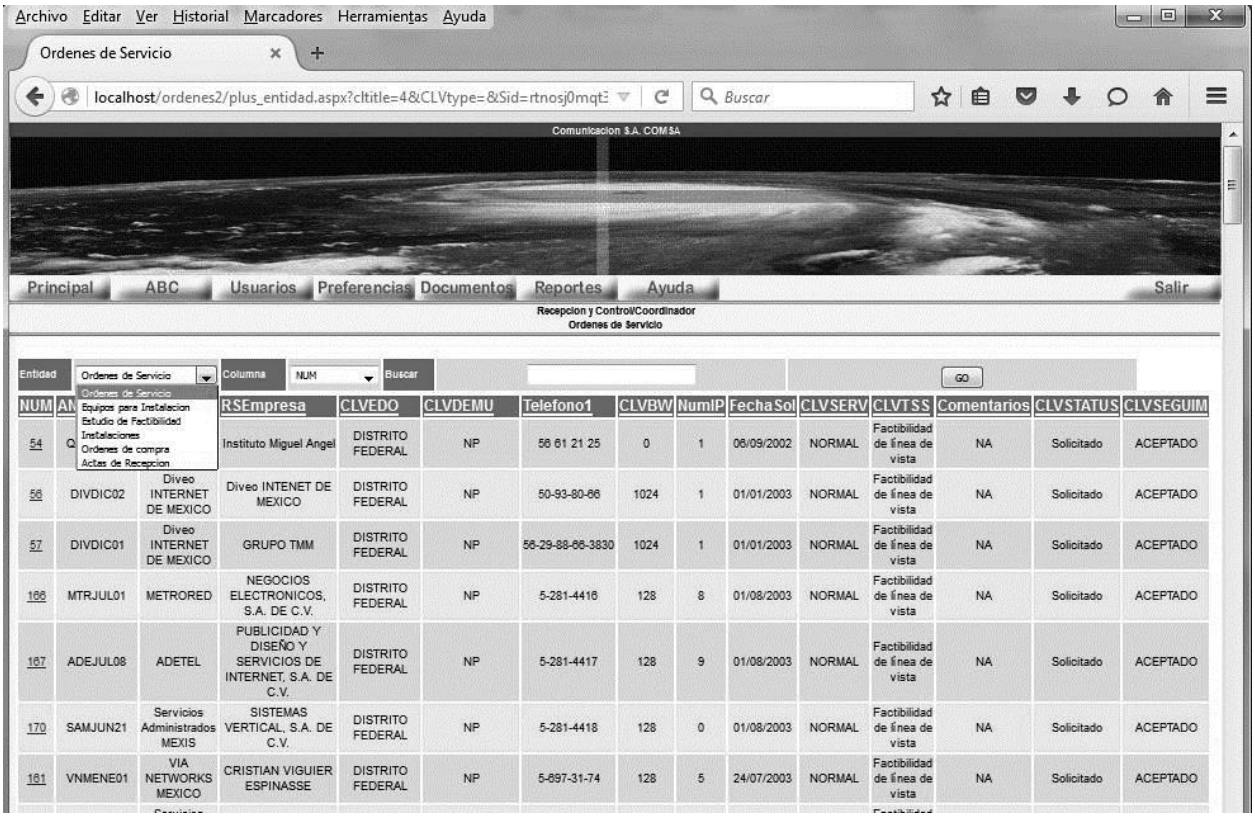


Figura 4.8.11 Menú Estaciones de Trabajo

Presentación

Muestra una presentación de la compañía o canal que ingrese la aplicación.

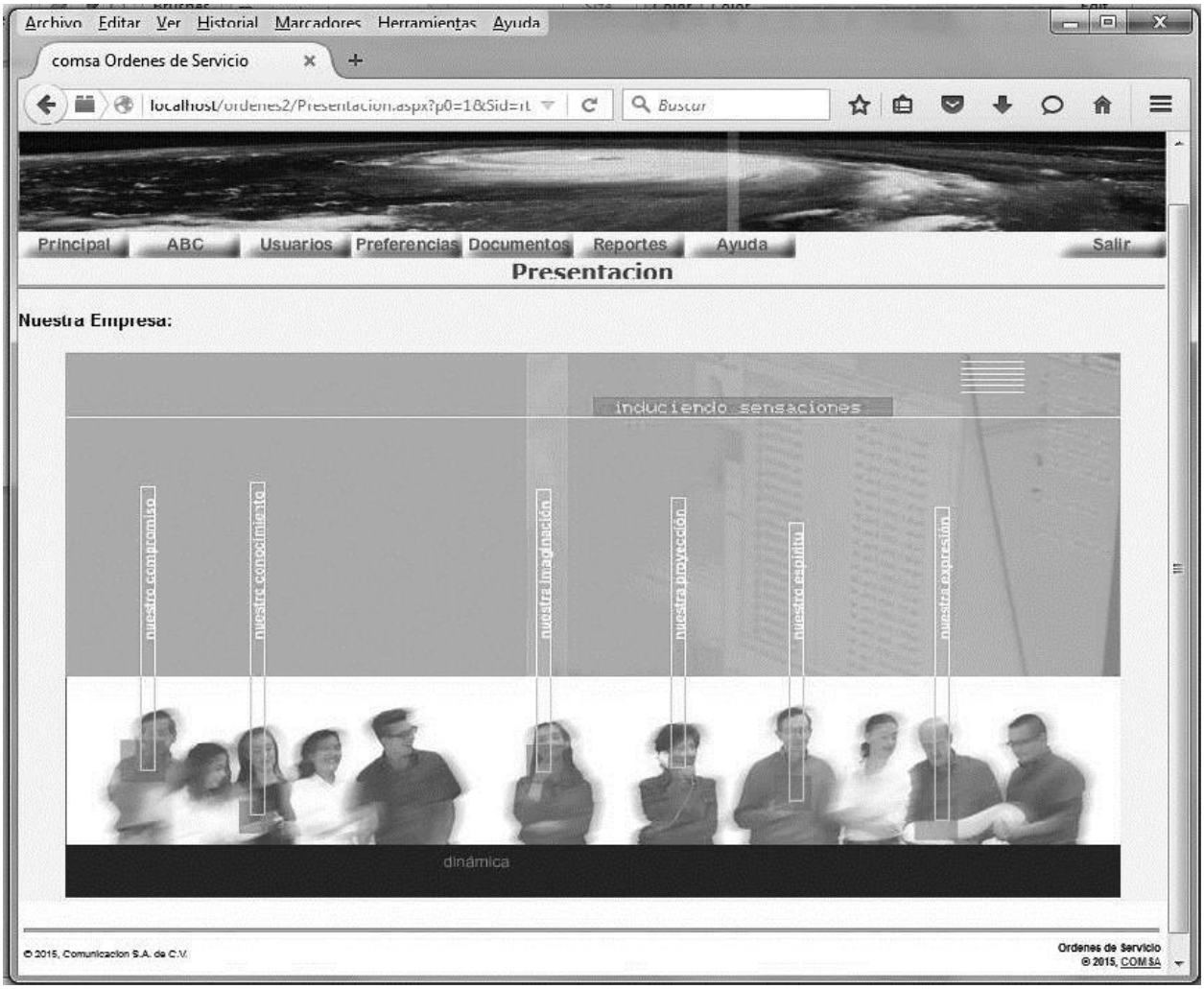


Figura 4.8.12 Menú Presentación

ABC

Alta de Ordenes
Consulta de Ordenes
Consulta de Catálogos

Menú de altas y cambios.

Alta de Ordenes

Da de alta o se modifica una OS si se tiene privilegios para ello.

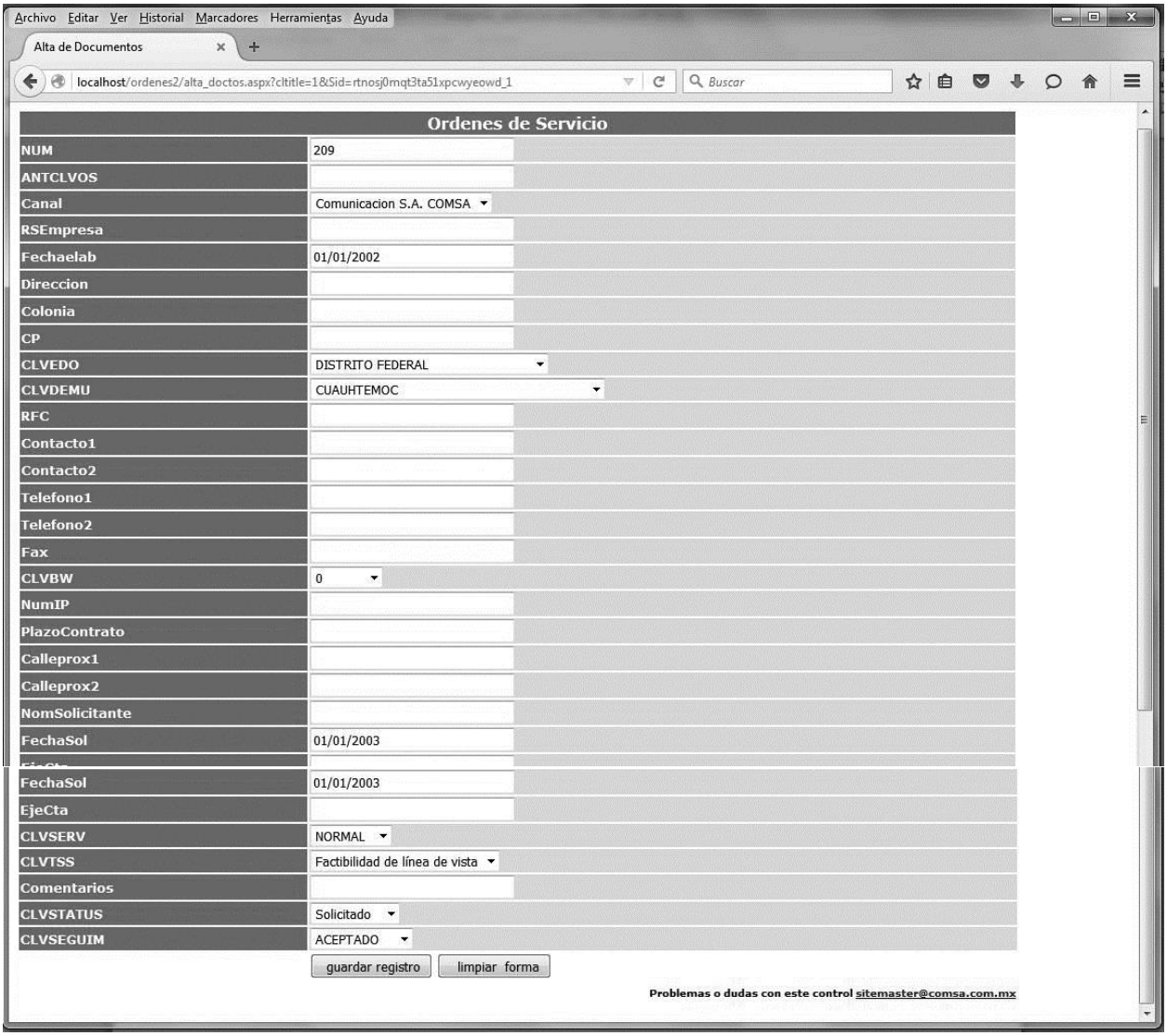
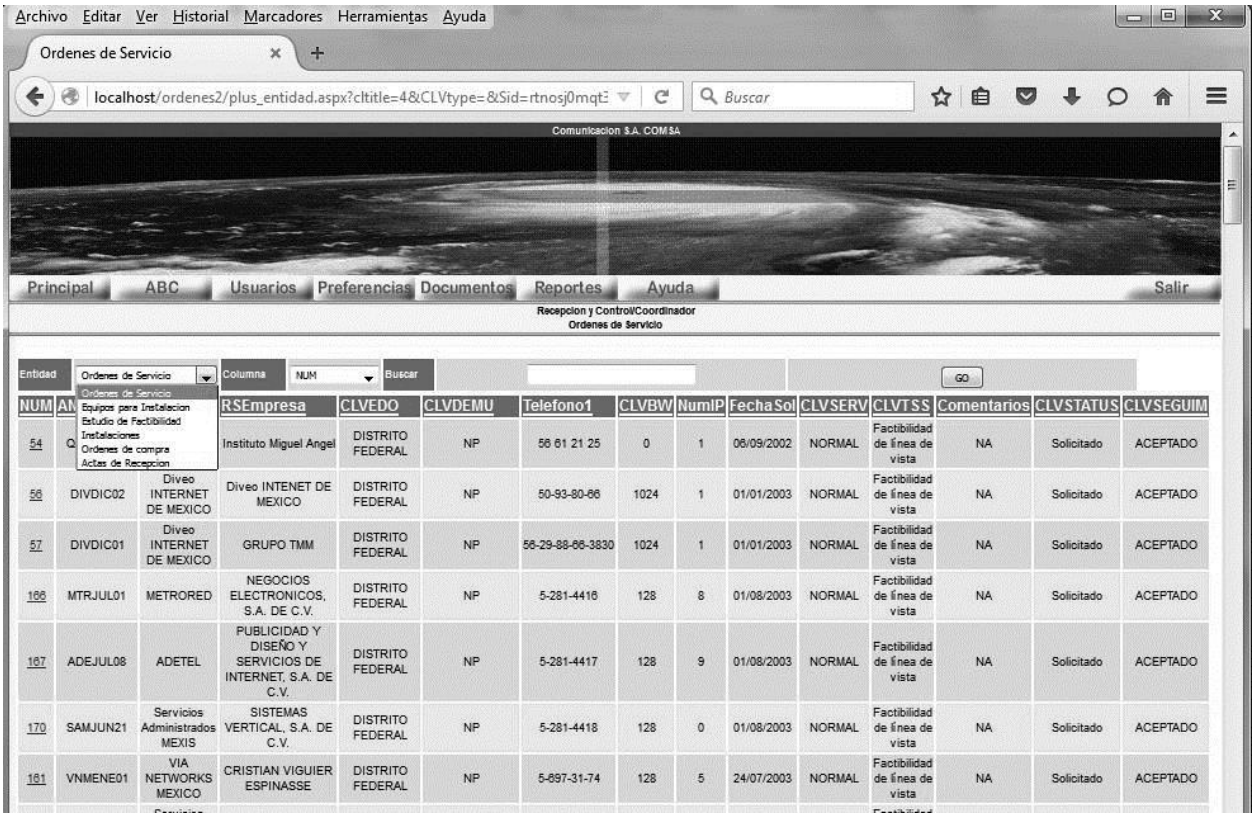


Figura 4.8.13 Menú ABC Alta Documentos

Consulta de Ordenes

Se pueden consultar los diferentes documentos que genera una orden de acuerdo al nivel del usuario.



NUM	AN		RSEmpresa	CLVEDO	CLVDEMU	Telefono1	CLVBW	NumIP	FechaSol	CLVSERV	CLVTSS	Comentarios	CLVSTATUS	CLVSEGUIM
54			Instituto Miguel Angel	DISTRITO FEDERAL	NP	56 61 21 25	0	1	06/09/2002	NORMAL	Factibilidad de linea de vista	NA	Solicitado	ACEPTADO
56	DIVDIC02	Diveo INTERNET DE MEXICO	Diveo INTENET DE MEXICO	DISTRITO FEDERAL	NP	50-93-80-66	1024	1	01/01/2003	NORMAL	Factibilidad de linea de vista	NA	Solicitado	ACEPTADO
57	DIVDIC01	Diveo INTERNET DE MEXICO	GRUPO TMM	DISTRITO FEDERAL	NP	56-29-88-66-3830	1024	1	01/01/2003	NORMAL	Factibilidad de linea de vista	NA	Solicitado	ACEPTADO
166	MTRJUL01	METRORED	NEGOCIOS ELECTRONICOS, S.A. DE C.V.	DISTRITO FEDERAL	NP	5-281-4416	128	8	01/08/2003	NORMAL	Factibilidad de linea de vista	NA	Solicitado	ACEPTADO
167	ADEJUL08	ADETEL	PUBLICIDAD Y DISEÑO Y SERVICIOS DE INTERNET, S.A. DE C.V.	DISTRITO FEDERAL	NP	5-281-4417	128	9	01/08/2003	NORMAL	Factibilidad de linea de vista	NA	Solicitado	ACEPTADO
170	SAMJUN21	Servicios Administrados MEXIS	SISTEMAS VERTICAL, S.A. DE C.V.	DISTRITO FEDERAL	NP	5-281-4418	128	0	01/08/2003	NORMAL	Factibilidad de linea de vista	NA	Solicitado	ACEPTADO
161	VNMENE01	VIA NETWORKS MEXICO	CRISTIAN VIGUIER ESPINASSE	DISTRITO FEDERAL	NP	5-697-31-74	128	5	24/07/2003	NORMAL	Factibilidad de linea de vista	NA	Solicitado	ACEPTADO

Figura 4.8.14 Consulta de Documentos

Consulta de Catálogos

Se pueden consultar todos los catálogos del sistema.

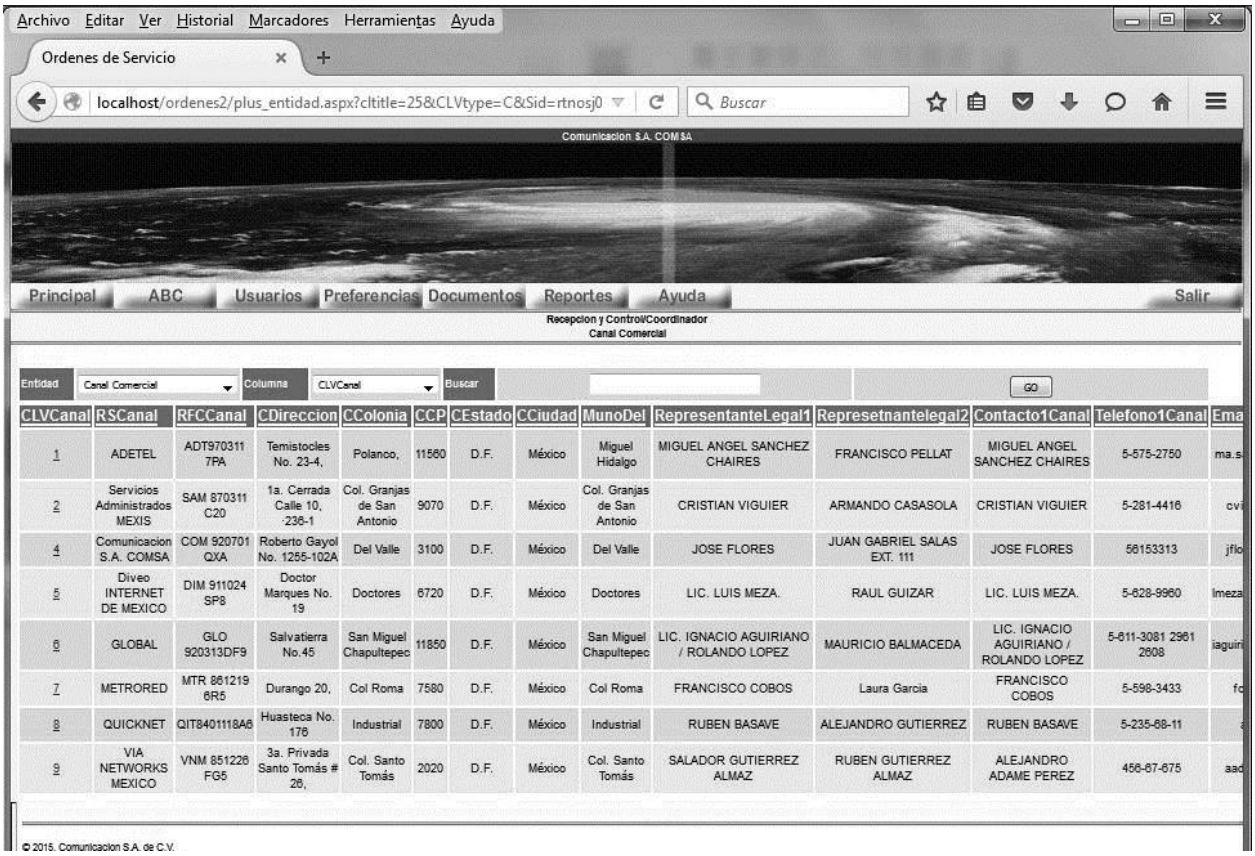


Figura 4.8.15 Canal comercial

Lista de los catálogos

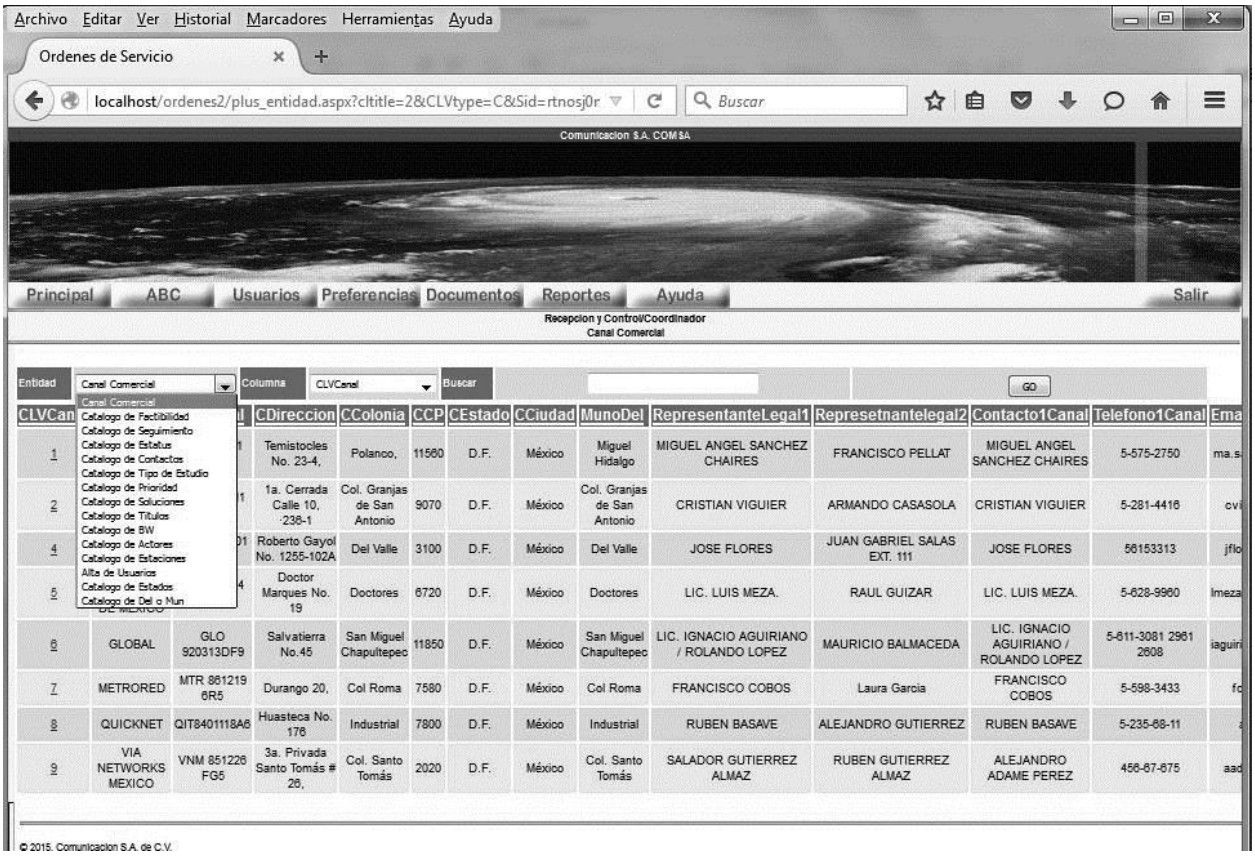


Figura 4.8.16 Menú Catálogos

Preferencias

Configuración de Campos
Configuración de Flujo
Alta de Catálogos
Alta de Formatos

Menú de configuración de la aplicación

Configuración de Campos
Se configuran los campos que aparecerán en las diferentes pantallas (el orden, el nombre y si serán visibles,)

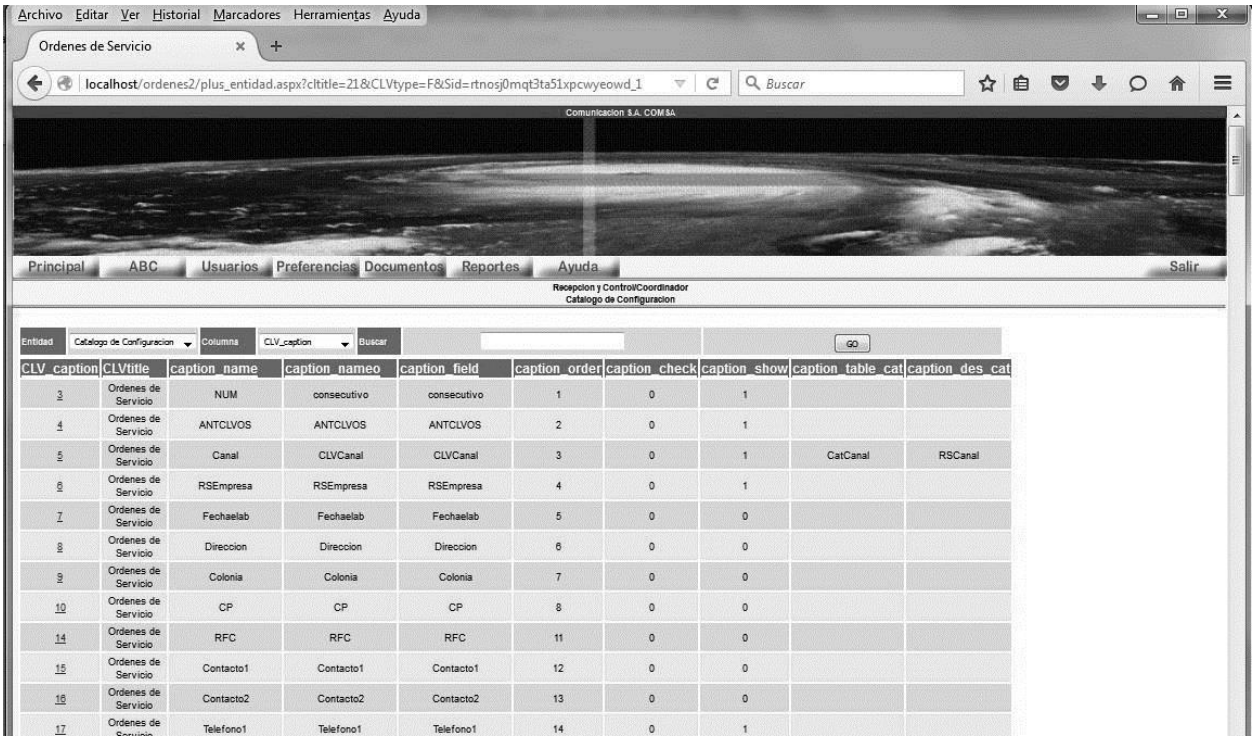


Figura 4.8.17 Configuración de campos

Alta de Catálogos
Se dan de alta o se modifican los datos de los diferentes catálogos

Delegación

Figura 4.8.18 Preferencias Modificación Delegación

Catálogo de factibilidad.

Figura 4.8.19 Preferencias Alta de Catálogos

Configuración de Flujo
En el se da de alta los pasos que deben seguir los documentos.

CLVFLUJO	Actor	Estacion	Documento	STATUS	OrdStatus	GenDoc
1	ContactoC	Canales	Ordenes de Servicio	Solicitado	Solicitado	No docto
80	ContactoC	Canales	Actas de Recepcion	Solicitado	Aceptada	No docto
5	ContactoC	Canales	Ordenes de compra	Solicitado	Rechazado	No docto
79	ContactoC	Canales	Actas de Recepcion	Solicitado	Rechazado	No docto
4	ContactoC	Canales	Ordenes de compra	Solicitado	Aceptada	No docto
2	ContactoC	Canales	Ordenes de Servicio	Rechazado	Solicitado	No docto
6	ContactoC	Canales	Ordenes de compra	Rechazado	Rechazado	No docto
3	ContactoC	Canales	Ordenes de Servicio	Aceptada	Aceptada	No docto
7	ContactoC	Canales	Ordenes de compra	Aceptada	Aceptada	No docto
81	ContactoC	Canales	Actas de Recepcion	Aceptada	Aceptada	No docto
12	Coordinador	Recepcion y Control	Ordenes de compra	Solicitado	Solicitado	No docto
75	Coordinador	Recepcion y Control	Actas de Recepcion	Solicitado	Solicitado	No docto
44	Coordinador	Recepcion y Control	Estudio de Factibilidad	Solicitado	Solicitado	No docto

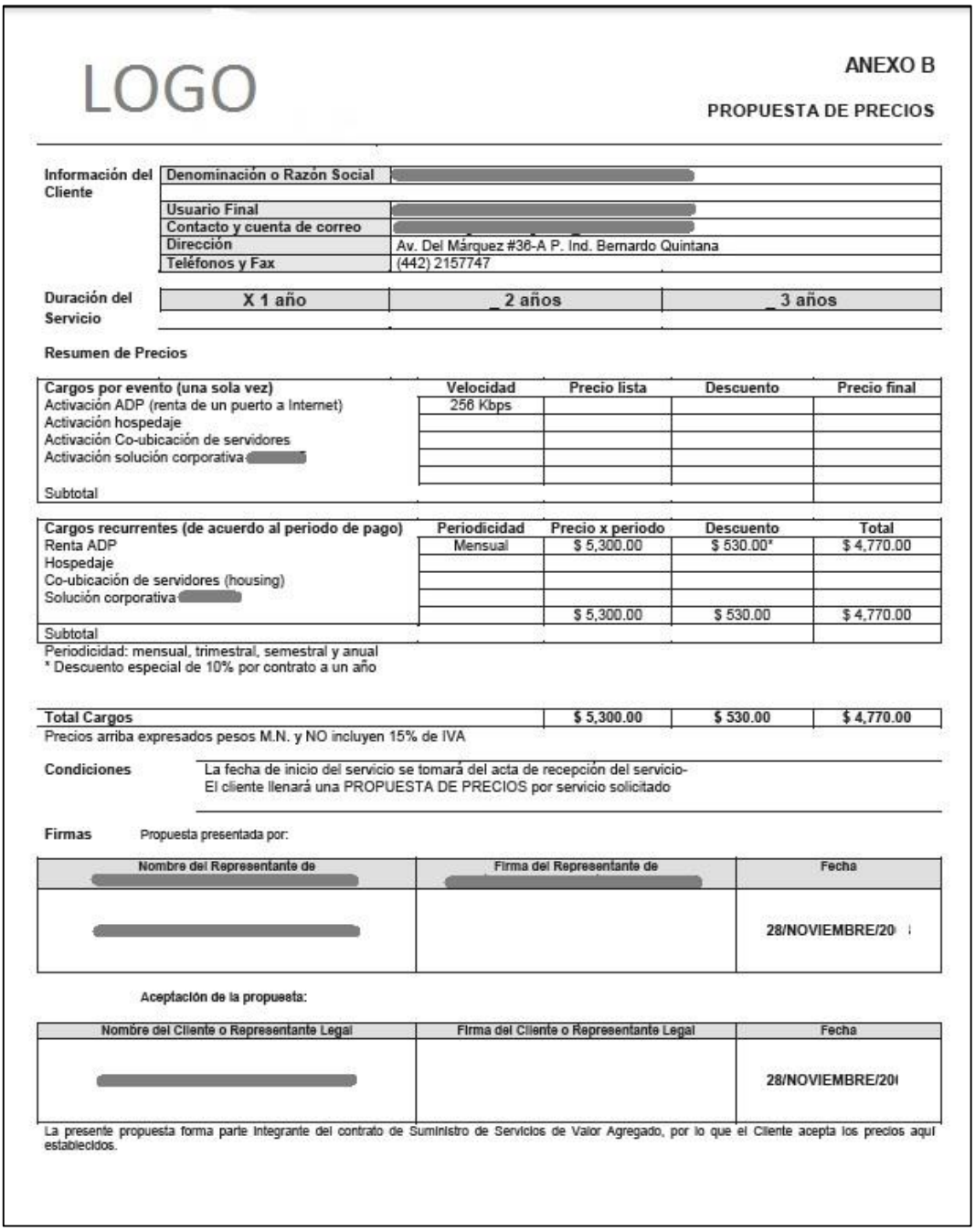
Figura 4.8.20 Preferencias Configuración Flujo

Documentos

Formatos de Formularios
Formatos de Contratos
Productos y precios
Cotización

Menú de formatos de documentos digitales.

Formatos de Formularios



LOGO

ANEXO B

PROPUESTA DE PRECIOS

Información del Cliente	**Denominación o Razón Social**	[redacted]
	Usuario Final	[redacted]
	Contacto y cuenta de correo	[redacted]
	Dirección	Av. Del Márquez #36-A P. Ind. Bernardo Quintana
	Teléfonos y Fax	(442) 2157747

Duración del Servicio	**X 1 año**	**_ 2 años**	**_ 3 años**

Resumen de Precios

Cargos por evento (una sola vez)	**Velocidad**	**Precio lista**	**Descuento**	**Precio final**
Activación ADP (renta de un puerto a Internet)	256 Kbps			
Activación hospedaje				
Activación Co-ubicación de servidores				
Activación solución corporativa [redacted]				
Subtotal				

Cargos recurrentes (de acuerdo al periodo de pago)	**Periodicidad**	**Precio x periodo**	**Descuento**	**Total**
Renta ADP	Mensual	$ 5,300.00	$ 530.00*	$ 4,770.00
Hospedaje				
Co-ubicación de servidores (housing)				
Solución corporativa [redacted]				
		$ 5,300.00	$ 530.00	$ 4,770.00
Subtotal				

Periodicidad: mensual, trimestral, semestral y anual
* Descuento especial de 10% por contrato a un año

Total Cargos	**$ 5,300.00**	**$ 530.00**	**$ 4,770.00**

Precios arriba expresados pesos M.N. y NO incluyen 15% de IVA

Condiciones La fecha de inicio del servicio se tomará del acta de recepción del servicio-
El cliente llenará una PROPUESTA DE PRECIOS por servicio solicitado

Firmas Propuesta presentada por:

Nombre del Representante de [redacted]	**Firma del Representante de** [redacted]	**Fecha**
[redacted]		28/NOVIEMBRE/20

Aceptación de la propuesta:

Nombre del Cliente o Representante Legal	**Firma del Cliente o Representante Legal**	**Fecha**
[redacted]		28/NOVIEMBRE/20

La presente propuesta forma parte integrante del contrato de Suministro de Servicios de Valor Agregado, por lo que el Cliente acepta los precios aquí establecidos.

Figura 4.8.21 Documentos Formularios

Reportes

Documentos
Impresión de Flujo

Menú donde se seleccionan las diferentes opciones de reportes.

Documentos
Se selecciona el documento del que se obtendrán los reportes.

Figura 4.8.22 Menú Reportes

Selección de parámetros

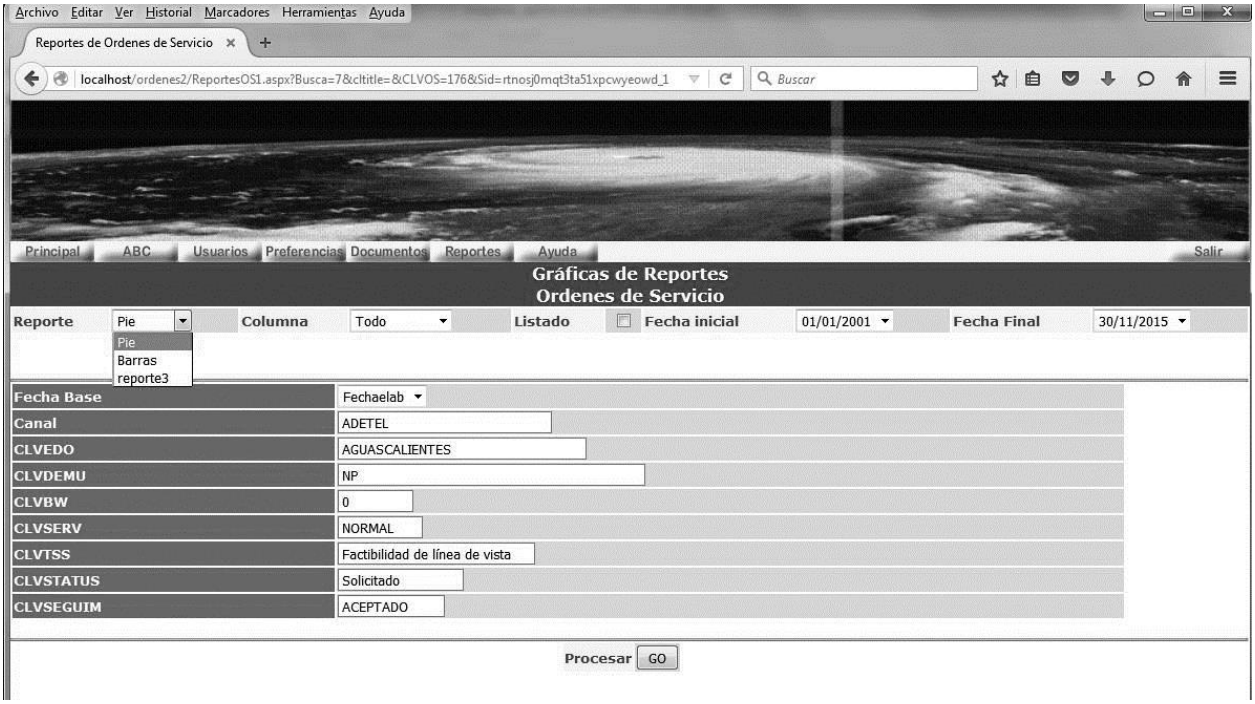


Figura 4.8.23 Reportes parámetros

Resultado de una consulta.

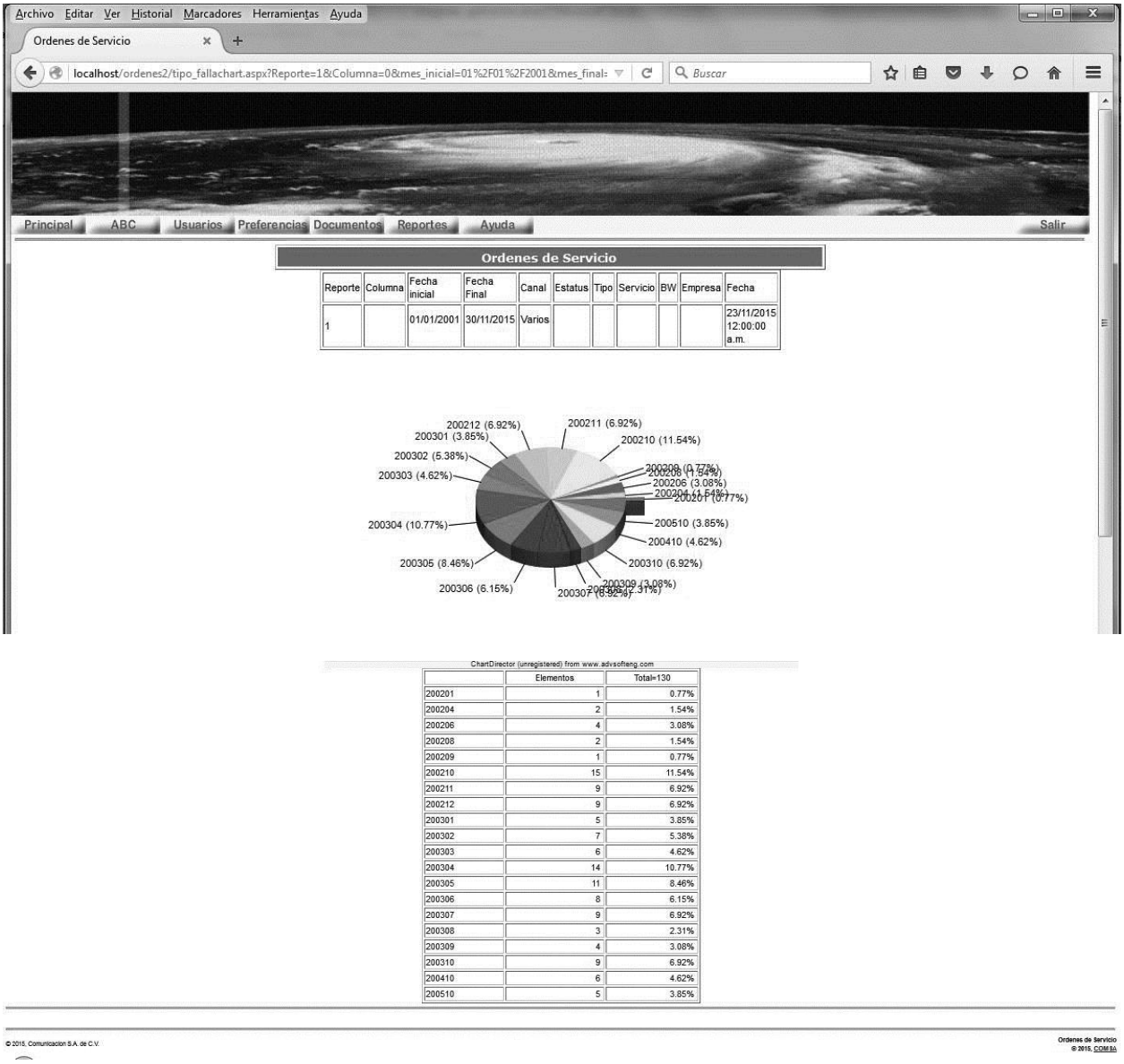


Figura 4.8.24 Reporte típico OS PIE

Otra opción de parámetros

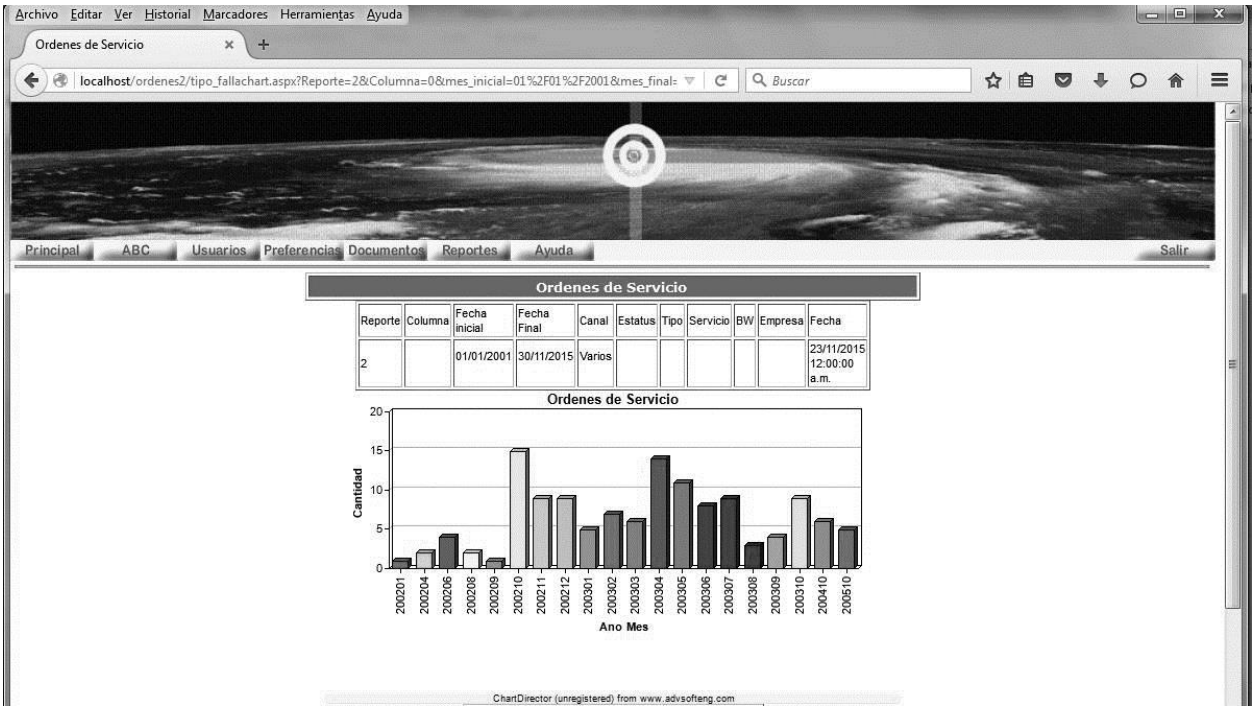
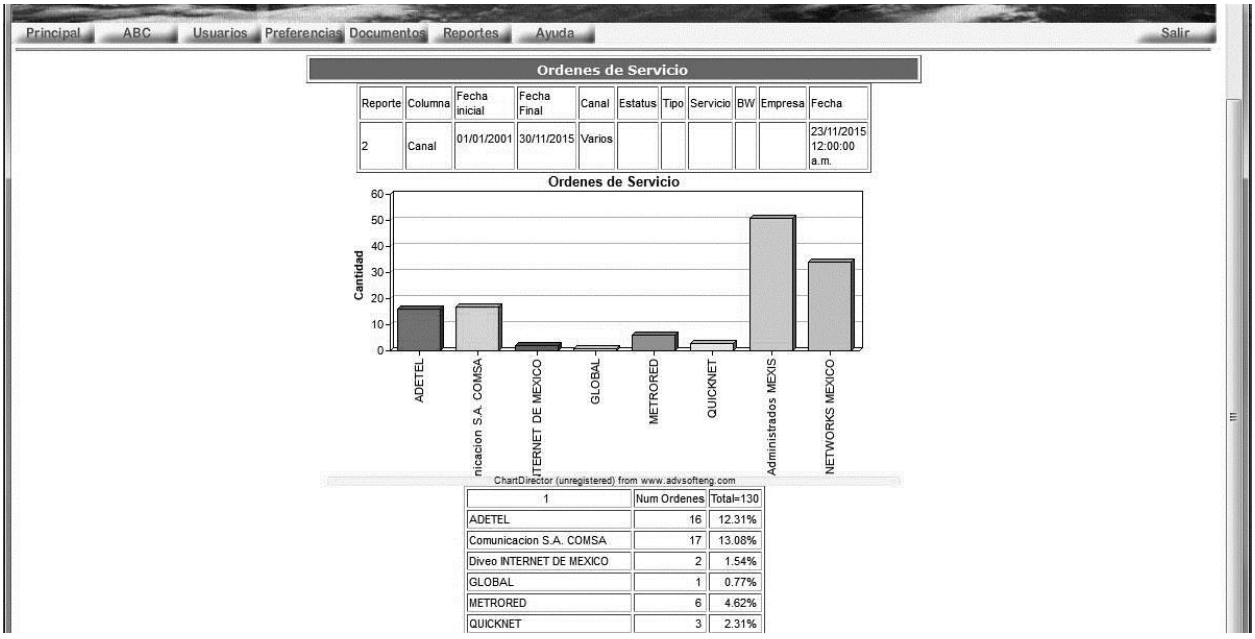
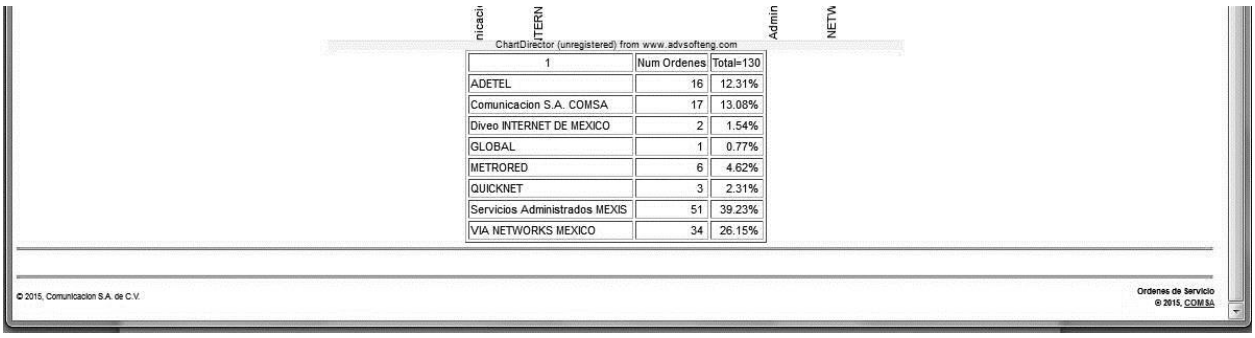


Figura 4.8.25 Reporte típico OS Barras

Información de los datos del reporte

\





1	Num Ordenes	Total=130
ADETEL	16	12.31%
Comunicacion S.A. COMSA	17	13.08%
Diveo INTERNET DE MEXICO	2	1.54%
GLOBAL	1	0.77%
METRORED	6	4.62%
QUICKNET	3	2.31%
Servicios Administrados MEXIS	51	39.23%
VIA NETWORKS MEXICO	34	26.15%

Figura 4.8.26 Reporte típico OS Barras datos

Ayuda

Se muestra un archivo de conceptos básicos utilizados en la aplicación.

Documento de ayuda.

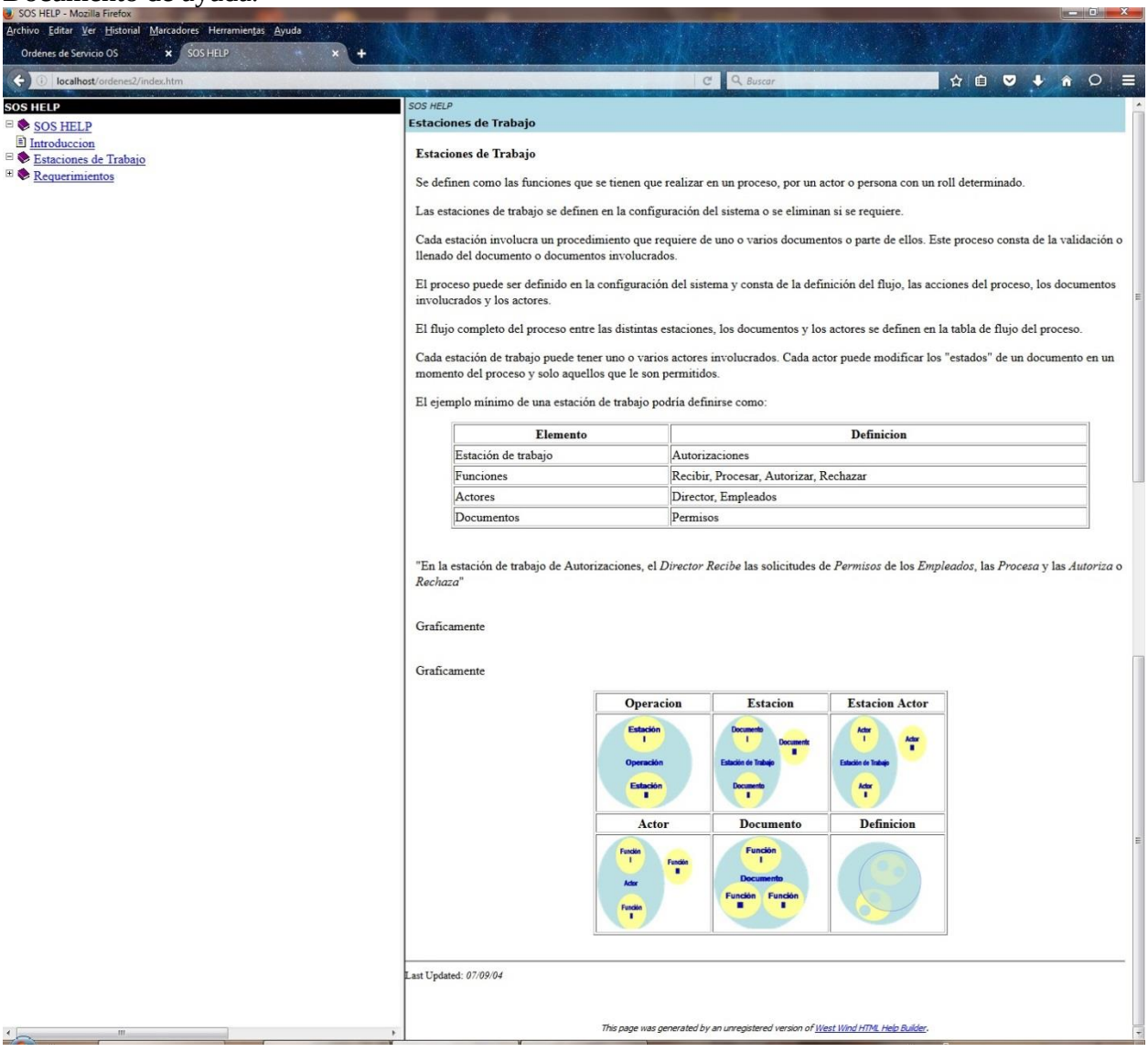


Figura 4.8.27 Menú ayuda

9. Diseño de Reportes

Todos los reportes de la base de datos se diseñaron siguiendo un estándar de presentación. El sistema plantea la presentación grafica en pantalla de los datos almacenados en cualquier momento y no la impresión de ningún reporte

Aunque existe una relación entre cada uno de los documentos del sistema, ya que todos se generan a partir de la orden de servicio, los documentos se presentan por separado para su análisis. Esto debido entre otras cosas a que por ejemplo existen muchas más ordenes de servicio realizadas y completadas en su totalidad, que servicios instalados y esto porque finalmente el equipo no fue instalado o las condiciones contractuales no llegaron a buen término

Los reportes en pantalla presentan básicamente cuatro secciones:

- Parámetros de Selección y filtros
- Gráficas
- Tabla resumen.
- Listado

Parámetros de Selección y filtros

En los parámetros se presentan en una tabla donde se puede apreciar básicamente:

- Documento del que se obtuvo la información
- Periodo del reporte
- Columna base de reporte
- Filtros

Gráfica

Se presentan dos tipos de graficas de pie y barras. El sistema fue diseñado para agregar mas tipos de gráficos en el menú.

Tabla resumen.

En ella se muestra las agrupaciones y el total o cuenta por agrupación y porcentaje de que representa dicha agrupación

Listado

Se proporciona un listado de los registros utilizados para la obtención del reporte. Esta sección corresponde a una petición de usuario que deseaba contar con los datos para la obtención de otros reportes, aunque la base puede ser explotada mediante productos de Microsoft mediante conexiones OLEDB.
Esta sección esta deshabilitada por default y solo se realiza mediante confirmación para todos los reportes.
En las figuras 4.8.22 a la 4.8.26 de la sección del Front End se pueden ver ejemplos de algunos reportes y las secciones.

5. *PRUEBAS*

Las pruebas de software son las actividades empíricas y técnicas cuyo objetivo es proporcionar información objetiva e independiente sobre la calidad del producto a la parte. Las pruebas son parte del proceso de control de calidad.
Las pruebas son el conjunto de actividades que se desarrollan en cualquier momento del desarrollo de una aplicación para asegurar su funcionamiento y calidad. Como existen distintos modelos de desarrollo de software, existen así distintos modelos de pruebas

Las pruebas nos permiten:

- Comprobar la funcionalidad de los requerimientos de los clientes.
- Detectar y registrar defectos.
- Verificar la calidad del producto entregado.

Dicho en otras palabras:

Las pruebas son los procesos orientados a demostrar que el software no tiene errores y que el sistema realiza las funciones para las que fue creado. Sin embargo el proceso de la realización de pruebas debe estar orientado a detectar defectos y fallas. Una prueba exitosa es aquella que detecta errores

Para aclarar cuál es el resultado de una prueba debemos aclarar que:

- **Error** Es una equivocación de una persona al desarrollar alguna actividad del desarrollo de software
- **Defecto** Se produce cuando una persona comete un error
- **Falla** Es un desvío respecto del comportamiento esperado del sistema.

La realización de pruebas es un proceso laborioso y costoso en términos de tiempo, pero es menos costoso que no realizarlo

Las pruebas deben realizarse desde las primeras etapas del desarrollo ya que conforme se avanza, el costo de los errores es mayor. Figura 5.1.1

El ambiente ideal de las pruebas es aquel que es independiente del desarrollo del software, de esta manera se logra objetividad en las pruebas.

Las actividades, técnicas, documentación, enfoques y demás elementos que condicionarán las pruebas a realizar, deben ser seleccionadas y utilizadas de la manera más eficiente según el contexto del proyecto. Es de este modo que podemos tratar de establecer una clasificación de las pruebas que pueden realizarse. Figura 5.1.2

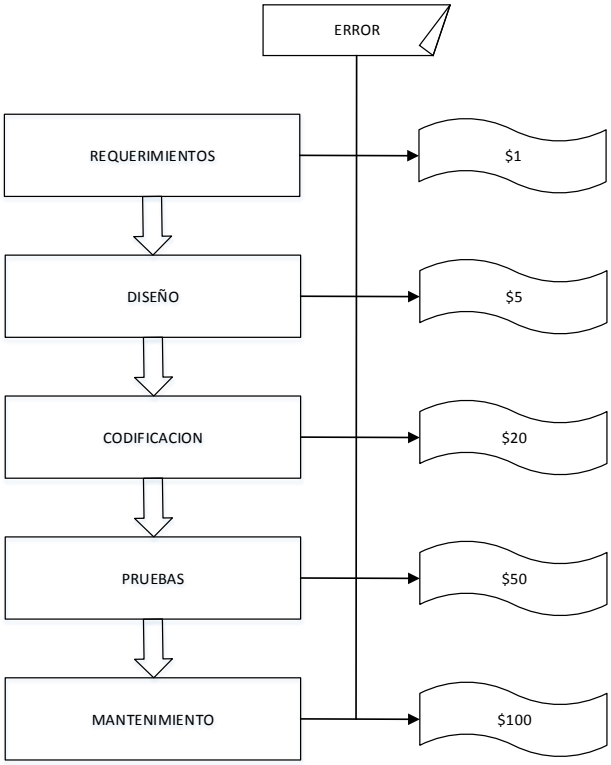


Figura 5.9.1 Costos de Errores

Categorías de Pruebas

Las pruebas pueden clasificarse de acuerdo a varios criterios, dependiendo que objetivo se busca y como se realizará. Sin ser exhaustivo en la tabla de la figura 5.1.2 se describen brevemente algunas pruebas.

Prueba	Descripción
Estáticas	Son el tipo de pruebas que se realizan sin ejecutar el código de la aplicación. La revisión de documentos pueden realizar "pruebas de escritorio" con el objetivo de seguir los flujos de la aplicación.
Dinámicas	Requieren la ejecución de la aplicación. Permiten el uso de técnicas de caja negra y caja blanca con mayor amplitud. Debido a la naturaleza dinámica de la ejecución de pruebas es posible medir con mayor precisión el comportamiento de la aplicación desarrollada.
Compatibilidad	Se comprueba el funcionamiento del software desarrollado en muchas plataformas: sistemas operativos, navegadores, redes, hardware
Regresión	Se evalúa el correcto funcionamiento del software desarrollado frente a evoluciones o cambios funcionales. El propósito de éstas es asegurar que los casos de prueba que ya habían sido probados y fueron exitosos permanezcan así. Se recomienda que este tipo de pruebas sean automatizadas para reducir el tiempo y esfuerzo en su ejecución.
Integración	Posterior a las pruebas modulares de los componentes de un sistema. Se centra principalmente en probar la comunicación entre los componentes de un mismo sistema, comunicación entre sistemas o

	entre hardware y software
Ejecución	
Manuales	Se refiere a que para realizarlas se establecen métodos manuales
Automáticas	Se refiere a que para realizarlas se establecen métodos Automáticos ya sea por programas especiales o rutinas especializadas.
Enfoques	
Caja blanca	Se desarrollan conociendo la estructura interna del sistema y se asegura que las operaciones realizan todas las operaciones internas necesarias.
Caja negra	Verificar la aplicación y sus procesos internos, mediante la interacción con la aplicación vía GUI y analizar la salida (resultados).
Aleatorio	
Niveles	
Unitarias	Se focaliza en ejecutar cada módulo (o unidad mínima a ser probada, ej. = una clase) lo que provee un mejor modo de manejar la integración de las unidades en componentes mayores. Busca asegurar que el código funciona de acuerdo con las especificaciones y que el módulo lógico es válido
Integración	Identificar errores introducidos por la combinación de programas probados unitariamente. Determina cómo la base de datos de prueba será cargada. Verificar que las interfaces entre las entidades externas (usuarios) y las aplicaciones funcionan correctamente. Verificar que las especificaciones de diseño sean alcanzadas. • Determina el enfoque para avanzar desde un nivel de integración de las componentes al siguiente. • Describe cómo verificar que las interfaces entre las componentes de software funcionan correctamente. • Determina cómo la base de datos de prueba será cargada. • Determina el enfoque para avanzar desde un nivel de integración de las componentes al siguiente. • Decide qué acciones tomar cuando se descubren problemas.
Sistema	Las pruebas del sistema deben enfocarse en requisitos que puedan ser tomados directamente de casos de uso y reglas y funciones de negocios. El objetivo de estas pruebas es verificar el ingreso, procesamiento y recuperación apropiado de datos, y la implementación apropiada de las reglas de negocios. Este tipo de pruebas se basan en técnicas de caja negra, esto es, verificar el sistema (y sus procesos internos), la interacción con las aplicaciones que lo usan vía GUI y analizar las salidas o resultados. En esta prueba se determina qué pruebas de Sistema (usabilidad, volumen, desempeño, etc.) asegurarán que la aplicación alcanzará sus objetivos de negocio. La prueba de Sistema incluye:

	• Prueba funcionalidad • Prueba de Usabilidad • Prueba de Performance • Prueba de Documentación y Procedimientos • Prueba de Seguridad y Controles • Prueba de Volumen • Prueba de Esfuerzo (Stress) • Prueba de recuperación • Prueba de múltiples sitios Para sistemas web se recomienda: • Humo. • Usabilidad • Performance • Funcionalidad La prueba de sistema es compleja porque intenta validar un número grande de características al mismo tiempo, a diferencia de otras pruebas que sólo se centran en uno o dos aspectos al mismo tiempo
Funcionales	
Funcionales	Las pruebas Funcionales deben enfocarse en los requisitos funcionales, las pruebas pueden estar basadas directamente en los Casos de Uso (o funciones de negocio), y las reglas del negocio. Las metas de estas pruebas son: • Verificar la apropiada aceptación de datos, • Verificar el procesamiento y recuperación y la implementación adecuada de las reglas del negocio. Este tipo de pruebas están basadas en técnicas de caja negra. Se ejecuta cada caso de uso, flujo de caso de uso, o función, usando datos válidos e inválidos, para verificar: • Que los resultados esperados ocurran cuando se usen datos válidos. • Que sean desplegados los mensajes apropiados de error y precaución cuando se usan datos inválidos. • Que se aplique apropiadamente cada regla de negocio
De humo	Los objetivos son los siguientes: • Detectar los errores en versiones tempranas y de manera fácil • Probar el sistema constantemente • Garantizar poco esfuerzo en la integración final del sistema Toma éste nombre debido a que su objetivo es probar el sistema constantemente buscando que saque “humo” o falle. En algunos proyectos este tipo de prueba va junto con las pruebas funcionales. Permite detectar problemas que por lo regular no son detectados en

	las pruebas normales. Algunas veces, si las Pruebas ocurren tarde en el ciclo de desarrollo está será una forma de garantizar el buen desarrollo. • Las pruebas de humo ***NO SON*** exhaustivas, pero van de extremo a extremo de la aplicación. • Asegurar los resultados de las pruebas unitarias • Se reducen los riesgos y la baja calidad.
Pruebas de Regresión	Determinar si los cambios recientes en una parte de la aplicación tienen efecto adverso en otras partes. En esta prueba se vuelve a probar el sistema a la luz de los cambios realizados durante el debugging, mantenimiento o desarrollo de la nueva versión del sistema buscando efectos adversos en otras partes
Pruebas de Aceptación	La prueba de aceptación es ejecutada antes de que la aplicación sea instalada dentro de un ambiente de producción. La prueba de aceptación es generalmente desarrollada y ejecutada por el cliente o un especialista de la aplicación y es conducida a determinar como el sistema satisface sus criterios de aceptación validando los requisitos que han sido levantados para el desarrollo, incluyendo la documentación y procesos de negocio. Basado en esta prueba el cliente determina si acepta o rechaza el sistema Estas pruebas están destinadas a probar que el producto está listo para el uso operativo. Suelen ser un subconjunto de las Pruebas de Sistema. Sirve para que el usuario pueda validar si el producto final se ajusta a los requisitos fijados, es decir, si el producto está listo para ser implantado para el uso operativo en el entorno del usuario. Los casos prueba de aceptación han de ser planificados, organizados y formalizados de manera que se determine el cumplimiento de los requisitos del sistema. Para la realización de estas pruebas se necesita disponer de los siguientes documentos: • Especificación de requisitos del sistema. • Manual de usuario. • Manual de administrador
Alpha testing	La verificación involucra la ejecución de partes o todo del sistema en ambientes simulados. La retroalimentación de esta fase produce cambios en el software para resolver los errores y fallas que se descubren Las pruebas de sistema se realizan: • En el lugar en donde fue desarrollado el sistema. • En un ambiente controlado, done el desarrollador está presente. Se selecciona un grupo de usuarios para la prueba y se les pide trabajen con el sistema.
Beta testing	Realizar la validación del sistema por parte del usuario

	La validación involucra el uso del software en un ambiente real • Se selecciona un grupo de usuarios que ponen a trabajar el sistema. Usan el sistema en sus actividades cotidianas, procesan transacciones y producen salidas normales del sistema. • Las transacciones y personas que usan el sistema son reales y trabajan en su área de trabajo real. • El desarrollador no está presente. • Los usuarios están advertidos que el sistema puede fallar
No funcionales	
Seguridad	Las pruebas de seguridad y control de acceso se centran en dos áreas claves de seguridad: • Seguridad del sistema, incluyendo acceso a datos o Funciones de negocios y • Seguridad del sistema, incluyendo ingresos y accesos remotos al sistema. Las pruebas de seguridad de la aplicación garantizan que: • Los usuarios están restringidos a funciones específicas o su acceso está limitado únicamente a los datos que está autorizado a acceder. • Solamente aquellos usuarios autorizados a acceder al sistema son capaces de ejecutar las funciones del sistema a través de los mecanismos apropiados. Debido a la creciente preocupación de la sociedad por la privacidad de la información, muchos programas tienen objetivos específicos de seguridad. El foco principal es probar la vulnerabilidad del sistema frente a accesos o manipulaciones no autorizadas. Algunas consideraciones de prueba son: • Controles de acceso físico • Acceso a estructuras de datos específicas a través de los programas de aplicación. • Seguridad en sitios remotos • Existencia de datos confidenciales en reportes y pantallas • Controles manuales, incluyendo aquellos para autorización y aprobación, formularios, documentación numerada, transmisión de datos, balances y conversión de datos. • Controles automáticos, incluyendo aquellos para edición de datos, chequeo de máquinas, errores del operador, acceso a datos elementales y archivos, acceso a funciones, auditoría, entre otros.
Usabilidad	Determina cuán bien el usuario podrá usar y entender la aplicación. Identifica las áreas de diseño que hacen al sistema de difícil uso para

	el usuario. La prueba de usabilidad detecta problemas relacionados con la conveniencia y practicidad del sistema desde el punto de vista del usuario Las pruebas intentan verificar que la aplicación no presenta los siguientes problemas de usabilidad típicos: • El sistema es demasiado complejo y difícil de usar. • Es difícil instalar y entender el sistema • La recuperación de errores es pobre y los mensajes de error no tienen significado • La sintaxis de los comandos es difícil de aprender y recordar • El sistema obliga al usuario a recordar formatos y secuencias fijas • Los procedimientos no son simples ni obvios • El sistema no tiene instrucciones de ayuda por computadora y tiene manuales pobres. • Los diagramas, pantallas, reportes y gráficos son de calidad y apariencia pobre • El sistema carece de herramientas de construcción adecuadas y requiere múltiples comandos • La lógica y conveniencia de los botones, switches, displays y mensajes de ayuda deben ser testeados. La prueba de usabilidad puede ser conducida por un grupo separado.
Desempeño	Las pruebas de desempeño miden tiempos de respuesta, índices de procesamiento de transacciones y otros requisitos sensibles al tiempo. El objetivo es verificar y validar los requisitos de desempeño que se han especificado. Las pruebas de desempeño usualmente se ejecutan varias veces, utilizando en cada una, carga diferente en el sistema. La prueba inicial debe ser ejecutada con una carga similar a la esperada en el sistema. Una segunda prueba debe hacerse utilizando una carga máxima. Adicionalmente, las pruebas de desempeño pueden ser utilizadas para perfilar y refinar el desempeño del sistema como una función de condiciones tales como carga o configuraciones de hardware Las principales actividades son: • Comparar el desempeño del sistema actual con los requisitos, • Poner a punto el sistema para mejorar las métricas de desempeño y proyectar la capacidad futura de carga del sistema. Los objetivos de nivel de servicio definidos deben guiar la prueba de performance. Algunas características que afectan el desempeño son:

	• Errores lógicos • Procesamiento ineficiente • Diseño pobre: muchas interfaces, instrucciones y entradas/salidas. • Cuellos de botella en discos, CPU ó canales de entrada/salida • Salidas del sistema • Tiempos de respuesta • Capacidad de almacenamiento • Tasa de entrada/salida de datos • Número de transacciones que pueden ser manejadas simultáneamente
Pruebas de internacionalización y localización	Se desarrollan en sistemas que funcionan en distintos países y regiones. Las pruebas de desarrollan cambiando los parámetros como país y o localización
Escalabilidad	Las pruebas se refieren a la capacidad para manejar uno o mil conceptos y si el sistema puede manejarlos. Ejemplo manejo de un usuario y 1000 usuarios.
Mantenibilidad	Las pruebas se realizan para saber qué tan fácil es mantener al sistema
Instalabilidad	Las pruebas se realizan para saber qué tan fácil es instalar el sistema

Figura 5.9.2 Clasificación de Pruebas

Principios básicos.
Para la realización de pruebas en el desarrollo de sistemas clásicos existen criterios o principios básicos que deben tomarse en cuenta.

- Una parte necesaria de una prueba es la definición de los resultados esperados
- Un programador debe evitar probar su propio desarrollo
- Una organización debe probar sus propios desarrollos
- Los resultados deben revisarse a profundidad.
- Las pruebas deben incluir entradas inválidas e inesperadas así como las esperadas y válidas.
- Revisar un programa para verificar que funciona es solo la mitad del trabajo, la otra mitad es que no haga lo que no se espera
- No tirar las pruebas a la basura a menos que el sistema sea una basura.
- No planear esfuerzo de pruebas asumiendo que no se encontraran errores.
- La probabilidad de encontrar errores en una sección es proporcional a los errores ya encontrados.
- Las pruebas constituyen una tarea creativa e intelectualmente desafiante.

En varios de los métodos de diseño y construcción de aplicaciones más recientes, el criterio de pruebas está integrado en la metodología de diseño y construcción. De ejemplo tenemos para la meta-metodología de Desarrollo guiado por la funcionalidad FDD (**Feature-Driven Development**) el proceso número 5 BBF :

Proceso número 5 Construir por funcionalidad (BBF)

Es una actividad de predefinición para producir una función completamente valuada por el cliente.

Comienza con el paquete de diseño, los desarrolladores de clases implementan los elementos necesarios para que la clase sea soportada por el diseño de la función. El código desarrollado es probado e inspeccionado. El orden es determinado por el programador en jefe. Después de una inspección exitosa el código es mandado a construcción.

Criterio de entrada

El proceso de diseño por funcionalidad está completo. El paquete de diseño ha sido exitosamente inspeccionado.

Secuencia

Actividad	Encargado	Opción
Implementar clases y métodos.	Equipo de funcionalidad.	Requerido
Inspeccionar Código	Equipo de funcionalidad.	Requerido
Prueba unitaria	Equipo de funcionalidad.	Requerido
Autorizado para Construcción	Jefe de programación, Equipo de funcionalidad	Requerido
Verificación		
Inspección de código y Prueba unitaria.	Jefe de programación, Equipo de funcionalidad	Requerido

Figura 5.9.3 Secuencias FDD

Criterio de Salida

El resultado de este proceso es:

- Clases y métodos que han sido exitosamente probados.
- Clases y métodos que han sido aprobados para su construcción.
- Funciones completas y evaluadas por el cliente.

En la siguiente tabla figura 5.1.4 se puede ver un resumen de los niveles de pruebas

Test	Objetivo	Participantes	Ambiente	Método
Unitario	Detectar errores en los datos, lógica, algoritmos	Programadores	Desarrollo	Caja Blanca
Integración	Detectar errores de interfaces y relaciones entre componentes	Programadores	Desarrollo	Caja Blanca, Top Down, Bottom Up
Funcional	Detectar errores en la implementación de requerimientos	Testers y analistas	Desarrollo	Funcional
Sistema	Detecta fallas en el cubrimiento de los requerimientos	Testers y analistas	Desarrollo	Funcional
Aceptación	Detecta fallas en la implementación del sistema	Testers, analistas y Cliente	Productivo	Funcional

Figura 5.9.4 Niveles de Pruebas

Como ya se ha visto el proceso de pruebas es muy laborioso y costoso en tiempo por lo que el uso de programas o automatización de pruebas para acortar estos es fundamental, sobre todo en sistemas grandes.

Herramientas para pruebas

Algunas de las razones para automatizar la realización de pruebas son:

- El ciclo manual es muy largo.
- El proceso manual es propenso a errores o desviaciones.
- El ciclo se acorta y se liberan recursos.
- Genera un ambiente de confianza.
- Se puede obtener retroalimentación temprana y con alta frecuencia.
- Genera conocimiento del desarrollo del sistema
- Genera documentación del código consistente.
- Genera menores costos.

Por supuesto no todo puede automatizarse pero donde se puede realizar, el beneficio es representativo. Las pruebas que no se deben automatizar son por ejemplo las de usabilidad, exploratorias, y las de fácil ejecución manual.

En el anexo de herramientas para pruebas se da una lista de estos sistemas automáticos.

7. Pruebas Establecidas y Realizadas

Para el sistema de órdenes se planearon pocas pruebas, debido entre otras cosas a la falta de tiempo, personal y recursos para realizarlas o autorizarlas.

Las pruebas que se planearon son:

- Prueba de Sistema,
- Pruebas de Seguridad
- Pruebas Modulares
- Pruebas Compatibilidad
- Prueba de Volumen
- Pruebas de Integridad
- Pruebas de Calidad de Diseño.

Los elementos en cada una de estas pruebas son:

Prueba de Sistema,
Verificación del ciclo de negocio.

Creación y consulta de Órdenes de servicio
Creación y consulta de Estudios de factibilidad
Creación y consulta de Estudios AIR
Creación y consulta de Orden de Compra
Creación y consulta de Instalaciones
Creación y consulta de Equipos para Instalaciones
Creación y consulta de Actas de Recepción

Se verificó la creación de los registros en la base de datos, en el ciclo que corresponde y que estos se pudieran observar por las pantallas y usuarios que se requirió.

Se crearon ejemplos base de las diferentes opciones y se corroboró su funcionamiento.

Verificación de Requerimientos

Se revisó el documento de requerimientos y se aceptaron o rechazaron cada uno de los requerimientos.

Pruebas de Seguridad

Se estableció la prueba de claves de entrada
Se estableció la verificaron las claves de acceso a la base de datos
Se estableció la verificación de los roles de usuario en la aplicación.
Se estableció la realizaron algunas pruebas de seguridad básica de aplicaciones web

- SQL injection
- SQL batch
- Salto de barda.

Se estableció la prueba de seguridad de plataforma de la aplicación de acuerdo a las recomendaciones para aplicaciones web.

Se eligieron grupos de usuarios y claves validas e invalidas.
Se verificaron con el SM los usuarios y privilegios en la base de datos

Se seleccionó un ejemplo de cada rol en el sistema y se verificó a que tenía acceso.
Se intentó ingresar los código comunes a esta práctica como ‘or 1=1
Se identificaron las páginas internas y se intentó entrar sin pasar por el módulo de ingreso
Se verificaron las recomendaciones de seguridad para Windows Server, IIS, ASP.NET, y SQL server.

Pruebas Modulares
Se verifico la funcionalidad de los *módulos de Reportes y Consultas de órdenes*

Se diseñaron tipos de reportes de acuerdo a los requerimientos y se verificaron los datos obtenidos.
Ordenes de servicio Actuales
Ordenes de servicio por periodos
Ordenes de servicio Canceladas
Ordenes de servicio por canal
Ordenes de servicio de un canal

Actas de Recepción Actuales
Actas de Recepción por periodo
Actas de Recepción Canceladas
Actas de Recepción por canal
Actas de Recepción de un canal

Estudios de factibilidad por estatus
Estudios de factibilidad por estatus x periodo por canal

Se verificó la funcionalidad de la pantalla de consultas, probando su funcionalidad (ordenando cada columna, buscando una palabra por columna) y visualización de una orden.

Pruebas Compatibilidad
Se estableció la prueba de acceso mediante Microsoft Office.

Se configuro un usuario de solo lectura mediante OLEDB.
Se acceso la base desde una base de Access y una hoja de Excel usando estos drivers.

Prueba de Volumen

Se planteó el establecimiento de pruebas de volumen por medio de una aplicación en ambiente de producción.

Se planteó la construcción de 2000 registros de los diferentes documentos y un tipo de reporte por periodo grande.

Pruebas de Integridad de base de datos.

Se plantearon dos pruebas a realizar
Con los datos ejemplo.
Con los datos históricos proporcionados.

Se verificó que tuvieran la información ingresada en cada documento y su relación en la tabla respectiva.

Se verificaron fechas de creación, entrega, recepción, documentos, claves anteriores.

Pruebas de Calidad de Diseño.

Se planteó como calidad del diseño la eficiencia del código (número de páginas) y las secciones en las mismas.

Se entregaron las páginas en la versión 2.0
Se explicó y mostro la creación de las mismas modularmente.

8. Beneficios del Sistema

Los beneficios de un sistema se definen como aquellas actividades o funciones que tienen una mejora con respecto a las existentes antes del uso del mismo.
La valuación del beneficio entonces debe de tomar en cuenta:

- La comparación entre las funciones anteriores, y sus deficiencias, y las nuevas.
- El costo de sustituirlas.
- El costo de desarrollar nuevas
- El tiempo que duraran los beneficios.
- Los beneficios alternos.
- El costo de mantenerlos.

Considerando lo anterior cabe recordar en la etapa de diseño donde se establece cuáles son las capacidades y recursos con los que cuenta una organización para el desarrollo de una aplicación y si los requerimientos están de acuerdo a esta capacidad. Acotar las capacidades del sistema a los recursos de una organización puede ser la clave para el mejor costo beneficio del desarrollo de una aplicación.

El uso de sistemas es una ventaja competitiva. Cuando una organización invierte en la realización de sistemas las inversiones parecen elevadas, sin embargo debe considerarse que el uso de sistemas es una necesidad y mientras más pronto la empresa se acostumbre al uso de la tecnología, los beneficios serán mayores.

Los beneficios del sistema desarrollado son los establecidos en sus requerimientos además de los alternos por el uso de nueva tecnología.

- La información necesaria esta en un solo lugar.
- Se pueden resolver preguntas como:
  - ¿Cuantas órdenes de servicio se han realizado?
  - ¿De quién son estas órdenes?,
  - ¿Cuántas órdenes tiene un canal comercial?
- Agiliza los procesos de atención de las áreas.
- Se puede dar seguimiento a una Orden a través de los procesos.
- Ahorra tiempo y papeleo.
- Se puede saber cómo están funcionando los procesos.
- La información está disponible en cualquier momento y desde cualquier lugar.
- Los reportes se realizan en cualquier momento.
- Se pueden desarrollar nuevos tipos de reportes.
- Se pueden acceder los datos para realizar distintos reportes.
- Los procesos y el sistema pueden modificarse.
- Se puede añadir nueva funcionalidad.
- Se explotan mejor algunas herramientas ya adquiridas.
- Se tiene acceso a nuevas herramientas y tecnología.
- El personal se encuentra mejor capacitado para la explotación de las herramientas.
- Se cuenta con herramientas comerciales para realizar la administración y mantenimiento de las diferentes herramientas.

Los mayores beneficios del sistema implantado se verán con el tiempo.
Cuando se implanta un sistema, donde no lo había, se presenta una resistencia normal al uso, la adecuación de procesos hace menos agiles al principio los mismos. Sin embargo una vez madurados los distintos procesos y el uso del sistema, los beneficios son claros. En este punto los

usuarios empiezan a pedir nuevas funcionalidades o mejoras y la explotación y necesidad de la información es cada vez mayor.

9. Plan de Mantenimiento.

Introducción

El ciclo de vida del software incluye:

- Requerimientos
- Diseño
- Construcción
- Pruebas
- Mantenimiento.

El mantenimiento del software es una parte integral del ciclo de vida del software. Sin embargo históricamente no recibe la misma atención, ni recursos, como las otras fases. Históricamente el desarrollo ha tenido mucha más atención y prioridad que el mantenimiento en las organizaciones. Actualmente esto está cambiando y las organizaciones tratan de obtener más de sus inversiones en desarrollo haciendo que el software se mantenga funcionando por el mayor tiempo posible.
Cuestiones como el Y2K son ejemplos clásicos de la atención del concepto de mantenimiento. Además conceptos como el código abierto ha puesto especial atención al mantenimiento del código desarrollado por otros. Más aun el mantenimiento sigue siendo caro. Por estas razones es importante enfocarse en mayor desarrollo y lograr mejor productividad de las actividades de mantenimiento.

Definición de mantenimiento de software

Los esfuerzos en el desarrollo de software dan como resultado un producto de software que satisface los requerimientos del usuario. De acuerdo con lo anterior el software debe cambiar o evolucionar. Una vez en operación, las anomalías son descubiertas, los ambientes operativos cambian y nuevos requerimientos del usuario surgen

La fase de mantenimiento del ciclo de vida del software comienza después de la puesta en marcha, pero las actividades del mantenimiento comienzan mucho antes.

El mantenimiento del software sostiene al producto a través de su ciclo de vida. Las solicitudes de modificación son enlistadas y atendidas, el impacto de los cambios es determinado, el código es modificado, las pruebas son realizadas y una nueva versión del producto es liberada. El entrenamiento es proporcionado al usuario.

Estructura del Mantenimiento

Los conceptos para el mantenimiento de software son conceptos usados en ingeniería de software que son generalmente aceptados por la comunidad en el mantenimiento de software.
Estos son generales y no pertenecen a ningún dominio, modelo, o necesidad de negocio. Estos conceptos pueden ser usados en organizaciones pequeñas y medianas. Estos conceptos son consistentes con los encontrados en la literatura de ingeniería de software y estándares. Conceptos como Calidad, Mediciones y estándares se incluyen en estos conceptos.

Conceptos básicos

El concepto de mantenimiento de software es definido en los estándares de la IEEE 1219 como:

La modificación de productos de software después de la liberación para corregir fallas, mejorar el rendimiento o atributos, o para adaptar el producto a un entorno distinto.

El estándar ISO/IEC 12207 sobre los procesos del ciclo de vida esencialmente describen al mantenimiento de software como uno de los primeros procesos del ciclo de vida del software y lo describe como La modificación del código y su documentación asociada a un problema o necesidad de mejora. El objetivo es modificar el software actual mientras se preserva su integridad. El estándar ISO/IEC 12207 describe una actividad llamada "Procesos de implementación" cuya actividad establece el plan de mantenimiento y los procedimientos que serán usados durante el proceso de mantenimiento.

El ISO/IEC 14764 [ISO14764], El estándar internacional para el mantenimiento de software define al mantenimiento igual que el anterior pero pone énfasis en las actividades de mantenimiento de pre liberación como la planeación.

La definición generalmente aceptada por desarrolladores y encargados es:

La totalidad de las actividades requeridas para proveer un costo de soporte efectivo a un sistema. Las actividades son realizadas tanto durante las etapas de pre liberación como en las de post liberación, soportabilidad y determinación logística. Las actividades de pre liberación incluyen la modificación de software, entrenamiento y operación del help desk.

Un encargado de mantenimiento es definido por el ISO/IEC 12207 como una organización que realiza las actividades del mantenimiento.

- Proceso de Implementación
- Análisis de problemas y modificaciones
- Implementación de modificaciones
- Revisión de modificaciones y aceptación
- Migración y eliminación

A que se le da mantenimiento

Una percepción común es que el mantenimiento es solo arreglar errores. Sin embargo diversos estudios desarrollados durante años indican que cerca del 80 % de las actividades de mantenimiento son actividades no correctivas. Este esfuerzo es similar al desarrollo del software y sin embargo solo se le dedica un proceso.

Los encargados del mantenimiento observan el desarrollo del producto y el presente y trabajando con el usuario y los operadores se anticipan a los problemas y consideran cambios funcionales.

Algunos autores están de acuerdo con que el mantenimiento tiene que cubrir más conceptos que el mismo desarrollo, ya que tiene más cambios que conducir y controlar.

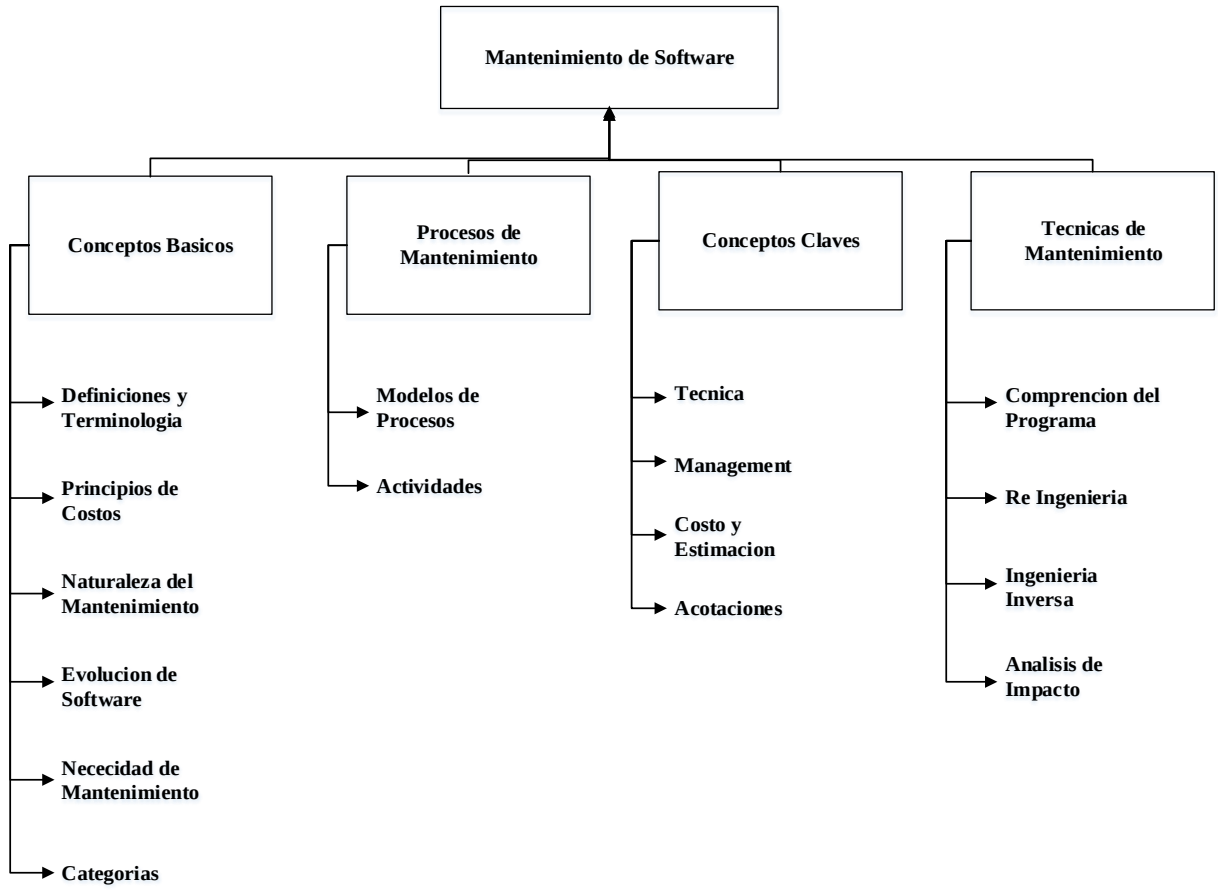


Figura 5.9.1 Estructura del Mantenimiento

Evolución de Software

El área de mantenimiento de software y evolución de los sistemas fue postulada por primera vez por Lehman en 1969. Postulando las ocho leyes de la evolución del software:

1. **Cambio continuo:** Un programa que se usa en un entorno real necesariamente debe cambiar o se volverá progresivamente menos útil y menos satisfactorio para el usuario.[3]
2. **Complejidad creciente**: A medida que un programa en evolución cambia, su estructura tiende a ser cada vez más compleja. Se deben dedicar recursos extras para preservar y simplificar su estructura.[3]
3. **Autorregulación** La evolución de los programas es un proceso autoregulado. Los atributos de los sistemas, tales como tamaño, tiempo entre entregas y la cantidad de errores documentados son aproximadamente invariantes para cada entrega del sistema.[3]
4. **Estabilidad organizacional:** Durante el tiempo de vida de un programa, su velocidad de desarrollo es aproximadamente constante e independiente de los recursos dedicados al desarrollo del sistema.[3]
5. **Conservación de la familiaridad**: A medida que un sistema evoluciona todo lo que está asociado con ello, como los desarrolladores, personal de ventas, y usuarios por ejemplo, deben mantener un conocimiento total de su contenido y su comportamiento para lograr una evolución satisfactoria. Un crecimiento exagerado disminuye esta capacidad. Por tanto este incremento promedio debe mantenerse.[3]
6. **Crecimiento continuado:** La funcionalidad ofrecida por los sistemas tiene que crecer continuamente para mantener la satisfacción de los usuarios.

7. **Decremento de la calidad**: La calidad de los sistemas software comenzará a disminuir a menos que dichos sistemas se adapten a los cambios de su entorno de funcionamiento.
8. **Retroalimentación del sistema**: Los procesos de evolución incorporan sistemas de retroalimentación multi agente y multi bucle y estos deben ser tratados como sistemas de retroalimentación para lograr una mejora significativa del producto.

Lehman estableció que el mantenimiento es un desarrollo evolucionando y que estas decisiones de mantenimiento son ayudadas por el entendimiento de lo que ocurre con los sistemas en el tiempo

Otros autores establecen que el mantenimiento es en realidad un desarrollo continuo con la excepción o limitación, que el sistema existe.

Necesidad de Mantenimiento
Algunas de las normas de mantenimiento establecen que:

- El mantenimiento es necesario para asegurar que el sistema continuara satisfaciendo las necesidades del usuario.
- El mantenimiento es aplicable al desarrollo de sistemas mediante cualquier modelo de desarrollo de software (Por ejemplo, espiral).
- El sistema cambia debido a las acciones correctivas y no correctivas.

El mantenimiento debe ser realizado con el fin de:
- Corregir errores.
- Corregir requerimientos y defectos de diseño.
- Mejorar el diseño.
- Hacer mejoras.
- Realizar interfaz con otros sistemas.
- Convertir programas para que distintos tipos de hardware, software, sistema de telecomunicaciones, puedan ser usados.
- Migrar los sistemas heredados.
- Eliminar sistemas.

Los cuatro aspectos más importantes del mantenimiento se centran en:

- Mantener el control del sistema sobre las funciones del día a día
- Mantener el control del sistema por encima de las modificaciones.
- El perfeccionamiento de las funciones existentes.
- Prevenir que el rendimiento del sistema caiga a niveles críticos

Categorías del mantenimiento

Las categorías definidas por el estándar ISO/IEC 14764, son:

Mantenimiento correctivo. Modificación reactiva de un producto de software que se realiza después de la entrega para corregir problemas descubiertos.

Mantenimiento adaptativo. La modificación de un producto de software que se realiza después de la entrega para mantenerlo utilizable en un entorno modificado o cambiarte.

Mantenimiento perfectivo. La modificación de un software después de la entrega del producto para mejorar el rendimiento o mantenibilidad.

Mantenimiento preventivo. La modificación de un software que se realiza después de la entrega para detectar y corregir fallas latentes en el producto antes de que se conviertan fallas eficaces.

La Norma ISO sobre Mantenimiento de Software [ISO14764] clasifica al mantenimiento adaptativo y perfectivo como mejoras y al mantenimiento correctivo y preventivo como correcciones.

Proceso de mantenimiento

El *Modelo de Madurez de Capacidades* (SW-CMM) proporciona un medio para medir los niveles de madurez de los procesos de una empresa. Lo importante, es que hay una correlación directa entre los niveles de madurez y ahorro de costos. Cuanto mayor sea el nivel de la madurez, mayor es el ahorro de costos. El SW-CMM se aplica igualmente al mantenimiento y encargado del mantenimiento deben mantener un proceso documentado

Modelos de los procesos de mantenimiento

El modelo de los procesos de mantenimiento descrito en la norma IEEE 1219 establece las siguientes fases:

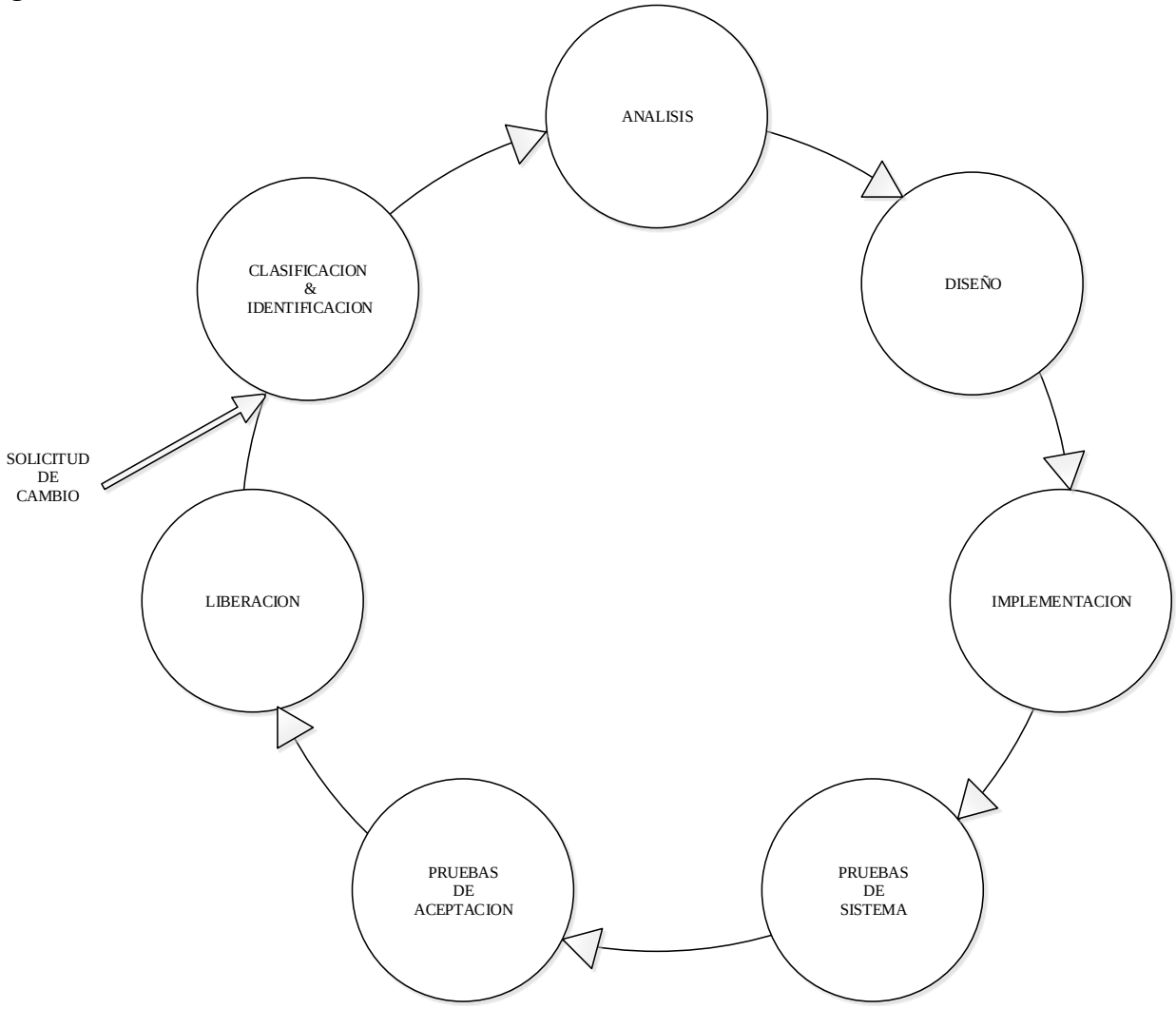


Figura 5.9.2 Actividades del Mantenimiento

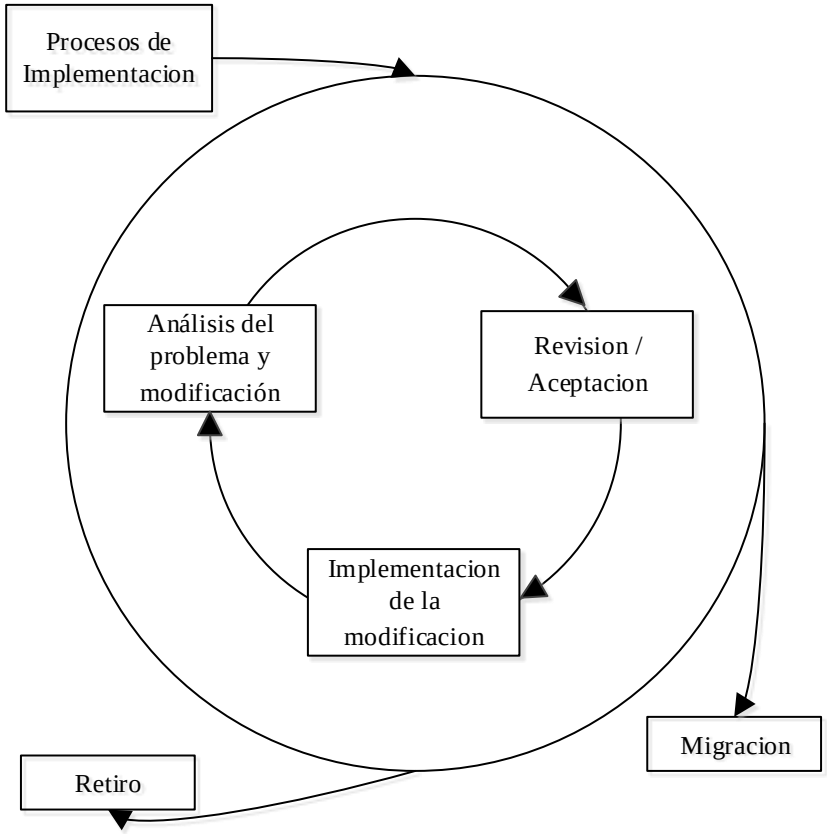


Figura 5.9.3 Actividades del Mantenimiento ISO / IEC

Las actividades de los procesos de mantenimiento desarrolladas en ISO / IEC 14764 se divide en tareas de la siguiente manera.

En el proceso de implementación

- Desarrollar planes y procedimientos de mantenimiento.
- Establecer procedimientos de solicitudes de modificación.
- Implementar el proceso de CM.

Problemas y modificaciones

- Con las primeras mediciones.
- Verificar el problema.
- Desarrollar opciones para implementar la modificación.
- Documentar los resultados.
- Obtener la aprobación para la opción de modificación.

Modificación e implementación

- Realizar un análisis detallado.
- Desarrollar, el código y probar la modificación.

Revisión de mantenimiento / tareas de aceptación

- Dirigir entrevistas, inspecciones.
- Obtener la aprobación de la modificación.

La migración:

- Garantizar que la migración está de acuerdo con la norma ISO / IEC 12207.

- Desarrollar un plan de migración.
- Notificar a los usuarios de los planes de migración.
- Llevar a cabo operaciones paralelas.
- Notificar al usuario que ha iniciado la migración.
- Llevar a cabo una revisión posterior a la operación.
- Asegúrese de que los datos viejos estén accesibles.

Eliminación/Retiro

- Desarrollar un plan de retiro.
- Notificar a los usuarios de los planes de jubilación.
- Llevar a cabo operaciones paralelas.
- Notificar al usuario que el retiro se ha iniciado.
- Asegúrese de que los datos viejos estén accesibles.

Actividades de Mantenimiento

Las actividades para el mantenimiento son similares al del desarrollo del software.
Los responsables del mantenimiento deben:

Realizar análisis, diseño, codificación, pruebas y documentación.
Realizar el seguimiento de los requerimientos
Mantener la documentación actualizada.

Actividades solo para Mantenimiento.

Los encargados del mantenimiento deben poseer un conocimiento íntimo de la estructura del código y contenido. Ese conocimiento es usado para de realizar análisis de impacto. El análisis de impacto identifica todos los sistemas y productos de sistemas afectados por una solicitud de cambio y desarrolla una estimación de los recursos necesarios para llevarla a cabo.
Además, se determina el riesgo de hacer el cambio.
La solicitud de cambio, a veces llamado una solicitud de modificación y, a menudo llamado un reporte de problemas, debe ser el primero analizado y traducido en términos de software. Los encargados entonces, identifican los componentes afectados. Varias soluciones se proporcionan a continuación una recomendación sobre el mejor curso de acción.

Actividades de apoyo

Los encargados del mantenimiento podrían también realizar actividades complementarias como la configuración de la gestion (CM), verificación, control de calidad, revisiones, auditoria y la formación de usuarios. El mantenimiento de software para la norma IEEE 1219 describe la gestión de configuración como un proceso fundamental. Los procesos de CM deben proveer verificación, validación, certificación de cada paso para identificar, autorizar, implementar y liberar un producto de software. El entrenamiento para los encargados es también una actividad necesaria.

Configuración de la gestión (CM).

El software y cualquier cambio realizado se deben controlar. Este control se debe realizar mediante un proceso y un programa aprobado para la gestión de configuración SMC. El proceso SMC es implementado para desarrollar y seguir el plan de configuración de gestión y

procedimientos de operación. Los encargados participan en juntas de control para determinar cuándo las mejoras se deben detener y proceder a la migración.

Calidad
Las actividades y técnicas para el aseguramiento de la Calidad del Software (SQA) y V & V debe seleccionarse en concordancia con todos los demás procesos para alcanzar el nivel de calidad deseado.

Actividad de Planificación del Mantenimiento

Mientras que se considera que generalmente los desarrollos duran de uno a dos años la fase de mantenimiento dura varios años. Planear los recursos necesarios para el mantenimiento es necesario. Los planes para e l mantenimiento deben empezar considerando objetivos de calidad.
Un concepto y un plan de mantenimiento deben de ser desarrollados. El concepto debe manejar:

- El alcance del mantenimiento del software.
- La adaptación del proceso después de la liberación.
- La designación de quién va a proporcionar el mantenimiento.
- Una estimación de los costes del ciclo de vida.

Después de esto se prepara el plan de mantenimiento (IEEE 1219 y ISO/IEC 14764).

Cuestiones clave en el Mantenimiento.

Es importante entender que el mantenimiento del software proporciona problemas técnicos y de gestión específicas para ingenieros de software.
Encontrar un error en 500 mil líneas de código es todo un desafío. La competencia por recursos con los desarrolladores también lo es al igual que la planeación de una versión futura mientras se envían los parches de emergencia. Algunos de los problemas y retos claves del mantenimiento son:

Los problemas técnicos como:
- Comprensión limitada
- Pruebas
- Análisis de Impacto
- Mantenibilidad

La gestión
- Alineamiento con la Organización
- Equipos de trabajo.
- Cuestiones de proceso
- Aspectos de Organización
- El encargado del mantenimiento.
- El Outsourcing
- Estructura Organizacional.

El costo y la estimación del Mantenimiento
- Costo
- Estimación del Costo
- Los modelos paramétricos
- Experiencia

Finalmente
Dimensionamiento del Mantenimiento de Software
Medidas específicas

- Tamaño del software
- Personal de sistemas
- Solicitudes de mantenimiento número/estado
- Mejoras número y estatus
- Utilización de los recursos de la computadora
- Densidad de fallas
- Volatilidad de software
- Informe de discrepancia de Tiempo de apertura
- Radio de error/compostura
- Fiabilidad del software
- Complejidad de diseño
- Distribución del tipo de falla

Las técnicas para mantenimiento

El mantenimiento efectivo se logra utilizando técnicas específicas para el mantenimiento.

- Programa de Comprensión
- Reingeniería
- La ingeniería inversa
- Análisis de impacto

Mantenimiento de la aplicación
Como se puede leer la etapa de mantenimiento es de importancia crítica y muy costosa, para el caso de la aplicación de órdenes, el mantenimiento quedo a cargo, como se estableció en los requerimientos, de la empresa.

Se planteó la utilización del software del help desk para atender los requerimientos de la aplicación. Además se establecieron las rutinas de respaldo y actualizaciones a las diferentes plataformas (SO, WEB, ASP.NET, SQL Server, etc.) parte de las actividades del encargado de la plataforma.

Las modificaciones al código y a la funcionalidad del sistema quedo a cargo del encargado capacitado para ello.

6. *CONCLUSIONES*

Una vez realizada la entrega del sistema podemos concluir:

La mayoría de los requerimientos iniciales fueron cubiertos. Las etapas de pruebas sirvieron a su propósito y las iteraciones concluyeron satisfactoriamente. Aunque no totalmente previsto el cambio en algunos procesos de la empresa, pudieron ser enfrentados con éxito, lo que incluye la modificación de algunos requerimientos iniciales.

El sistema logra realizar los objetivos para los que fue diseñado inicialmente. Se pueden observar los cambios en los procesos y la seguridad que da el tener una sola fuente de información. La presentación ante los socios comerciales causo una buena impresión y la confianza que se planeaba en los requerimientos. Los reportes son realizados sin ningún esfuerzo y se está planeando la realización de nuevos.

El sistema proporciona beneficios propios del manejo del mismo, pero también beneficios en la utilización de nuevas tecnologías que les será de utilidad a la empresa.

La implementación del sistema presento múltiples problemas entre los empleados, debido a la organización y la falta de experiencia del personal. La capacitación del personal encargado fue particularmente complicada debido a los pocos recursos asignados para ello.

La implementación de este sistema deja mucha experiencia de la relación que debe existir entre cliente, usuario, implementador. Un manejo inadecuado de estas relaciones puede ocasionar múltiples retrasos y problemas.

La entrega del código para su modificación y mantenimiento no es una tarea fácil de lograr para una empresa pequeña con recursos limitados. La madurez en el negocio, la buena administración, el conocimiento de las herramientas y sistemas son necesarios para entender que las aplicaciones y sistemas son una ventaja competitiva en cualquier negocio y requieren de recursos y atención.

El control de la aplicación, mantenimiento, modificación, le permitirán a la empresa lograr uno de los objetivos principales modificar la funcionalidad de la aplicación conforme sus procesos lo requieran. El negocio cambiante, las nuevas tecnologías y las condiciones de la empresa pueden ocasionar que el ciclo de vida del sistema sea muy breve pero se espera que este sea un apoyo en la madurez de la empresa.

Los sistemas y servicios prediseñados son una alternativa viable para la empresa pequeña que le permiten alcanzar la madurez necesaria en muchos aspectos.

Para lograr el desarrollo de la aplicación fue necesario contar con la experiencia necesaria, no solo de las técnicas de desarrollo de aplicaciones sino técnicamente en la programación y conocimiento de las diferentes herramientas utilizadas, la experiencia en la realización de otros sistemas, y de gerencia y administración que permitieron generar y presentar propuestas de solución al cliente.

Los conceptos aprendidos en la carrera permiten iniciar el desarrollo profesional. Las capacidades desarrolladas durante la carrera, como la resolución de problemas, la adquisición de conocimientos y el desarrollo de pensamiento analítico son herramientas fundamentales en el ámbito profesional. Permiten enfrentar el diseño de aplicaciones, trabajar con distintos

profesionales y encaminarse en la rama del conocimiento que más llame la atención y adquirir la experiencia necesaria

.

7. *REFERENCIAS*

1. Bibliografía

Analisis estratégico para el desarrollo de la pequeña y mediana empresa
Universidad Nacional Autonoma de Mexico
Universidad Politecnica de Cartagena
Universidad Veracruzana
Universidad de Murcia
Dr. Domingo García Pérez de Lema
2006

Proceso Contable
Graciela Picazo Cornejo
ISBN 978-607-733-167-4
2012

Diseño de un modelo de planificación estratégica
Isamelys Velásquez
Reseña curso
2013

Porque Mueren las Pymes
Juan Carlos Fernández
Presentacion
2008-2010

La problemática de las PYMES en Mexico, bajo un análisis de competitividad sistematica.
Catedra de Contabilidad y administracion
Universidad de Occidente
Dr. Jorge Sánchez Sandoval
PDF.
Mayo 2015

Scrum y XP desde las trincheras
Cómo hacemos Scrum
Henrik Kniberg
Traducción
C4Media Inc
2007

Sistemas de información para la administración
James A Senn
Grupo editorial Latinoamericano
Traducción
Information systems in management
Ed 1990

Análisis y Diseño de Sistemas
Kendall & Kendall
Prentice Hall
Tercera edición 1997

Administración de los sistemas de información
Organización y tecnología
Kenneth C. Laudon
Jane P. Laudon
Prentice Hall
Tercera edición 1996

Análisis y diseño de sistemas de información
James a Senn
Segunda edición 1995
Mc Grawn Hill

Diseño de páginas web interactivas con JavaScript
Juan Carlos Oros Cabello
Segunda edición
Alfaomega
ISBN 970-15-0518-2

Programacion de Active Server Pages
Scot Hillier
Daniel Mezick
Microsoft Press
Mc Graw Hill
ISBN 1-57231-700-0

You Don't Know JS: Up & Going
Kyle Simpson
O'Reilly Media
Ebook
March 2015
ISBN: 978-1-4919-2441-9

Analisis y Diseño Estructurado Moderno
Edward Yourdon
Prentice Hall Hispanoamericana S.A.
PDF book
1993
ISBN:968-860-303-0

Software Maintenance
Thomas M. Pigoski
Technical software services Inc.
Capitulo 6.
IEEE – Trial Versión 1.00
Mayo 2001

IEEE Computer Society
Annual International Conference on Software Maintenance
(ICSM).

FDD Processes US Letter

Metodología FDD
Catedra de ingeniería de Software
Universidad ORT Uruguay.
Luis Calabria
2003

Extreme Programming Explained. Embrace Change
Beck, K.
Pearson Education,
1999

Métodologías ágiles para el desarrollo de software
eXtreme Programming (XP)
Patricio Letelier
Mª Carmen Penadés
Universidad Politécnica de Valencia (UPV)
2009

2. Internet

http://msdn2.microsoft.com/en-us/library/bb266332.aspx The Architecture Journal

http://msdn2.microsoft.com/en-us/library/ms972975.aspx#usercontrols_topic6 Adding Properties and Methods to a User Control

http://msdn2.microsoft.com/en-us/library/ms972975.aspx#usercontrols_topic9 Creating and Raising a Custom Event

http://support.microsoft.com/kb/307598 ASP.NET State Management Overview. ViewState in ASP.NET.

http://www.extremeexperts.com/Net/Articles/ViewState.aspx ASP.Net ViewState Overview.

http://msdn2.microsoft.com/en-us/library/wtxbf3hh.aspxASP.NET Master Pages Overview (Microsoft Developer Network)

http://msdn2.microsoft.com/en-us/library/ex526337.aspx ASP.NET Web Site Layout from MSDN

https://es.wikipedia.org/wiki/CategoriaADa:Plataforma_.NET

https://es.wikipedia.org/wiki/CategoriaADa:Frameworks_para_aplicaciones_web

https://msdn.microsoft.com/es-es/library/7sw4ddf8%28v=vs.94%29.aspx

https://www.w3.org/TR/html401/intro/sgmltut.html#h-3.2.2.

http://technet.microsoft.com/en-us/scriptcenter/bb410849.aspx

http://msdn2.microsoft.com/en-us/library/zw39ybk8.aspx VisualBasicScript

http://msdn.microsoft.com/es-es/library/bb500435.aspx Novedades de SQL Server 2014

http://www.microsoft.com/spain/sql/default.mspx Página web de Microsoft SQL Server

https://msdn.microsoft.com/en-us/library/aa479011.aspx#aspnet-usingjavascript_topic

https://msdn.microsoft.com/es-MX/library/aa711638%28v=vs.71%29.aspx/visual basic .net

https://msdn.microsoft.com/es-es/library/zfzfkea6%28v=vs.100%29.aspx/controle servidor web

https://msdn.microsoft.com/es-es/library/system.web.ui.htmlcontrols.htmlgenericcontrol%28v=vs.100%29.aspx

https://en.wikipedia.org/wiki/Relational_database_management_system

http://technet.microsoft.com/es-es/library/bb500435%28v=sql.100%29.aspx/novedades de sqlserver 2008

https://technet.microsoft.com/es-es/library/ms166026%28v=sql.90%29.aspx#feedback/referencias sqlserver

3. Anexos

(i) Anexo Grandes Proveedores

TELMEX
Seguridad y calidad, con gran capacidad de acceso dedicado y el mayor ancho de banda de América Latina.

Velocidades disponibles:

- E1 (2,048 Kbps)
- E3 (34 Mbps)
- STM1 (155 Mbps)
- FAST ETHERNET 4 Mbps hasta 100Mbps
- GIGA ETHERNET 100Mbps hasta 1 Gbps

- Ideal para más de 30 usuarios y diferentes aplicaciones, como: servidores web, de correo electrónico, FTP, comercio electrónico, etc.
- Se entregan desde 16 Direcciones IP Homologadas (Fijas), en el protocolo IPV4, y de igual forma se soporta el protocolo IPV6 con el que se entrega un bloque de direcciones de tamaño /48 equivalente a 65,500 subredes , garantizando el mejor desempeño de sus aplicaciones en Internet.
- Puede integrar Módulo de Seguridad y Módulo Data Center.
- Soporte vía telefónica o en sitio, las 24 horas del día y los 365 días del año. Con un tiempo máximo de atención de 4 horas.
- Incluye: Herramienta de Monitoreo WEB, Registro de Dominio ante el NIC , Consultoría y Diseño de su Red, Soporta Aplicaciones Server, Soporte técnico en sitio.
- Servicio de Seguridad Perimetral opcional que ofrece: Implementación Tecnológica (FW,IPS, VPN, Filtrado de Contenido) Monitoreo Avanzado, Gestión, Seguridad Perimetral, Soporte Técnico, Indicadores.

Modalidades del servicio

Servicio Básico
Conexión dedicada a Internet.

Servicio Total
Conexión dedicada a Internet que incluye Valores agregados Triara tales como:

- Hospedaje WEB
- Conferencia WEB
- Correo Negocios
- Monitoreo en línea vía web.
- Registro de dominio ante el NIC.

Esquema Puerto Extendido.

- Enrutadores de acceso y el equipo de comunicación necesario.
  La mejor tecnología y una infraestructura de soporte técnico en ingeniería, para atender remotamente o en sitio cualquier incidencia de la solución.
- Mantenimiento de Equipos.

Esquemas de Facturación por Consumo, en base a la utilización del servicio.

Este servicio aplica para velocidades E1, E3 (34 Mbps) y STM1 (155 Mbps), 50 Mbps, 100Mbps, 250Mbps, 500Mbps, 750Mbps y 1 Gbps.

Todas las modalidades de Internet Directo Empresarial cuentan con asesoría especializada de las áreas de consultoría y diseño.

Requisitos técnicos.

- Equipo de ruteo con interfaces V.35 para par de cobre o G.703, para fibra óptica.
- Las Velocidades E3 (34 Mbps) y STM1 (155 Mbps) se necesitará realizar una consultoría, para determinar el tipo de interfaz.

Comparación de Servicios.

	Infinitum Negocio	Infinitum Negocio Red	Infinitum Negocio Premium	Internet Directo empresarial
Velocidad	Recepción hasta **2 Mb** Transmisión hasta **384 Kbps**	Recepción hasta **4 Mb** Transmisión hasta **640 Kbps**	Recepción hasta **6 Mb** Transmisión hasta **768 Kbps**	Nx64 E1 (**2,048 Kbps**) E3 (**34 Mb**) STM1 (**155 Mb**)
Vía de acceso	Por línea de teléfono	Por línea de teléfono	Por línea de teléfono	Por enlace dedicado
Correo	**Correo Negocio** 2 direcciones de correo con el nombre de su negocio	**Correo Negocio** 2 direcciones de correo con el nombre de su negocio	**Correo Negocio** 2 direcciones de correo con el nombre de su negocio.	---
Ideal para	Hasta 5 usuarios	Hasta 10 usuarios	Hasta 30 usuarios	Más de 30 usuarios
Direcciones IP	**IP** Dinámica	**IP** Dinámica	**IP** Dinámica	Desde **16 Direcciones IP** Homologadas (fijas)
Tiempos de atención a incidencias	Máximo 72 horas	Máximo 72 horas	Máximo 72 horas	Máximo 4 horas

Figura 7.3.1 Caracteristicas de servicios

Internet Directo Empresarial es ideal para todo tipo de empresas o negocios, que desean conectar una red, de más de 30 computadoras y/o ejecutar aplicaciones en Internet, garantizando el ancho de banda contratado.

Internet Directo Empresarial es especialmente apropiado, para empresas que buscan aplicaciones como:

- Atención a Clientes en línea o vía correo electrónico.
- Conectar redes de mas de 30 computadoras, para salidas a Internet.
- Obtención de información a nivel mundial.
- Servidores de correo electrónico.
- Transferencia de archivos (FTP).
- Velocidad síncrona.
- Implantación de sitios para comercio electrónico.
- Acceso a redes privadas virtuales (VPN).
- Educación a distancia.
- Transmisión por Internet en vivo (Video Streaming). Actualmente nuestros Clientes son Empresas del Sector Industria, Financiero, Gobierno, Servicios y Turismo, proveedores de servicios de Internet, tiendas departamentales, universidades y Pequeña y Mediana Empresa.

Soluciones de IT
La integración y administración de tecnologías y procesos mejora la productividad y operación de su empresa

Servicios en la nube

- Factura electrónica
- Servidores virtuales
- LMS e-learning
- Escritorio virtual
- Transmisión de audio y video

Servicios de data center

- Correo corporativo
- Co ubicación
- Almacenamiento y respaldo
- Monitoreo de sistemas
- Administración de sistemas
- Hospedaje dedicado

Desarrollo de software

- Fábrica de software
- Fábrica de pruebas

Conectividad y transporte

Mejore su conectividad para cualquier tipo de protocolo de accesos a su red, con anchos de banda flexibles

Servicios de red privada

- Red VPN
- Enlaces Ethernet
- Enlaces
- Enlaces seguros
-

Servicios de internet

- Internet directo empresarial
- Infinitum empresarial
- I Hospitality

Servicios agregados

- Tráfico seguro

AXTEL

Empresa mexicana de telecomunicaciones con mayor crecimiento en el segmento de banda ancha, y una de las compañías con soluciones TIC (Tecnologías de Información y Comunicación)

Es una empresa de telecomunicaciones que ha invertido más de 43 mil millones de pesos en la creación de infraestructura básica de acceso.

Es el segundo operador más grande de telefonía fija y de larga distancia de México.

Cuenta con una red propia en 39 de las principales ciudades de México, así como conectividad en 200 ciudades del territorio nacional.

La extensión de su red de fibra es de más 12 mil 200 kilómetros, incluyendo más de mil 800 kilómetros de anillos metropolitanos y más de 2 mil 800 kilómetros de red FTTH. Opera la red inalámbrica fija más grande del mundo.

Sus ventas anuales son superiores a los 10 mil millones de pesos.

Atiende todos los segmentos de mercado (hogares, empresas, gobierno, bancos y otros operadores de telecomunicaciones) con servicios de banda ancha, redes privadas virtuales, data centers, seguridad administrada y servicio de voz.

Cuenta con los recursos tecnológicos más avanzados del mercado para proveer las mejores soluciones de comunicación

Las tecnologías de acceso que emplea son: cable de fibra óptica, acceso inalámbrico fijo, radio punto a punto, radio punto a multipunto y tecnología de cobre.

La Red Digital está constituida por la combinación de las mejores soluciones tecnológicas existentes en el mercado de las telecomunicaciones. Éstas son resultado de años de investigación, desarrollo y experiencia por parte de compañías líderes en el mundo.

Ser una compañía joven le ha permitido a capitalizar las enormes ventajas de los avances tecnológicos para proporcionar a sus clientes un servicio confiable y de alta calidad a un costo sumamente competitivo.

Las ventajas que se desprenden de su Red Digital son:

- Red totalmente digital. Contar con una red totalmente digital hace posible brindar un servicio con la mejor calidad, seguridad, confiabilidad y velocidad, que la ofrecida a través de la transmisión analógica.

- Confiabilidad. La Red Digital cuenta con los atributos necesarios para procurar la provisión de servicios ante cualquier eventualidad.
- Rápida instalación. Los servicios se pueden instalar y activar el mismo día.
- Fáciles actualizaciones y ampliaciones. Al ser totalmente digital, la red de AXTEL permite efectuar actualizaciones de equipo o ampliación de servicios sin ninguna dificultad. Flexibilidad. Permite que sus clientes puedan integrar fácilmente diferentes soluciones de comunicación de acuerdo a sus necesidades.

Servicios que ofrece:

- Líneas Privadas para tu Empresa.
- Comunicación directa y eficaz
- Lan to Lan para Carriers. Enlaces punto a multipunto
- Lan to Lan para tu Empresa. Conexión segura de alta velocidad
- Multiservicios IP en tu Empresa. Conexión integral
- VPN Banda Ancha para tu Empresa. Transferencia Segura de datos

Líneas Privadas

Conecta las oficinas de tu empresa con una comunicación directa y eficiente a través de un canal digital 100% privado, que te permite el transporte simultáneo de voz, datos y video de forma confidencial y segura a nivel local, nacional e internacional.
Modalidades del Servicio

Punto-Punto
Interconexión que une dos localidades en distinta ubicación. Para este servicio, los anchos de banda disponibles son de n x 64 Kbps hasta E1.

Punto - Multipunto
Interconexión basada en la formación de redes “Hub & Spoke” conocida como estrella, que permite enlazar varias localidades remotas a un nodo central de mayor capacidad. Este servicio tiene anchos de banda disponibles para localidades remotas de n x 64 Kbps hasta E1 y para nodos centrales con capacidades de E1 o E3.

Estructura Tarifaria

Para las Líneas Privadas Nacionales, el cliente deberá contratar los Accesos Dedicados Correspondientes y un Circuito Inter-nodal Nacional, para el cual aplica la siguiente estructura tarifaria:

- Cargos de instalación: Se cobrará una cuota única inicial al requerir la instalación de cada Línea Privada Nacional. Este cargo varía de acuerdo a la velocidad del enlace.
- Cargos de renta mensual: Se cobrará una renta mensual por cada Línea Privada Nacional contratada. El monto de la renta mensual se calcula con base en la velocidad del enlace y la distancia entre los puntos a conectar. La renta mensual se compone de dos elementos:
- Cargo fijo: Es un cargo fijo mensual que aplica para cualquier enlace, de acuerdo a la velocidad del mismo.
- Cargo por kilómetro: Es un cargo que se calcula según la distancia aérea entre los dos POPs de AXTEL. Renta mensual = Cargo fijo + (Cargo por km x Distancia en kms)
- Cargos por ampliación: Se realizará un cargo fijo y único al solicitar la ampliación (incremento) en la capacidad de una Línea Privada Nacional existente. La renta mensual se ajustará a la nueva capacidad del enlace.

Para las Líneas Privadas Internacionales, el cliente deberá contratar el Accesos Dedicado

Nodo Empresarial (Acceso Digital)
Es la plataforma de servicios de Acceso Dedicado que se encuentra en el sitio del cliente y que permite transmitir información digital entre el sitio del Cliente y la Red. Los servicios de Acceso Dedicado del Nodo Empresarial son necesarios para contar con servicios como Internet Dedicado y Líneas Privadas (Locales y Nacionales).
Nodo y Nodo Fibra Consisten en la infraestructura requerida en el sitio del cliente para proveer servicios de Acceso Dedicado. En el caso de tecnología de acceso Radio PMP, no se requerirá el Nodo. En caso de requerirse modificaciones y/o adecuaciones al sitio del Cliente con el fin de poder instalar el equipo, el cargo correrá a cuenta del Cliente.

nx64 Kbps (de 64 kbps a 1024 Kbps)
Accesos Dedicados con capacidades desde 64 Kbps hasta 1024 Kbps que permiten transmitir información entre el sitio del Cliente y un punto de presencia de la Red de AXTEL.

E1 Acceso Dedicados con una capacidad de transmisión digital de 2.048 Mbps entre el Sitio del Cliente y el punto de presencia de la Red.

E1 Fraccionado
Acceso Dedicado con una capacidad de transmisión digital de hasta 31 canales de 64 Kbps cada uno entre el Sitio del Cliente y el punto de presencia.

E3, T3 y STM-1
Acceso Dedicado con una alta capacidad de transmisión digital entre el Sitio del Cliente y el punto de presencia.
Este acceso se podrá ofrecer tanto en Clear Channel como Descanalizado.

Precios

Líneas Privadas AXTEL NACIONALES

Velocidad	Rango de Kilómetros	Instalación	Ampliación	Renta Mensual	
				Cargo Fijo	Cargo por kilómetro
64 kbps.	< 81 km. 81-161 km. 161-805 km. > 805 km	$2,950.40	$ 1,475.20	$451.80 $942.30 $1,760.40 $2,475.90	$11.70 $8.10 $2.70 $2.70
128 kbps.	< 81 km. 81-161 km. 161-805 km. > 805 km	$3,438.40	$1,719.20	$859.50 $1,791.00 $3,342.60 $4,704.30	$21.60 $15.30 $6.30 $4.50
192 / 256 kbps.	< 81 km. 81-161 km. 161-805 km. > 805 km	$4,298.40	$2,149.20	$1,946.70 $4,078.80 $7,554.60 $10,783.80	$45.90 $33.30 $12.60 $9.00
320 / 384 kbps.	< 81 km. 81-161 km. 161-805 km. > 805 km	$4,728.00	$2,364.00	$3,134.70 $6,574.50 $12,155.40 $17,410.50	$72.90 $54.00 $20.70 $14.40
448 / 512 / 576 kbps.	< 81 km. 81-161 km. 161-805 km. > 805 km	$5,158.40	$2,579.20	$4,321.80 $9,071.10 $16,755.30 $24,037.20	$99.90 $73.80 $27.90 $19.80
640 / 704/ 768 kbps.	< 81 km. 81-161 km. 161-805 km. > 805 km	$5,588.00	$2,794.00	$5,509.80 $11,566.80 $21,356.10 $30,663.90	$126.00 $93.60 $36.00 $26.10
1,024 kbps.	< 81 km. 81-161 km. 161-805 km. > 805 km	$6,017.60	$3,008.80	$6,697.80 $14,063.40 $25,956.90 $37,290.60	$153.00 $113.40 $43.20 $31.50
2,048 kbps.	< 81 km. 81-161 km. 161-805 km. > 805 km	$9,834.40	$4,917.20	$8,924.40 $18,747.00 $35,237.70 $49,705.20	$203.40 $151.20 $57.60 $41.40
E3	< 81 km. 81-161 km. 161-805 km. > 805 km	$18,000.00	$9,000.00	$68,420 $124,980 $234,918 $331,368	$1,559 $1,008 $384 $276
E3 Descanalizado	< 81 km. 81-161 km. 161-805 km. > 805 km	$18,000.00	$9,000.00	$114,232 $239,961 $451,042 $636,226	$2,603 $1,935 $737 $529
T3	< 81 km. 81-161 km. 161-805 km. > 805 km	$18,000.00	$9,000.00	$88,946 $162,474 $305,393 $430,778	$2,027 $1,310 $499 $358

Líneas Privadas AXTEL NACIONALES

Velocidad	Rango de Kilómetros	Instalación	Ampliación	Renta Mensual	
				Cargo Fijo	Cargo por kilómetro
T3 Descanalizado	< 81 km. 81-161 km. 161-805 km. > 805 km	$18,000.00	$9,000.00	$149,929 $314,949 $591,993 $835,047	$3,417 $2,540 $967 $695

Figura 7.3.2 Axtel precios

ALESTRA

Empresa 100% mexicana, enfocada, en sus inicios, al mercado de telecomunicaciones, y que desde 2007, ha evolucionado integrando e impulsando las Tecnologías de Información en México, convirtiéndose actualmente en un líder en la industria que ha provocado un cambio de paradigma.
Actualmente, con una inversión anual superior a 1,000 millones de pesos y una red de última generación, provee las más innovadoras Tecnologías de la Información y servicios administrados de comunicación a todo el sector empresarial del país; corporativos nacionales, empresas multinacionales y clientes institucionales, a través de más de 17 mil km de fibra. Su portafolio incluye Centro de Datos, al igual que soluciones de valor agregado como Aplicaciones en la Nube y Seguridad, así como Verticales para los sectores Salud, Educación, Gobierno y Finanzas, además de consultoría para diseño y administración de soluciones complejas que integran las Telecomunicaciones y la Informática (TI) y que se proveen con los más altos estándares de calidad mundial.
Características

- 6,700 kms. de fibra óptica, la cual incluye 1,750 kms. de anillos metropolitanos
- Cubre 198 ciudades de México a través de distintos esquemas
- Más de 3,000 acuerdos con autopistas, vías ferrocarrileras y compañías utilitarias
- 40 Gb/s de tecnología DWDM (Dense Wavelength Division Multiplexer)
- Anillos Interestatales de Larga Distancia SDH (Synchronous Digital Hierarchy) / DWDM 30 Puntos de Presencia
- Interconexión con 199 ciudades a lo largo de la República Mexicana
- 5 enlaces fronterizos

Soluciones de VPN
Ofrece redes privadas virtuales basadas en protocolo IP. Solución de comunicación robusta y versátil que permite establecer una conexión entre las instalaciones de la empresa y sus socios de negocio, ubicados en la misma o en diferentes ciudades, para transmitir mensajes de voz, datos y video.

Ethernet
Los servicios Ethernet están diseñados para proveer conectividad de red avanzada entre sitios en modalidades que pueden ser punto a punto o multipunto en un completo y variado rango de velocidades tanto en el ámbito metropolitano, nacional e internacional.

Tarifas Enlaces Locales Especiales

Capacidad Kbps	Instalación Sitio Nuevo	Ampliación	Renta Mensual
64	10,400	8,000	907
128	12,393	16,000	1,500
192	15,000	24,000	2,000
256	18,413	30,000	2,500
384	60,000	30,000	2,650
512	60,000	30,000	2,950
1024	62,000	30,000	3,800
1544	62,000	30,000	4,375
2048	74,799	30,000	4,375
34 Mbps	350,000	100,000	65,000
45 Mbps	350,000	100,000	65,000
155 Mbps	1,050,000	1,050,000	150,000

Figura 7.3.3 Alestra precios

(ii) Anexo Índice alfabético de CSS

At-rules

- @charset
- @font-face
- @import
- @media
- @keyframes
- @-ms-viewport
- @namespace
- @page

Functions

- attr()
- calc()
- counter()
- linear-gradient()
- radial-gradient()
- repeating-linear-gradient()
- repeating-radial-gradient()
- url()

Media

- aspect-ratio
- color
- color-index
- device-aspect-ratio
- device-height
- -webkit-device-pixel-ratio
- device-width
- height
- -ms-high-contrast
- monochrome
- orientation
- resolution
- -ms-view-state
- width

Propiedades CSS

Propiedad CSS	Propiedades para Script
-ms-accelerator	msAccelerator
align-content	alignContent
align-items	alignItems
align-self	alignSelf
animation	animation
animation-delay	animationDelay
animation-direction	animationDirection
animation-duration	animationDuration
animation-fill-mode	animationFillMode
animation-iteration-count	animationIterationCount
animation-name	animationName
animation-play-state	animationPlayState
animation-timing-function	animationTimingFunction
-webkit-appearance	webkitAppearance
backface-visibility	backfaceVisibility
background	background
background-attachment	backgroundAttachment
background-clip	backgroundClip
background-color	backgroundColor
background-image	backgroundImage
background-origin	backgroundOrigin
background-position	backgroundPosition
-ms-background-position-x	backgroundPositionX
-ms-background-position-y	backgroundPositionY
background-repeat	backgroundRepeat
background-size	backgroundSize
-ms-behavior	behavior
-ms-block-progression	msBlockProgression
border	border
border-bottom	borderBottom
border-bottom-color	borderBottomColor
border-bottom-left-radius	borderBottomLeftRadius
border-bottom-right-radius	borderBottomRightRadius
border-bottom-style	borderBottomStyle
border-bottom-width	borderBottomWidth
border-collapse	borderCollapse
border-color	borderColor
border-image	borderImage
border-image-outset	borderImageOutset
border-image-repeat	borderImageRepeat

border-image-slice	borderImageSlice
border-image-source	borderImageSource
border-image-width	borderImageWidth
border-left	borderLeft
border-left-color	borderLeftColor
border-left-style	borderLeftStyle
border-left-width	borderLeftWidth
border-radius	borderRadius
border-right	borderRight
border-right-color	borderRightColor
border-right-style	borderRightStyle
border-right-width	borderRightWidth
border-spacing	borderSpacing
border-style	borderStyle
border-top	borderTop
border-top-color	borderTopColor
border-top-left-radius	borderTopLeftRadius
border-top-right-radius	borderTopRightRadius
border-top-style	borderTopStyle
border-top-width	borderTopWidth
border-width	borderWidth
bottom	bottom
box-shadow	boxShadow
box-sizing	boxSizing
break-after	breakAfter
break-before	breakBefore
break-inside	breakInside
caption-side	captionSide
clear	clear
clip	clip
color	color
column-count	columnCount
column-fill	columnFill
column-gap	columnGap
column-rule	columnRule
column-rule-color	columnRuleColor
column-rule-style	columnRuleStyle
column-rule-width	columnRuleWidth
columns	columns
column-span	columnSpan
column-width	columnWidth
content	content
-ms-content-zoom-chaining	msContentZoomChaining

-ms-content-zooming	msContentZooming
-ms-content-zoom-limit	msContentZoomLimit
-ms-content-zoom-limit-max	msContentZoomLimitMax
-ms-content-zoom-limit-min	msContentZoomLimitMin
-ms-content-zoom-snap	msContentZoomSnap
-ms-content-zoom-snap-points	msContentZoomSnapPoints
-ms-content-zoom-snap-type	msContentZoomSnapType
counter-increment	counterIncrement
counter-reset	counterReset
cursor	cursor
direction	direction
display	display
empty-cells	emptyCells
-ms-filter†	filter†
flex	flex
flex-basis	flexBasis
flex-direction	flexDirection
flex-flow	flexFlow
flex-grow	flexGrow
flex-shrink	flexShrink
flex-wrap	flexWrap
float	styleFloat
-ms-flow-from	msFlowFrom
-ms-flow-into	msFlowInto
font	font
font-family	fontFamily
font-feature-settings	fontFeatureSettings
font-size	fontSize
font-stretch	fontStretch
font-style	fontStyle
font-variant	fontVariant
font-weight	fontWeight
-ms-grid-column	msGridColumn
-ms-grid-column-align	msGridColumnAlign
-ms-grid-columns	msGridColumns
-ms-grid-column-span	msGridColumnSpan
-ms-grid-row	msGridRow
-ms-grid-row-align	msGridRowAlign
-ms-grid-rows	msGridRows
-ms-grid-row-span	msGridRowSpan
height	height
-ms-high-contrast-adjust	msHighContrastAdjust
-ms-hyphenate-limit-chars	msHyphenateLimitChars

-ms-hyphenate-limit-lines	msHyphenateLimitLines
-ms-hyphenate-limit-zone	msHyphenateLimitZone
-ms-hyphens	msHyphens
-ms-ime-mode	imeMode
-ms-interpolation-mode†	msInterpolationMode†
justify-content	justifyContent
-ms-layout-flow†	layoutFlow†
-ms-layout-grid	layoutGrid
-ms-layout-grid-char	layoutGridChar
-ms-layout-grid-line	layoutGridLine
-ms-layout-grid-mode	layoutGridMode
-ms-layout-grid-type	layoutGridType
left	left
letter-spacing	letterSpacing
line-break†	lineBreak†
line-height	lineHeight
list-style	listStyle
list-style-image	listStyleImage
list-style-position	listStylePosition
list-style-type	listStyleType
margin	margin
margin-bottom	marginBottom
margin-left	marginLeft
margin-right	marginRight
margin-top	marginTop
max-height	maxHeight
max-width	maxWidth
min-height	minHeight
min-width	minWidth
opacity	opacity
order	order
orphans	orphans
outline	outline
outline-color	outlineColor
outline-style	outlineStyle
outline-width	outlineWidth
overflow	overflow
-ms-overflow-style	msOverflowStyle
-ms-overflow-x	overflowX
-ms-overflow-y	overflowY
padding	padding
padding-bottom	paddingBottom
padding-left	paddingLeft

padding-right	paddingRight
padding-top	paddingTop
page-break-after	pageBreakAfter
page-break-before	pageBreakBefore
page-break-inside	pageBreakInside
perspective	perspective
perspective-origin	perspectiveOrigin
position	position
-ms-progress-appearance	msProgressAppearance
quotes	quotes
right	right
ruby-align	rubyAlign
ruby-overhang	rubyOverhang
ruby-position	rubyPosition
-ms-scrollbar-3dlight-color	scrollbar3dLightColor
scrollbar-arrow-color	scrollbarArrowColor
scrollbar-base-color	scrollbarBaseColor
-ms-scrollbar-darkshadow-color	scrollbarDarkShadowColor
scrollbar-face-color	scrollbarFaceColor
-ms-scrollbar-highlight-color	scrollbarHighlightColor
-ms-scrollbar-shadow-color	scrollbarShadowColor
-ms-scrollbar-track-color	scrollbarTrackColor
-ms-scroll-chaining	msScrollChaining
-ms-scroll-limit	msScrollLimit
-ms-scroll-limit-x-max	msScrollLimitXMax
-ms-scroll-limit-x-min	msScrollLimitXMin
-ms-scroll-limit-y-max	msScrollLimitYMax
-ms-scroll-limit-y-min	msScrollLimitYMin
-ms-scroll-rails	msScrollRails
-ms-scroll-snap-points-x	msScrollSnapPointsX
-ms-scroll-snap-points-y	msScrollSnapPointsY
-ms-scroll-snap-type	msScrollSnapType
-ms-scroll-snap-x	msScrollSnapX
-ms-scroll-snap-y	msScrollSnapY
-ms-scroll-translation	msScrollTranslation
table-layout	tableLayout
text-align	textAlign
-ms-text-align-last	textAlignLast
-ms-text-autospace	textAutospace
-ms-text-combine-horizontal	msTextCombineHorizontal
text-decoration	textDecoration
text-indent	textIndent
-ms-text-justify	textJustify

-ms-text-kashida-space	textKashidaSpace
text-shadow	textShadow
-ms-text-overflow	textOverflow
-ms-text-size-adjust	msTextSizeAdjust
text-transform	textTransform
-ms-text-underline-position	textUnderlinePosition
top	top
touch-action	touchAction
transform	transform
transform-origin	transformOrigin
transform-style	transformStyle
transition	transition
transition-delay	transitionDelay
transition-duration	transitionDuration
transition-property	transitionProperty
transition-timing-function	transitionTimingFunction
unicode-bidi	unicodeBidi
-ms-user-select	msUserSelect
vertical-align	verticalAlign
visibility	visibility
white-space	whiteSpace
widows	widows
width	width
word-break	wordBreak
word-spacing	wordSpacing
-ms-word-wrap	wordWrap
-ms-wrap-flow	msWrapFlow
-ms-wrap-margin	msWrapMargin
-ms-wrap-through	msWrapThrough
-ms-writing-mode	writingMode
z-index	zIndex
-ms-zoom	zoom

Selectores

Elementos de Selectores

- Class
- ID
- Namespaced
- Type
- Universal

Attribute Selectors

- Equality **[=]**
- Existence **[]**
- Hyphen **[|=]**
- Whitespace **[~=]**
- Prefix **[^=]**
- Substring **[*=]**
- Suffix **[$=]**

Combinators

- Adjacent Sibling **(+)**
- Child **(>)**
- Descendant
- General Sibling **(~)**

Pseudo-classes

- :active
- :checked
- :disabled
- :empty
- :enabled
- :first (@page)
- :first-child
- :first-of-type
- :focus
- :hover
- :indeterminate
- :invalid
- :lang()
- :last-child
- :last-of-type
- :left (@page)
- :link
- :-ms-input-placeholder
- :-ms-keyboard-active
- :not(s)
- :nth-child(n)
- :nth-last-child(n)
- :nth-last-of-type(n)
- :nth-of-type(n)
- :only-child
- :only-of-type
- :optional
- :required
- :right (@page)
- :root
- :target
- :valid
- :visited

Pseudo-elements

- ::after
- ::before
- ::first-letter
- ::first-line
- ::-ms-check
- ::-ms-clear
- ::-ms-expand
- ::-ms-fill
- ::-ms-fill-lower
- ::-ms-fill-upper
- ::-ms-reveal
- ::-ms-thumb
- ::-ms-ticks-after
- ::-ms-ticks-before
- ::-ms-tooltip
- ::-ms-track
- ::-ms-value
- ::selection

(iii) Anexo Gramática de Visual Basic NET

Elementos	Gramática
Gramática léxica	*Start* ::= [*LogicalLine*+] *LogicalLine* ::= [*LogicalLineElement*+] [*Comment*] *LineTerminator* *LogicalLineElement* ::= *WhiteSpace* | *Token* *Token* ::= *Identifier* | *Keyword* | *Literal* | *Separator* | *Operador*
Caracteres y líneas	*Character* ::= < any Unicode character except a *LineTerminator* >
Terminadores de línea	LineTerminator ::= < Unicode carriage return character (0x000D) > | < Unicode line feed character (0x000A) > | < Unicode carriage return character > < Unicode line feed character > | < Unicode line separator character (0x2028) > | < Unicode paragraph separator character (0x2029) >
Continuación de línea	LineContinuation ::= WhiteSpace _ [WhiteSpace+] LineTerminator
Espacio en blanco	*WhiteSpace* ::= < Unicode blank characters (class Zs) > | *LineContinuation*
Comentarios	*Comment* ::= *CommentMarker* [*Character*+] *CommentMarker* ::= ' | **REM**
Identificadores	Identifier ::= NonEscapedIdentifier [TypeCharacter] | EscapedIdentifier NonEscapedIdentifier ::= < IdentifierName but not Keyword > EscapedIdentifier ::= [IdentifierName] IdentifierName ::= IdentifierStart [IdentifierCharacter+] IdentifierStart ::= AlphaCharacter | UnderscoreCharacter IdentifierCharacter IdentifierCharacter ::= UnderscoreCharacter | AlphaCharacter | NumericCharacter | CombiningCharacter | FormattingCharacter AlphaCharacter ::= < Unicode alphabetic character (classes Lu, Ll, Lt, Lm, Lo, Nl) > NumericCharacter ::= < Unicode decimal digit character (class Nd) > CombiningCharacter ::= < Unicode combining character (classes Mn, Mc) > FormattingCharacter ::= < Unicode formatting character (class Cf) > UnderscoreCharacter ::= < Unicode connection character (class Pc) > IdentifierOrKeyword ::= Identifier | Palabra clave
Caracteres de	TypeCharacter ::=

tipo	IntegerTypeCharacter | LongTypeCharacter | DecimalTypeCharacter | SingleTypeCharacter | DoubleTypeCharacter | StringTypeCharacter IntegerTypeCharacter ::= % LongTypeCharacter ::= & DecimalTypeCharacter ::= @ SingleTypeCharacter ::= ! DoubleTypeCharacter ::= # StringTypeCharacter ::= $
Palabras clave	Keyword ::= < member of keyword table at Keywords >
Literales	*Literal* ::= *BooleanLiteral* | *NumericLiteral* | *StringLiteral* | *CharacterLiteral* | *DateLiteral* | *Nothing* *NumericLiteral* ::= *IntegerLiteral* | *FloatingPointLiteral*
Literales booleanos	*BooleanLiteral* ::= **True** | **False**
Literales enteros	IntegerLiteral ::= IntegralLiteralValue [IntegralTypeCharacter] IntegralLiteralValue ::= IntLiteral | HexLiteral | OctalLiteral IntegralTypeCharacter ::= ShortCharacter | IntegerCharacter | LongCharacter | IntegerTypeCharacter | LongTypeCharacter ShortCharacter ::= S IntegerCharacter ::= I LongCharacter ::= L IntLiteral ::= Digit+ HexLiteral ::= & H HexDigit+ OctalLiteral ::= & O OctalDigit+ Digit ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 HexDigit ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F OctalDigit ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7
Literales de punto flotante	FloatingPointLiteral ::= FloatingPointLiteralValue [FloatingPointTypeCharacter] | IntLiteral FloatingPointTypeCharacter FloatingPointTypeCharacter ::= SingleCharacter | DoubleCharacter |

	DecimalCharacter | SingleTypeCharacter | DoubleTypeCharacter | DecimalTypeCharacter SingleCharacter ::= F DoubleCharacter ::= R DecimalCharacter ::= D FloatingPointLiteralValue ::= IntLiteral . IntLiteral [Exponent] | IntLiteral [Exponent] | IntLiteral Exponent Exponent ::= E [Sign] IntLiteral Sign ::= + | -
Literales de cadena	*StringLiteral* ::= **"** [*StringCharacter*+] **"** *StringCharacter* ::= < *Character* except for **"** > | **""**
Literales de carácter	*CharacterLiteral* ::= **"** *StringCharacter* **" C**
Literales de fecha	*DateLiteral* ::= **#** [*Whitespace*+] [*DateValue*] [*Whitespace*+] [*TimeValue*] [*Whitespace*+] **#** *DateValue* ::= *MonthValue DateSeparator DayValue DateSeparator YearValue* *DateSeparator* ::= **/** | **-** *TimeValue* ::= *HourValue* [**:** *MinuteValue*] [**:** *SecondValue*] [*Whitespace*+] [*AMPM*] *MonthValue* ::= *IntLiteral* *DayValue* ::= *IntLiteral* *YearValue* ::= *IntLiteral* *HourValue* ::= *IntLiteral* *MinuteValue* ::= *IntLiteral* *SecondValue* ::= *IntLiteral* *AMPM* ::= **AM** | **PM**
Nothing	*Nothing* ::= **Nothing**
Separadores	*Separator* ::= **(** | **)** | **!** | **#** | **,** | **.** | **:**
Caracteres de operador	*Operator* ::= **&** | ***** | **+** | **-** | **/** | **** | **^** | **<** | **=** | **>**
	Directivas de pre procesamiento
Directivas de compilación condicional	*Source* ::= [*ConditionalElement*+] *ConditionalElement* ::= *ConditionalIfGroup* | *ConditionalConstantDeclaration* | *LogicalLine*
Constantes condicionales	*ConditionalConstantDeclaration* ::= **#** **Const** *IdentifierOrKeyword* **=** *ConstantExpression LineTerminator*
Instrucciones de compilación condicional	ConditionalElseIfGroup ::= # ElseIf ConstantExpression [Then] LineTerminator [ConditionalElement+]
Directivas de	Source ::= [ExternalSourceElement+]

origen externo	ExternalSourceElement ::= ExternalSourceGroup | LogicalLine ExternalSourceGroup ::= # ExternalSource (StringLiteral , IntLiteral) LineTerminator [LogicalLine+] # End ExternalSource LineTerminator
Directivas de región	Start ::= [RegionElement+] RegionElement ::= RegionGroup | LogicalLine RegionGroup ::= # Region [NoDebug] StringLiteral LineTerminator [LogicalLine+] # End Region LineTerminator ConditionalElseGroup ::= # Else LineTerminator [ConditionalElement+]
Conceptos generales	Conceptos generales
Accesibilidad	*AccessModifier ::=* **Public** | **Protected** | **Friend** | **Private**
Nombres de tipos y espacios de nombres	QualifiedIdentifier ::= Identifier | QualifiedIdentifier . IdentifierOrKeyword
Bloques de atributos	Attributes ::= < AttributeList > AttributeList ::= Attribute | AttributeList , Attribute Attribute ::= [AttributeModifier :] TypeName [([AttributeArguments])] AttributeModifier ::= Assembly | Módulo
Argumentos de atributos	AttributeArguments ::= AttributePositionalArgumentList | AttributePositionalArgumentList , VariablePropertyInitializerList | VariablePropertyInitializerList AttributePositionalArgumentList ::= ConstantExpression | AttributePositionalArgumentList , ConstantExpression VariablePropertyInitializerList ::= VariablePropertyInitializer VariablePropertyInitializerList , VariablePropertyInitializer VariablePropertyInitializer ::= Identifier : = ConstantExpression
Archivos de código fuente y espacios de nombres	Source ::= [OptionDirective+] [ImportsDirective+] [Attributes] [NamespaceBody]
Opciones de	*OptionDirective* ::= *OptionExplicitDirective* | *OptionStrictDirective* |

compilación	*OptionCompareDirective*
Instrucción Option Explicit	*OptionExplicitDirective* ::= **Option Explicit** [*OnOff*] *LineTerminator* *OnOff* ::= **On** | **Desactivado**
Instrucción Option Strict	*OptionStrictDirective* ::= **Option Strict** [*OnOff*] *LineTerminator*
Instrucción Option Compare	*OptionCompareDirective* ::= **Option Compare** *CompareOption LineTerminator* *CompareOption* ::= **Binary** | **Text**
Instrucción Imports	ImportsDirective ::= Imports ImportsClauses LineTerminator ImportsClauses ::= ImportsClause | ImportsClauses , ImportsClause ImportsClause ::= ImportsAliasClause | RegularImportsClause
Alias de importación	*ImportsAliasClause* ::= *Identifier* = *QualifiedIdentifier*
Importaciones regulares	*ImportsNamespaceClause* ::= *QualifiedIdentifier*
Declaraciones de espacios de nombres	NamespaceDeclaration ::= Namespace QualifiedIdentifier LineTerminator [NamespaceMemberDeclaration+] End Namespace LineTerminator
Miembros de espacios de nombres	NamespaceMemberDeclaration ::= NamespaceDeclaration | TypeDeclaration TypeDeclaration ::= ModuleDeclaration | NonModuleDeclaration NonModuleDeclaration ::= EnumDeclaration | StructureDeclaration | InterfaceDeclaration | ClassDeclaration | DelegateDeclaration
Tipos	*TypeName* ::= *QualifiedIdentifier* | **Object** | *PrimitiveTypeName* | *ClassTypeName* | *ArrayTypeName*
Tipos primitivos	*PrimitiveTypeName* ::= *NumericTypeName* | **Boolean** | **Date** | **Tipo Char** *NumericTypeName* ::= *IntegralTypeName* | *FloatingPointTypeName* | **Tipo Decimal** *IntegralTypeName* ::= **Byte** | **Short** | **Integer** | **Tipo Long** *FloatingPointTypeName* ::= **Single** | **Double**

Enumeraciones	EnumDeclaration ::= [Attributes] [EnumModifier+] Enum Identifier [As IntegralTypeName] LineTerminator EnumMemberDeclaration+ End Enum LineTerminator EnumModifier ::= AccessModifier | Shadows
Miembros de enumeraciones	*EnumMemberDeclaration* ::= [*Attributes*] *Identifier* [= *ConstantExpression*] *LineTerminator*
Estructuras	StructDeclaration ::= [Attributes] [StructModifier+] Structure Identifier LineTerminator [TypeImplementsClause+] [StructMemberDeclaration+] End Structure LineTerminator StructureModifier ::= AccessModifier | Shadows
Implementaciones de interfaces de estructuras	TypeImplementsClause ::= Implements Implements LineTerminator Implements ::= TypeName | Implements , TypeName
Miembros de estructura	StructMemberDeclaration ::= NonModuleDeclaration VariableMemberDeclaration | ConstantMemberDeclaration | EventMemberDeclaration | MethodMemberDeclaration | PropertyMemberDeclaration | SharedConstructorDeclaration
Clases	ClassTypeName ::= String ClassDeclaration ::= [Attributes] [ClassModifier+] Class Identifier LineTerminator [ClassBase] [TypeImplementsClause+] [ClassMemberDeclaration+] End Class LineTerminator
Modificadores de clase	*ClassModifier* ::= *AccessModifier* | **Shadows** | **MustInherit** | **NotInheritable**
Especificación base de clase	*ClassBase ::=* **Inherits** *TypeName LineTerminator*
Miembros de clase	ClassMemberDeclaration ::= NonModuleDeclaration | EventMemberDeclaration |

	VariableMemberDeclaration | ConstantMemberDeclaration | MethodMemberDeclaration | PropertyMemberDeclaration | ConstructorMemberDeclaration
Módulos estándar	ModuleDeclaration ::= [Attributes] [AccessModifier+] Module Identifier LineTerminator [ModuleMemberDeclaration+] End Module LineTerminator
Miembros de módulos estándar	ModuleMemberDeclaration ::= NonModuleDeclaration | VariableMemberDeclaration | ConstantMemberDeclaration | EventMemberDeclaration | MethodMemberDeclaration | PropertyMemberDeclaration | SharedConstructorDeclaration
Interfaces	InterfaceDeclaration ::= [Attributes] [InterfaceModifier+] Interface Identifier LineTerminator [InterfaceBases+] [InterfaceMemberDeclaration+] End Interface LineTerminator InterfaceModifier ::= AccessModifier | Shadows
Herencia de interfaz	InterfaceBase ::= Inherits InterfaceBases LineTerminator InterfaceBases ::= TypeName InterfaceBases , TypeName
Miembros de interfaces	InterfaceMemberDeclaration ::= NonModuleDeclaration | EventMemberDeclaration | MethodMemberDeclaration | PropertyMemberDeclaration
Matrices	ArrayTypeName ::= TypeName ArrayTypeModifier ArrayTypeModifier ::= ([RankList]) RankList ::= , | RankList , ArrayNameModifier ::= ArrayTypeModifier | ArrayInitializationModifier
Delegados	*DelegateTypeDeclaration* ::= [*DelegateModifier*+] **Delegate** *MethodDeclaration* *DelegateModifier* ::= *AccessModifier* | **Shadows**
Miembros de tipo	Miembros de tipo

Métodos	*MethodMemberDeclaration* ::= *MethodDeclaration* | *ExternalMethodDeclaration*
Declaraciones de métodos regulares	MethodDeclaration ::= SubDeclaration | FunctionDeclaration SubDeclaration ::= [Attributes] [ProcedureModifier+] Sub Identifier [([FormalParameterList])] [HandlesOrImplements] LineTerminator [Block] [End Sub LineTerminator] FunctionDeclaration ::= [Attributes] [ProcedureModifier+] Function Identifier [([FormalParameterList])] [As [Attributes] TypeName] [HandlesOrImplements] LineTerminator [Block] [End Function LineTerminator] ProcedureModifier ::= AccessModifier | Shadows | Shared | Overridable | NotOverridable | MustOverride | Overrides | Overloads HandlesOrImplements ::= HandlesClause | MethodImplementsClause
Declaraciones de métodos regulares	ExternalMethodDeclaration ::= ExternalSubDeclaration | ExternalFunctionDeclaration ExternalSubDeclaration ::= [Attributes] [ExternalMethodModifier+] Declare [CharsetModifier] Sub Identifier LibraryClause [AliasClause] [([FormalParameterList])] LineTerminator ExternalFunctionDeclaration ::= [Attributes] [ExternalMethodModifier+] Declare [CharsetModifier] Function Identifier LibraryClause [AliasClause] [([FormalParameterList])] [As [Attributes] TypeName] LineTerminator ExternalMethodModifier ::= AccessModifier | Shadows | Overloads CharsetModifier ::= Ansi | Unicode | Auto LibraryClause ::= Lib StringLiteral AliasClause ::= Alias StringLiteral
Parámetros de métodos	FormalParameterList ::= FormalParameter | FormalParameterList , FormalParameter FormalParameter ::= [Attributes] ParameterModifier+ Identifier [As TypeName] [= ConstantExpression]

	ParameterModifier ::= ByRef | ByVal | Optional | ParamArray
Implementaciones de métodos de interfaz	ImplementsClause ::= [Implements ImplementsList] ImplementsList ::= InterfaceMemberSpecifier | ImplementsList , InterfaceMemberSpecifier InterfaceMemberSpecifier ::= TypeName . Identificador
Control de eventos	HandlesClause ::= [Handles EventHandlesList] EventHandlesList ::= EventMemberSpecifier | EventHandlesList , EventMemberSpecifier EventMemberSpecifier ::= Identifier . [Identifier .] Identifier MyBase . [Identifier .] Identifier
Constructores	*ConstructorMemberDeclaration* ::= *InstanceConstructorDeclaration* | *SharedConstructorDeclaration*
Constructores de instancia	InstanceConstructorDeclaration ::= [Attributes] [InstanceConstructorModifier+] Sub New [([FormalParameterList])] LineTerminator [Block] End Sub LineTerminator InstanceConstructorModifier ::= AccessModifier | Overloads
Miembros compartidos	SharedConstructorDeclaration ::= [Attributes] Shared Sub New [( )] LineTerminator [Block] End Sub LineTerminator
Eventos	EventMemberDeclaration ::= [Attributes] [EventModifiers+] Event Identifier ParametersOrType [ImplementsClause] LineTerminator ParametersOrType ::= [([FormalParameterList])] | As TypeName
Constantes	ConstantMemberDeclaration ::= [Attributes] [ConstantModifier+] Const Identifier [As TypeName] = ConstantExpression LineTerminator ConstantModifier ::= AccessModifiers | Shadows
Variables	VariableMemberDeclaration ::= [Attributes] [VariableModifier+] [Dim] VariableDeclarators LineTerminator VariableModifier ::= AccessModifiers | Shadows | Shared | ReadOnly | WithEvents VariableDeclarators ::= VariableDeclarator | VariableDeclarators , VariableDeclarator

	VariableDeclarator ::= VariableIdentifiers [As TypeName] | VariableIdentifier [As [New] TypeName [(ArgumentList)]] [= VariableInitializer] VariableIdentifiers ::= VariableIdentifier | VariableIdentifiers , VariableIdentifier
Inicializadores de variables	*VariableInitializer* ::= *RegularInitializer* | *ArrayElementInitializer*
Inicializadores regulares	*RegularInitializer* ::= *Expression*
Inicializadores de elementos de matriz	ArrayElementInitializer ::= { [VariableInitializerList] } VariableInitializerList ::= VariableInitializer | VariableInitializerList , VariableInitializer VariableInitializer ::= Expression | ArrayElementInitializer
Inicializadores de tamaño de matriz	ArrayInitializationModifier ::= (InitializationRankList) InitializationRankList ::= Expression | InitializationRankList , Expression
Propiedades	PropertyMemberDeclaration ::= [Attributes] PropertyModifier+ Property Identifier [([FormalParameterList])] [As TypeName] [MethodImplementsClause] LineTerminator PropertyAccessorDeclaration+ [End Property LineTerminator] PropertyModifier ::= ProcedureModifier | Default | ReadOnly | WriteOnly PropertyAccessorDeclaration ::= PropertyGetDeclaration | PropertySetDeclaration
Declaraciones de captadores	PropertyGetDeclaration ::= [Attributes] Get LineTerminator [Block] End Get
Declaraciones de establecedores	PropertySetDeclaration ::= [Attributes] Set LineTerminator [Block] End Set
Instrucciones	Instrucciones
Bloques	Block ::= [LabeledLine+] LabeledLine ::= [LabelName :] [Statements] LineTerminator LabelName ::= Identifier | IntLiteral Statements ::= [Statement] | Statements [Statement]

Instrucciones de declaración de variables locales	LocalDeclarationStatement ::= LocalModifier LocalDeclarator StatementTerminator LocalModifier ::= Dim | Const LocalDeclarator ::= LocalIdentifiers [As TypeName] Identifier [ArrayNameModifier] [As [New] TypeName [([ArgumentList])]] [= VariableInitializer] LocalIdentifiers ::= Identifier [ArrayNameModifier] | LocalIdentifiers , Identifier [ArrayNameModifier] LocalVariableName ::= Identifier
Instrucción With	WithStatement ::= With Expression StatementTerminator [Block] End With StatementTerminator
Instrucción SyncLock	SyncLockStatement ::= SyncLock Expression StatementTerminator [Block] End SyncLock StatementTerminator
Instrucciones de eventos	EventStatement ::= RaiseEventStatement | AddHandlerStatement | RemoveHandlerStatement
Instrucción RaiseEvent	*RaiseEventStatement* ::= **RaiseEvent** *EventMemberName* [**(** [*ArgumentList*] **)**] *StatementTerminator*
Instrucciones AddHandler y RemoveHandler	*AddHandlerStatement* ::= **AddHandler** *HandlerArguments StatementTerminator* *RemoveHandlerStatement* ::= **RemoveHandler** *HandlerArguments StatementTerminator* *HandlerArguments* ::= *EventExpression* , *ArgumentExpression*
Instrucciones de asignación	AssignmentStatement ::= SimpleAssignmentStatement | DelegateAssignmentStatement | CompoundAssignmentStatement | MidAssignmentStatement
Instrucciones de asignaciones simples	*RegularAssignmentStatement* ::= *VariableExpression* **=** *Expression StatementTerminator* *VariableExpression* ::= *Expression*
Instrucciones de asignaciones de delegado	*DelegateAssignmentStatement* ::= *VariableExpression* **= AddressOf** *InvocationTargetExpression StatementTerminator*
Instrucciones de asignaciones compuestas	*CompoundAssignmentStatement* ::= *VariableExpression* *CompoundBinaryOperator* **=** *Expression StatementTerminator* *CompoundBinaryOperator* ::= ^ | ***** | / | \ | **+** | **-** | **&**
Instrucción de asignación Mid	*MidAssignmentStatement* ::= **Mid** [**$**] **(** *VariableExpression* , *Expression* [, *Expression*] **) =** *Expression StatementTerminator*

Instrucción de invocación	*InvocationStatement* ::= [**Call**] *InvocationExpression StatementTerminator*
Instrucciones condicionales	*ConditionalStatement* ::= *IfStatement* | *SelectStatement*
If...Then...Else (Instrucciones)	IfStatement ::= BlockIfStatement | LineIfThenStatement BlockIfStatement ::= If BooleanExpression [Then] StatementTerminator [Block] [ElseIfStatement+] [ElseStatement] End If StatementTerminator ElseIfStatement ::= ElseIf BooleanExpression [Then] StatementTerminator [Block] ElseStatement ::= Else StatementTerminator [Block] LineIfThenStatement ::= If BooleanExpression Then [Statements] [Else Statements] StatementTerminator BooleanExpression ::= Expression
Instrucciones Select...Case	SelectStatement ::= Select [Case] Expression StatementTerminator [CaseStatement+] [CaseElseStatement] End Select StatementTerminator CaseStatement ::= Case CaseClauses StatementTerminator [Block] CaseClauses ::= CaseClause | CaseClauses , CaseClause CaseClause ::= [Is] ComparisonOperator Expression | Expression [To Expression] ComparisonOperator ::= = | < > | < | > | = > | = < CaseElseStatement ::= Case Else StatementTerminator [Block]
Instrucciones de bucle	LoopStatement ::= WhileStatement | DoLoopStatement | ForStatement | ForEachStatement
Instrucciones While...End While y Do...Loop	WhileStatement ::= While BooleanExpression StatementTerminator [Block] End While StatementTerminator

	DoLoopStatement ::= Do [WhileOrUntil BooleanExpression] StatementTerminator [Block] Loop [WhileOrUntil BooleanExpression] StatementTerminator WhileOrUntil ::= While | Until
For...Next (Instrucciones)	ForStatement ::= For LoopControlVariable = Expression To Expression [Step Expression] StatementTerminator [Block] Next [NextExpressionList] StatementTerminator LoopControlVariable ::= Identifier As TypeName | Expression NextExpressionList ::= VariableExpression | NextExpressionList , VariableExpression
For Each...Next (Instrucciones)	ForEachStatement ::= For Each LoopControlVariable In Expression StatementTerminator [Block] Next [VariableExpression] StatementTerminator
Instrucciones de control de excepciones	ExceptionHandlingStatement ::= StructuredExceptionStatement | UnstructuredExceptionStatement
Instrucciones de control estructurado de excepciones	StructuredExceptionStatement ::= ThrowStatement | TryStatement ThrowStatement ::= Throw [Expression] StatementTerminator TryStatement ::= Try StatementTerminator [Block] [CatchStatement+] [FinallyStatement] End Try StatementTerminator
Bloques Finally	FinallyStatement ::= Finally StatementTerminator [Block]
Bloques Catch	CatchStatement ::= Catch [Identifier As TypeName] [When BooleanExpression] StatementTerminator [Block]
Instrucciones de control no estructurado de excepciones	UnstructuredExceptionStatement ::= ErrorStatement | OnErrorStatement | ResumeStatement
Instrucción Error	*ErrorStatement* ::= **Error** *Expression StatementTerminator*
Instrucción On	OnErrorStatement ::= On Error ErrorClause

Error	StatementTerminator ErrorClause ::= Resume Next | GoTo -1 | GoTo 0 | GotoStatement
Instrucción Resume	*ResumeStatement* ::= **Resume** [*ResumeClause*] *StatementTerminator* *ResumeClause* ::= **Next** | *LabelName*
Instrucciones de flujo de control	ControlFlowStatement ::= GotoStatement | ExitStatement | StopStatement | EndStatement | ReturnStatement GotoStatement ::= GoTo LabelName StatementTerminator ExitStatement ::= Exit ExitKind StatementTerminator ExitKind ::= Do | For | While | Select | Sub | Function | Property | Pruebe StopStatement ::= Stop StatementTerminator EndStatement ::= End StatementTerminator ReturnStatement ::= Return [Expression]
Instrucciones de control de matrices	ArrayHandlingStatement ::= RedimStatement | EraseStatement
Instrucción ReDim	RedimStatement ::= ReDim [Preserve] RedimClauses+ StatementTerminator RedimClauses ::= RedimClauses | RedimClause , RedimClauses RedimClause ::= VariableExpression ArrayInitializationModifier
Instrucción Erase	EraseStatement ::= Erase VariableExpressions StatementTerminator VariableExpressions ::= VariableExpression | VariableExpressions , VariableExpression
Expresiones	Expression ::= SimpleExpression | InvocationExpression | MemberAccessExpression | IndexExpression | NewExpression | CastExpression | TypeOfExpression | OperatorExpression
Expresiones	*ConstantExpression* ::= *Expression*

constantes	
Expresiones variables	*VariableExpression* ::= *Expression*
Expresiones de eventos	EventExpression ::= Expression . IdentifierOrKeyword | [MeExpression .] IdentifierOrKeyword | EventMemberName
Expresiones simples	SimpleExpression ::= LiteralExpression | ParenthesizedExpression | MeExpression | MetaTypeExpression | LocalVariableExpression | TypeOfIsOperatorExpression | IsOperatorExpression
Expresiones literales	*LiteralExpression* ::= *Literal*
Expresiones entre paréntesis	*ParenthesizedExpression* ::= **(** *Expression* **)**
Expresión Me	*MeExpression* ::= **Me**
Expresiones GetType	*MetaTypeExpression* ::= **GetType (** *TypeName* **)**
Expresiones de variable local	*LocalVariableExpression* ::= *LocalVariableName*
Expresiones TypeOf...Is	*TypeOfIsOperatorExpression* ::= **TypeOf** *Expression* **Is** *TypeName*
Expresiones Is	*IsOperatorExpression* ::= *Expression* **Is** *Expression*
Expresiones de invocación	InvocationExpression ::= InvocationTargetExpression [([ArgumentList])] InvocationTargetExpression ::= DelegateExpression | [[Expression] .] IdentifierOrKeyword | MyClass . IdentifierOrKeyword | MyBase . IdentifierOrKeyword | MethodMemberName DelegateExpression ::= Expression
Listas de argumentos	ArgumentList ::= PositionalArgumentList , NamedArgumentList | PositionalArgumentList | NamedArgumentList PositionalArgumentList ::= ArgumentExpression | PositionalArgumentList , [ArgumentExpression] NamedArgumentList ::= Identifier : = ArgumentExpression | NamedArgumentList , Identifier : = ArgumentExpression

	ArgumentExpression ::= Expression | DelegateArgumentExpression
Expresiones de argumentos de delegado	*DelegateArgumentExpression* ::= **AddressOf** *InvocationTargetExpression*
Expresiones de acceso a miembros	MemberAccessExpression ::= [[Expression] .] IdentifierOrKeyword | VariableMemberName | PropertyMemberName | ConstantMemberName | EnumMemberName | DictionaryAccessExpression
Acceso a miembros de tipo diccionario	*DictionaryAccessExpression* ::= [*Expression*] **!** *IdentifierOrKeyword*
Expresiones de índice	*IndexExpression* ::= *Expression* **(** *ArgumentList* **)**
Expresiones New	NewExpression ::= ObjectCreationExpression | ArrayCreationExpression | DelegateCreationExpression
Expresiones de creación de objetos	*ObjectCreationExpression* ::= **New** *TypeName* [**(** [*ArgumentList*] **)**]
Expresiones de creación de matrices	ArrayCreationExpression ::= New TypeName (ArgumentList) ArrayElementInitializer | New ArrayTypeName ArrayElementInitializer
Expresiones de creación de delegados	*DelegateCreationExpression* ::= **New** *TypeName* **(** *InvocationTargetExpression* **)**
Expresiones de conversión	CastExpression ::= CType (Expression , TypeName) | CastTarget (Expression) CastTarget ::= CBool | CByte | CChar | CDate | CDec | CDbl | CInt | CLng | CObj | CShort | CSng | CStr
Operadores	OperatorExpression ::= UnaryOperatorExpression | BinaryOperatorExpression BinaryOperatorExpression ::= ArithmeticOperatorExpression | RelationalOperatorExpression | LikeOperatorExpression | ConcatenationOperatorExpression | ShortCircuitLogicalOperatorExpression | LogicalOperatorExpression | ShiftOperatorExpression
Operadores unarios	UnaryOperatorExpression ::=

	UnaryPlusExpression | UnaryMinusExpression | UnaryLogicalNotExpression
Operador unario más	*UnaryPlusExpression* ::= **+** *Expression*
Operador unario Menos	*UnaryMinusExpression* ::= **-** *Expression*
Operador de negación lógica	*UnaryLogicalNotExpression* ::= **Not** *Expression*
Operadores aritméticos	ArithmeticOperatorExpression ::= AdditionOperatorExpression | SubtractionOperatorExpression | MultiplicationOperatorExpression | DivisionOperatorExpression | ModuloOperatorExpression | ExponentOperatorExpression
Operador de suma	*AdditionOperatorExpression* ::= *Expression* **+** *Expression*
Operador de resta	*SubtractionOperatorExpression* ::= *Expression* **-** *Expression*
Operador de multiplicación	*MultiplicationOperatorExpression* ::= *Expression* ***** *Expression*
Operador de división	DivisionOperatorExpression ::= RegularDivisionOperatorExpression | IntegerDivisionOperatorExpression RegularDivisionOperatorExpression ::= Expression / Expression IntegerDivisionOperatorExpression ::= Expression \ Expression
Operador Mod	*ModOperatorExpression* ::= *Expression* **Mod** *Expression*
Operador de exponenciación	*ExponentOperatorExpression* ::= *Expression* ^ *Expression*
Operadores relacionales	RelationalOperatorExpression ::= Expression = Expression | Expression < > Expression | Expression < Expression | Expression > Expression | Expression < = Expression | Expression > = Expression
Operador Like	*LikeOperatorExpression* ::= *Expression* **Like** *Expression*
Operador de concatenación	*ConcatenationOperatorExpression* ::= *Expression* **&** *Expression*
Operadores lógicos	LogicalOperatorExpression ::= Expression And Expression | Expression Or Expression | Expression Xor Expression | Expression AndAlso Expression | Expression OrElse Expression

Operadores de desplazamiento	ShiftOperatorExpression ::= Expression << Expression | Expression >> Expression

(iv) Anexo HTML Etiquetas ordenadas Alfabéticamente

Elemento	Descripción
<!--...-->	Define un comentario
<!DOCTYPE>	Define el tipo de docuemento
<a>	Define un hyperlink
<abbr>	Define una abreviación o acronimo
<acronym>	No soportada en HTML5. Use <abbr> instead.Define an acronym
<address>	Define contact information for the author/owner of a document
<applet>	No soportada en HTML 5. Use <embed> or <object> instead.Define an embedded applet
<area>	Define an area inside an image-map
<article>	Define an article
<aside>	Define content aside from the page content
<audio>	Define sound content
<b>	Define bold text
<base>	Specifies the base URL/target for all relative URLs in a document
<basefont>	No soportada en HTML 5. Use CSS instead.Specifies a default color, size, and font for all text in a document
<bdi>	Isolates a part of text that might be formatted in a different direction from other text outside it
<bdo>	Overrides the current text direction
<big>	No soportada en HTML 5. Use CSS instead.Define big text
<blockquote>	Define a section that is quoted from another source
<body>	Define the document's body
 	Define a single line break
<button>	Define a clickable button
<canvas>	Used to draw graphics, on the fly, via scripting (usually JavaScript)
<caption>	Define a table caption
	No soportada en HTML 5. Use CSS instead. Define centered text
<cite>	Define the title of a work
<code>	Define a piece of computer code
<col>	Specifies column properties for each column within a <colgroup>element
<colgroup>	Specifies a group of one or more columns in a table for formatting
<datalist>	Specifies a list of pre-defined options for input controls
<dd>	Define a description/value of a term in a description list
<del>	Define text that has been deleted from a document
<details>	Define additional details that the user can view or hide
<dfn>	Represents the defining instance of a term
<dialog>	Define a dialog box or window
<dir>	No soportada en HTML 5. Use instead.Define a directory list
	Define a section in a document
<dl>	Define a description list
<dt>	Define a term/name in a description list
	Define emphasized text
<embed>	Define a container for an external (non-HTML) application
<fieldset>	Groups related elements in a form
<figcaption>	Define a caption for a <figure>
<figure>	Specifies self-contained content
<font>	No soportada en HTML 5. Use CSS instead.Define font, color, and size for text

<footer>	Define a footer for a document or section
<form>	Define an HTML form for user input
<frame>	No soportada en HTML 5.Define un frame en un frameset
<frameset>	No soportada en HTML 5.Define un conjunto de frames
<h1> to <h6>	Define encabezado HTML
<head>	Define la información del documento
<header>	Define el encabezado
<hr>	Define un cambio temático del contenido
<html>	Define la raíz un HTML
<i>	Define una parte del texto en voz alternativa
<iframe>	Define un inline frame
<img>	Define una imagen
<input>	Define un control de entrada
<ins>	Define un texto que ha sido insertado
<kbd>	Define entrada de teclado
<keygen>	Define un campo generador key-pair
<label>	Define una etiqueta para <input>
<legend>	Define un caption para <fieldset>
	Define un elemento de lista
<link>	Define la relación entre un documento y una fuente externa
<main>	Especifica el contenido principal
<map>	Define un mapa de imagen en el cliente
<mark>	Define una texto resaltado
<menu>	Define una lista de comandos
<menuitem>	Define un elemento del menú
<meta>	Define el metadata del documento
<meter>	Define una medida en un rango dado
<nav>	Define los links de navegación
<noframes>	No soportada en HTML 5.Define un contenido alterno para clientes que no soportan frames
<noscript>	Define un contenido alterno para clientes que no soportan scripts
<object>	Define un objeto
	Define una lista ordenada
<optgroup>	Define un grupo de opciones en una lista drop-down
<option>	Define una opción en una lista drop-down
<output>	Define el resultado de un calculo
<p>	Define un párrafo
<param>	Define un parámetro para un objeto
<pre>	Define un texto pre formateado
<progress>	Representa el progreso de una accion
<q>	Define comillas cortas
<rp>	Define que mostrar si no se soporta anotaciones de ruby
<rt>	Define una explicación/pronunciación de caracteres Asiáticos
<ruby>	Define un texto ruby
<s>	Define un texto que ya no es correcto
<samp>	Define una salida ejemplo de un programa
<script>	Define un script de cliente
<section>	Define una sección
<select>	Define una lista drop-down
<small>	Define texto pequeño
<source>	Define múltiples medios para elementos media (<video> y <audio>

	Define una sección
<strike>	No soportada en HTML 5. Use <del> or <s> .Define texto sobresaltado
	Define texto importante
<style>	Define estilos en un documento
<sub>	Define texto subscrito
<summary>	Define un encabezado para un <details>
<sup>	Define texto superscript
<table>	Define una tabla
<tbody>	Define el contenido de una tabla
<td>	Define una celda en una tabla
<textarea>	Define una entrada multilinea (text area)
<tfoot>	Agrupa el contenido en el pie de tabla
<th>	Define el encabezado de una celda en una tabla
<thead>	Agrupa el contenido de un encabezado de tabla
<time>	Define la fecha/tiempo
<title>	Define un título para el documento
<tr>	Define un renglón en una tabla
<track>	Define el canal para texto en elementos media (<video> y <audio>)
<tt>	No soportada en HTML 5. Use CSS . Define texto teletipo
<u>	Define texto diferente al normal
	Define lista subrayada y no ordenada
<var>	Define una variable
<video>	Define un video o película
<wbr>	Define un posible salto de linea

(v) Anexo Desarrollo de software

Método/Herramienta	Descripción
Big Design Up Front (BDUF)	Se usa con el afán de crear modelos comprensibles de los requerimientos de un Sistema, del análisis de estos requerimientos, de una arquitectura que cumpla estos requerimientos y eventualmente de un diseño detallado .El aspecto importante de BDUF es su naturaleza serial y no aspectos de validación • Se usa comúnmente en XP . • BRUF y BDUF se usan juntos generalmente
Desarrollo ágil de software	Envuelve un enfoque para la toma de decisiones en los proyectos de software, que se refiere a métodos de ingeniería del software basados en el desarrollo iterativo e incremental, donde los requisitos y soluciones evolucionan con el tiempo según la necesidad del proyecto. Así el trabajo es realizado mediante la colaboración de equipos auto-organizados y multidisciplinarios, inmersos en un proceso compartido de toma de decisiones a corto plazo. Los métodos ágiles enfatizan las comunicaciones cara a cara en vez de la documentación. La mayoría de los equipos ágiles están localizados en una simple oficina abierta, a veces llamadas "plataformas de lanzamiento" (bullpen en inglés). La oficina debe incluir revisores, escritores de documentación y ayuda, diseñadores de iteración y directores de proyecto. Los métodos ágiles también enfatizan que el software funcional es la primera medida del progreso. Combinado con la preferencia por las comunicaciones cara a cara, generalmente los métodos ágiles son criticados y tratados como "indisciplinados" por la falta de documentación técnica.Algunos métodos ágiles de desarrollo de software • Adaptive Software Development (ASD) • Agile Unified Process • Crystal Clear • Feature Driven Development (FDD) • Lean Software Development (LSD) • Kanban (desarrollo) • Open Unified Process (OpenUP) • Programación Extrema (XP) • Método de desarrollo de sistemas dinámicos (DSDM) • Scrum • G300 • 6D-BUM
Desarrollo de diseño-dirigido (D3)	Es un proceso ágil para crear requerimientos innovadores para construir mejores soluciones. Trabaja de cerca con SCRUM y Extreme Programming (XP) para manejar e implementar estos requerimientos. También puede trabajar con procesos no-ágiles como RUP. Está basado en la siguiente filosofía: Diseñar es el arte de crear bellas, elegantes e innovadoras soluciones, que trabajen en el contexto del usuario y del cliente. Ningún proceso puede garantizar un diseño mejor; creando el entorno correcto y los grupos de personas es la única manera de traer innovación. El diseño es un accidente que patalea en la concepción, y D3 maximiza las oportunidades de que un accidente ocurra. D3 fue originalmente creado por Henry Jacob.
Desarrollo guiado por la funcionalidad (FDD)	**Feature-Driven Development (FDD)** Es un "framework" o una meta-metodología, que necesita ser adaptado al caso concreto. El padre de la metodología es Jeff De Luca. Y por ello la metodología ágil FDD tiene gran influencia de las ideas de Peter Coad, en su día gurú de la Orientación a Objetos, y hoy, en los tiempos de lo ágil, menos conocido. Quizás los antiguos del lugar recordarán en su día un famoso libro de Coad llamado "Java Modeling In Color With UML (1999)", en el que aunque el primer autor del libro era Coad… el libro también lo firmaba Jeff De Luca. El libro que mejor describe la metodología es el A Practical Guide to Feature-Driven Development y la web de referencia es la del autor: Jeff De Luca´s website. (i) Los procesos la metodología ágil FDD FDD es una metodología dirigida por modelos, y de iteraciones cortas. FDD define 5 procesos: Proceso 1 – Desarrollar el modelo global (Develop overall model), Proceso 2 – Construir una lista de características (Build feature list), Proceso 3 – Planificar (Plan by feature), Proceso 4 – Diseñar (Design by feature) y Proceso 5 – Construir (Build by feature). Los 3 primeros pueden considerarse la "iteración cero", aunque en FDD no le llaman así, y los consideran "procesos iníciales". Ya en el libro de gestión ágil hablábamos de que en la práctica los equipos de desarrollo necesitan un tiempo previo para arrancar, montar entornos, planificar, e incluso hacer un estudio previo de la arquitectura. Esto normalmente se suele hacer en lo que comúnmente se llama iteración cero (o sprint cero para quienes siguen Scrum). Pero, por ejemplo, en Scrum no se dice mucho de esta iteración cero, sin embargo FDD la detalla mucho más. Los dos primeros procesos son secuenciales y definen el modelo global. Los tres finales se iteran para cada "feature" (que vienen a ser los requisitos).

Desarrollo guiado por pruebas	Test-driven development (TDD) es una práctica de ingeniería de software que involucra otras dos prácticas: Escribir las pruebas primero (Test First Development) y Refactorización (Refactoring). Para escribir las pruebas generalmente se utilizan las pruebas unitarias (unit test en inglés). En primer lugar, se escribe una prueba y se verifica que las pruebas fallan. A continuación, se implementa el código que hace que la prueba pase satisfactoriamente y seguidamente se refactoriza el código escrito. El propósito del desarrollo guiado por pruebas es lograr un código limpio que funcione. La idea es que los requisitos sean traducidos a pruebas, de este modo, cuando las pruebas pasen se garantizará que el software cumple con los requisitos que se han establecido.
Desarrollo iterativo y creciente	Es un proceso de desarrollo de software creado en respuesta a las debilidades del modelo tradicional de cascada. Básicamente este modelo de desarrollo, que no es más que un conjunto de tareas agrupadas en pequeñas etapas repetitivas (iteraciones),1 es uno de los más utilizados en los últimos tiempos ya que, como se relaciona con novedosas estrategias de desarrollo de software y una programación extrema, es empleado en metodologías diversas. El modelo consta de diversas etapas de desarrollo en cada incremento, las cuales inician con el análisis y finalizan con la instauración y aprobación del sistema.
Diseño guiado por el dominio (DDD)	Es un enfoque para el desarrollo de software con necesidades complejas mediante una profunda conexión entre la implementación y los conceptos del modelo y núcleo del negocio. Las premisas del diseño guiado por el dominio son las siguientes: • Poner el foco primario del proyecto en el núcleo y la lógica del dominio. • Basar los diseños complejos en un modelo. • Iniciar una creativa colaboración entre técnicos y expertos del dominio para interactuar lo más cercano posible a los conceptos fundamentales del problema. El diseño guiado por el dominio no es una tecnología ni una metodología, este provee una estructura de prácticas y terminologías para tomar decisiones de diseño que enfoquen y aceleren el manejo de dominios complejos en los proyectos de software. El término fue acuñado por Eric Evans en su libro Domain-Driven Design - Tackling Complexity in the Heart of Software.
Filosofía UNIX	Todos los colabores de Unix se unieron a la creación, también aportaron valores a los principios filosóficos, que luego de mucho tiempo se aplicó a la forma de poder programar como en el lenguaje C, dando aspectos que podrían ayudar a programar en base a los principios filosóficos: • Regla de Modularidad: Escribe módulos o partes simples conectadas a través de interfaces los cuales comunican a otros programas. • Regla de Composición: Diseñar programas que se conectarán a otros programas, una salida del programa es la entrada de otro programa. • Imperio de la Robustez: La robustez es el hijo de la transparencia y la simplicidad. • Regla del Silencio: Cuando un programa no tiene nada sorprendente que decir, probablemente no tiene nada que decir. • Énfasis de la menor sorpresa: en lo que respecta en el diseño de la interfaz, no es necesario sorprender.
La catedral y el bazar	Analiza dos modelos de producción de software: la catedral representa el modelo de desarrollo más hermético y vertical característico del Software propietario y por otro lado el bazar, con su dinámica horizontal y "bulliciosa", que caracterizó al desarrollo del kernel Linux y otros proyectos de Software Libre que se potenciaron con el trabajo comunitario a través de Internet del código abierto
Ley de Brooks	Es un principio utilizado en el desarrollo de software que afirma que "añadir más efectivos a un proyecto de software en retraso, lo retrasará más".1 Fue acuñado por Fred Brooks en su trabajo de 1975 The Mythical Man-Month. El corolario de la ley de Brooks es que cuando se incorpora una persona en un proyecto, éste se ralentiza en lugar de acelerarse. Brooks también afirmó que "Nueve mujeres no pueden tener un bebé en un mes".
Método de desarrollo de sistemas dinámicos (DSDM)	Es un método que provee un framework para el desarrollo ágil de software, apoyado por su continua implicación del usuario en un desarrollo iterativo y creciente que sea sensible a los requerimientos cambiantes, para desarrollar un sistema que reúna las necesidades de la empresa en tiempo y presupuesto. Es uno de un número de métodos de desarrollo ágil de software y forma parte de la alianza ágil. DSDM fue desarrollado en el Reino Unido en los años 90 por un consorcio de proveedores y de expertos en la materia del desarrollo de sistemas de información (IS)
Metodología de diseño construccionista (CDM)	La Metodología de diseño construccionista (MDL) fue desarrollado por la inteligencia artificial (AI) investigador Kristinn R. Thorisson y sus estudiantes en la Universidad de Columbia y la Universidad de Reykjavik para su uso en el desarrollo de la robótica cognitiva, humanoides de comunicación y los sistemas de IA amplias CDM es una Metodología Constructivista de Diseño, es un enfoque para la construcción de sistemas altamente modulares de muchos componentes que interactúan. La fuerza de MDL reside en la simplificación del modelo de los sistemas funcionales complejos, que requieren

	múltiples evoluciones arquitectónicas de las jerarquías de flujo de datos y de control enredados. MDL se basa en pasos de diseño iterativos que conducen a la creación de una red de módulos que interactúan con nombre, que se comunican a través de corrientes de tipo explícito y mensajes discretos. MDL se ha utilizado en la creación de muchos sistemas, incluyendo la robótica, la animación facial, la simulación a gran escala y los seres humanos virtuales. Uno de los primeros sistemas fue MIRAGE, un ser humano simulado en un entorno de realidad aumentada que podrían interactuar con la gente a través del habla y el gesto. El Dr. Kristinn R. Thorisson es un investigador islandés Inteligencia Artificial, fundador del Instituto islandés por las máquinas inteligentes (IIIM) y co-fundador y ex co-director de CADIA: Centro de Análisis y Diseño de Agentes Inteligentes. Thorisson es uno de los principales defensores de la integración de sistemas de inteligencia artificial. Él es un defensor de la Inteligencia Artificial General (AGI) y ha propuesto una nueva metodología para lograr ia fuerte.
Model-driven Architecture (MDA)	Es un acercamiento al diseño de software, propuesto y patrocinado por el Object Management Group (OMG). MDA se ha concebido para dar soporte a la ingeniería dirigida a modelos de los sistemas de software.MDA es una arquitectura que proporciona un conjunto de guías para estructurar especificaciones expresadas como modelos. La funcionalidad del sistema será definida en primer lugar como un modelo independiente de la plataforma (Platform-Independent Model o PIM) a través de un lenguaje específico para el dominio del que se trate. Dado un modelo de definición de la plataforma (Platform Definition Model o PDM) correspondiente a CORBA, .NET, web, etcétera, el modelo PIM puede traducirse entonces a uno o más modelos específicos de la plataforma (Platform-Specific Models o PSM) para la implementación correspondiente, usando diferentes lenguajes específicos del dominio, o lenguajes de propósito general como Java, C#, Python, etc. La traducción entre el PIM y los PSM se realizan normalmente utilizando herramientas automatizadas, como herramientas de transformación de modelos (por ejemplo aquellas herramientas que cumplen con el nuevo estándar de OMG denominado Query/View/Transformation o QVT). El proceso completo se encuentra documentado en un documento que actualiza y mantiene OMG denominado "MDA guide"
Modelo en cascada	Es el enfoque metodológico que ordena rigurosamente las etapas del proceso para el desarrollo de software, de tal forma que el inicio de cada etapa debe esperar a la finalización de la etapa anterior. Al final de cada etapa, el modelo está diseñado para llevar a cabo una revisión final, que se encarga de determinar si el proyecto está listo para avanzar a la siguiente fase. Este modelo fue el primero en originarse y es la base de todos los demás modelos de ciclo de vida. La versión original fue propuesta por Winston W. Royce en 1970 y posteriormente revisada por Barry Boehm en 1980 e Ian Sommerville en 1985.
Modelo en espiral	Es un modelo de ciclo de vida del software definido por primera vez por Barry Boehm en 1986, utilizado generalmente en la Ingeniería de software. Las actividades de este modelo se conforman en una espiral, en la que cada bucle o iteración representa un conjunto de actividades. Las actividades no están fijadas a ninguna prioridad, sino que las siguientes se eligen en función del análisis de riesgo, comenzando por el bucle interior. **El IEEE clasifica al desarrollo en espiral como modelo no operativo en sus clasificaciones de MCV.**
No te repitas (DRY)	También conocido como “Una vez y sólo una”, es una filosofía de definición de procesos que promueve la reducción de la duplicación especialmente en computación. Según este principio toda pieza de información nunca debería ser duplicada debido a que la duplicación incrementa la dificultad en los cambios y evolución posterior, puede perjudicar la claridad y crear un espacio para posibles inconsistencias. Por "pieza de información" podemos entender, en un sentido amplio, desde datos almacenados en una base de datos pasando por el código fuente de un programa de software hasta llegar a información textual o documentación
No vas a necesitarlo (YAGNI)	En inglés 'You Ain't Gonna Need It') consiste en que no se debe nunca agregar funcionalidad excepto que sea necesario. La tentación de escribir código que no es necesario, pero que puede serlo en un futuro tiene las siguientes desventajas: Cuando se desarrollan nuevas funcionalidades se suele sacrificar el tiempo que se destinaría para la funcionalidad básica. Las nuevas características deben ser depuradas, documentadas y soportadas. Una nueva funcionalidad impone límites a lo que puede ser hecho en el futuro y puede impedir la implementación de una característica necesaria en el futuro. Hasta que está definido para qué se puede necesitar es imposible saber qué debe hacer. Puede suceder que cuando se requieran no funcionen correctamente. Puede derivar en un código inflado: El programa se vuelve grande y complicado pero que tampoco proporciona más funcionalidad. Excepto que haya especificaciones y algún tipo de control de versiones, esta característica puede no ser conocida por los programadores que podrían hacer uso de ella. Puede inducir a que se agreguen nuevas funcionalidades y como resultado puede llevar a un efecto 'bola de nieve' que puede consumir tiempo ilimitado y recursos, a cambio de ningún beneficio.
Peor es mejor	O estilo Nueva Jersey, es el nombre de una técnica de desarrollo de software, o filosofía de diseño, en la cual la simplicidad en la interfaz y en la implementación es más importante que

	cualquier otra propiedad del sistema (incluyendo corrección, consistencia y completitud).

Principio de Hollywood	La inversión de control, también conocida como *Principio de Hollywood* en referencia al *slogan* de los directivos de Hollywood *"No nos llames, nosotros te llamaremos"*, es un principio de diseño que busca mayor cohesión y menor acoplamiento entre los componentes de un sistema informático.
	A diferencia de la tradición procedural, los componentes de mayor nivel son responsables de proporcionar abstracciones a los de menor nivel. Las instancias concretas de estas abstracciones son reemplazables y necesitan una interfaz común. El flujo por lo tanto se invierte, ahora son las capas superiores las que controlan a las inferiores, y no al revés.
	<?php
	public function save(array $account_data) {
	$account = Account::createFromArray($account_data);
	// we call a higher-level class
	$em = \Application::GetDoctrine()->getEntityManager();
	$em->persist($account);
	// ...
	}
	// A higher level class provides an abstraction
	public function save(array $account_data, ObjectManager $om) {
	$account = Account::createFromArray($account_data);
	$om->persist($account);
	// ...
	}
	La técnica más común para conseguir la inversión de control es la **inyección de dependencias**. En los lenguajes de programación dinámicos existen algunas alternativas, como la metaprogramación, que no veremos en este material. La clave consiste en asegurar un flujo unidireccional de ejecución, desde clases de mayor nivel de abstracción hacia clases de mayor detalle. Los sistemas informáticos se convierten así en un conjunto coherente de piezas sustituíbles. Estas piezas no se transforman cada vez que surge un nuevo requisito, sino que se reemplazan.
	Tell, don't ask
	En la Biblioteca Pragmática, Andy Hunt y Dave Thomas escribieron un artículo llamado *Tell, don't ask* en el que describían este principio. El artículo empezaba con una cita del libro *Smalltalk By Example*, disponible gratuitamente en la Red.
	Procedural code gets information then makes decisions. Object-oriented code tells objects to do things.
	En su artículo, Hunt y Thomas advierten de los riesgos de romper el encapsulamiento a través de la extracción de los datos de otras clases. Esto es, "El principio fundamental de la programación orientada aobjetos es la unificación de los métodos y los datos"
	Exponiendo el estado de una clase a las demás para que aquellas puedan manipularla no sólo estamos violando su encapsulamiento y asumiendo riesgos, sino que estamos moviendo responsabilidades hacia las clases de mayor nivel.
	Cuando una clase requiera conocer el estado de otra clase, conviene preguntarse qué uso va a darse a esos datos, y mover siempre que sea posible esa manipulación a la clase expuesta, evitando la dispersión y propagación de estado.
	Martin Fowler escribe también sobre este principio, y aunque coincide en la importancia de la co-locación de estado y comportamiento, recuerda que también deben tomarse en consideración otros aspectos como la separación en capas

Principio KISS (Keep It Simple, Stupid o Manténlo simple)	El principio KISS establece que la mayoría de sistemas funcionan mejor si se mantienen simples que si se hacen complejos; por ello, la simplicidad debe ser mantenida como un objetivo clave del diseño, y cualquier complejidad innecesaria debe ser evitada. Este principio se registra por primera vez en la Marina de los Estados Unidos en 1960, y se atribuye principalmente a Kelly Johnson, ingeniero jefe en Lockheed Skunk Works.
Proceso Unificado	El Proceso Unificado de Desarrollo Software o simplemente Proceso Unificado es un marco de desarrollo de software que se caracteriza por estar dirigido por casos de uso, centrado en la arquitectura y por ser iterativo e incremental. El refinamiento más conocido y documentado del Proceso Unificado es el Proceso Unificado de Rational o simplemente RUP. El Proceso Unificado no es simplemente un proceso, sino un marco de trabajo extensible que puede ser adaptado a organizaciones o proyectos específicos. De la misma forma, el Proceso Unificado de Rational, también es un marco de trabajo extensible, por lo que muchas veces resulta imposible decir si un refinamiento particular del proceso ha sido derivado del Proceso Unificado o del RUP. Por dicho motivo, los dos nombres suelen utilizarse para referirse a un mismo concepto. El nombre Proceso Unificado se usa para describir el proceso genérico que incluye aquellos elementos que son comunes a la mayoría de los refinamientos existentes. También permite evitar problemas legales ya que Proceso Unificado de Rational o RUP son marcas registradas por IBM (desde su compra de Rational Software Corporation en 2003). El Proceso Unificado es un marco de desarrollo iterativo e incremental compuesto de cuatro fases denominadas Inicio, Elaboración, Construcción y Transición. Cada una de estas fases es a su vez dividida en una serie de iteraciones (la de inicio puede incluir varias iteraciones en

	proyectos grandes). Estas iteraciones ofrecen como resultado un incremento del producto desarrollado que añade o mejora las funcionalidades del sistema en desarrollo. El Lenguaje unificado de modelado, no es el sucesor de la oleada de métodos de análisis y diseño orientados a objetos que surgió a finales de la década de los 1980 y principios de la siguiente. El UML unifica, sobre todo, los métodos de Booch, Rumbaugh, Brühl (OMT) y Jacobson, pero su alcance ha llegado a formar parte fundamental de la Ingeniería de Software tras su estandarización en 1997 con el OMG (Object Management Group o Grupo de administración de objetos).
Proceso unificado abierto	El Proceso Unificado de Desarrollo Software o simplemente Proceso Unificado es un marco de desarrollo de software que se caracteriza por estar dirigido por casos de uso, centrado en la arquitectura y por ser iterativo e incremental. El refinamiento más conocido y documentado del Proceso Unificado es el Proceso Unificado de Rational o simplemente RUP. El Proceso Unificado no es simplemente un proceso, sino un marco de trabajo extensible que puede ser adaptado a organizaciones o proyectos específicos. De la misma forma, el Proceso Unificado de Rational, también es un marco de trabajo extensible, por lo que muchas veces resulta imposible decir si un refinamiento particular del proceso ha sido derivado del Proceso Unificado o del RUP. Por dicho motivo, los dos nombres suelen utilizarse para referirse a un mismo concepto. El nombre Proceso Unificado se usa para describir el proceso genérico que incluye aquellos elementos que son comunes a la mayoría de los refinamientos existentes. También permite evitar problemas legales ya que Proceso Unificado de Rational o RUP son marcas registradas por IBM (desde su compra de Rational Software Corporation en 2003). El primer libro sobre el tema se denominó, en su versión española, El Proceso Unificado de Desarrollo de Software (ISBN 84-7829-036-2) y fue publicado en 1999 por Ivar Jacobson, Grady Booch y James Rumbaugh, conocidos también por ser los desarrolladores del UML, el Lenguaje Unificado de Modelado. Desde entonces los autores que publican libros sobre el tema y que no están afiliados a Rational utilizan el término Proceso Unificado, mientras que los autores que pertenecen a Rational favorecen el nombre de Proceso Unificado de Rational.
Proceso unificado ágil (AUP)	Es una versión simplificada del Proceso Unificado de Racional (RUP). Este describe de una manera simple y fácil de entender la forma de desarrollar aplicaciones de software de negocio usando técnicas ágiles y conceptos que aún se mantienen válidos en RUP. El AUP aplica técnicas ágiles incluyendo Desarrollo Dirigido por Pruebas. Características.- • Iterativo e Incremental. • Descomposición de un proyecto grande en mini-proyectos • Cada mini-proyecto es una iteración • Las iteraciones deben estar controladas • Cada iteración trata un conjunto de casos de uso Ventajas del enfoque iterativo • Detección temprana de riesgos • Administración adecuada del cambio • Mayor grado de reutilización • Mayor experiencia para el grupo de desarrollo CICLO DE VIDA DEL PROCESO UNIFICADO AGIL Fase de Concepción.- • Objetivo: Definir la razón de ser y el alcance del proyecto. Estudio de oportunidad. Visión = QUÉ + PARA QUÉ + CUÁNTO • Actividades Especificación de los criterios de éxito del proyecto Definición de los requisitos Estimación de los recursos necesarios Cronograma inicial de fases • Artefactos (Pieza de información producida, modificada y utilizada en un Proceso) Documento de definición del proyecto Fase de Elaboración.- • Objetivo: Establecer un plan de proyecto y una arquitectura correcta del sistema

	• Actividades Análisis del dominio del problema Definición de la arquitectura básica Análisis de riesgos Planificación del proyecto • Artefactos Modelo del dominio Modelo de procesos Modelo funcional de alto nivel Arquitectura básica Fase de Construcción.- • Construcción Objetivo: Desarrollar el sistema a lo largo de una serie de iteraciones Actividades • Análisis • Diseño • Implementación / Codificación • Pruebas (individuales, de integración) Fase de Transición.- El sistema se lleva a los entornos de preproducción donde se somete a pruebas de validación y aceptación y finalmente se despliega en los sistemas de producción. VENTAJAS.- El personal sabe lo que esta haciendo: no obliga a conocer detalles. Simplicidad: apuntes concisos. Agilidad: procesos simplificados del RUP Centrarse en actividades de alto valor: esenciales para el desarrollo. Herramientas independientes: a disposición del usuario. Fácil adaptación de este producto: de fácil acomodo (HTML) DESVENTAJAS.- El AUP es un producto muy pesado en relación al RUP. Como es un proceso simplificado, muchos desarrolladores eligen trabajar con el RUP, por tener a disposición mas detalles en el proceso.
Proceso Unificado de Rational (RUP)	RUP (por sus siglas en ingés de Rational Unified Process) es un proceso de desarrollo de software desarrollado por la empresa Rational Software, actualmente propiedad de IBM.1 Junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, diseño, implementación y documentación de sistemas orientados a objetos. El RUP no es un sistema con pasos firmemente establecidos, sino un conjunto de metodologías adaptables al contexto y necesidades de cada organización. También se conoce por este nombre al software, también desarrollado por Rational, que incluye información entrelazada de diversos artefactos y descripciones de las diversas actividades. Está incluido en el Rational Method Composer (RMC), que permite la personalización de acuerdo con las necesidades. Originalmente se diseñó un proceso genérico y de dominio público, el Proceso Unificado, y una especificación más detallada, el Rational Unified Process, que se vendiera como producto independiente.
Programación extrema (XP)	O eXtreme Programming (de ahora en adelante, XP) es una metodología de desarrollo de la ingeniería de software formulada por Kent Beck, autor del primer libro sobre la materia, Extreme Programming Explained: Embrace Change (1999). Es el más destacado de los procesos ágiles de desarrollo de software. Al igual que éstos, la programación extrema se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. Los defensores de la XP consideran que los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos. Creen que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos. Se puede considerar la programación extrema como la adopción de las mejores metodologías de desarrollo de acuerdo a lo que se pretende llevar a cabo con el proyecto, y aplicarlo de manera dinámica durante el ciclo de vida del software.
Rápido y sucio	Quick-and-dirty es un término usado en referencia a una forma sencilla de solucionar un problema. Este uso es popular entre los hackers, que lo usan para describir una solución primitiva o una implementación que es imperfecta, poco elegante, o incluso inadecuada, pero que resuelve o enmascara el problema, y generalmente es más rápida y sencilla de usar que

	buscar una solución apropiada. Las soluciones Quick-and-dirty a menudo se centran en un caso concreto de un problema en lugar de arreglar la causa del problema general. Por eso, se usan algunas veces para mantener una parte del software o hardware funcionando temporalmente hasta que se pueda encontrar una solución adecuada. La frase también se usa frecuentemente para describir documentos o tutoriales que sólo dan una visión general de cómo hacer algo, sin entrar en demasiados detalles sobre por qué o cómo funciona.
Scrum	Scrum es el nombre con el que se denomina a los marcos de desarrollo ágiles caracterizados por: • Adoptar una estrategia de desarrollo incremental, en lugar de la planificación y ejecución completa del producto. • Basar la calidad del resultado más en el conocimiento tácito de las personas en equipos auto organizados, que en la calidad de los procesos empleados. • Solapamiento de las diferentes fases del desarrollo, en lugar de realizar una tras otra en un ciclo secuencial o en cascada. Scrum es el nombre con el que se denomina a los marcos de desarrollo ágiles caracterizados por: • Adoptar una estrategia de desarrollo incremental, en lugar de la planificación y ejecución completa del producto. • Basar la calidad del resultado más en el conocimiento tácito de las personas en equipos auto organizados, que en la calidad de los procesos empleados. • Solapamiento de las diferentes fases del desarrollo, en lugar de realizar una tras otra en un ciclo secuencial o en cascada. Scrum permite la creación de equipos autos organizados impulsando la co-localización de todos los miembros del equipo, y la comunicación verbal entre todos los miembros y disciplinas involucrados en el proyecto. Un principio clave de Scrum es el reconocimiento de que durante un proyecto los clientes pueden cambiar de idea sobre lo que quieren y necesitan (a menudo llamado requirements churn), y que los desafíos impredecibles no pueden ser fácilmente enfrentados de una forma predictiva y planificada. Por lo tanto, Scrum adopta una aproximación pragmática, aceptando que el problema no puede ser completamente entendido o definido, y centrándose en maximizar la capacidad del equipo de entregar rápidamente y responder a requisitos emergentes. Las características más marcadas que se logran notar en Scrum serían: gestión regular de las expectativas del cliente, resultados anticipados, flexibilidad y adaptación, retorno de inversión, mitigación de riesgos, productividad y calidad, alineamiento entre cliente y equipo, por último equipo motivado. Cada uno de estos puntos mencionados hacen que el Scrum sea utilizado de manera regular en un conjunto de buenas prácticas para el trabajo en equipo y de esa manera obtener resultados posibles. Existen varias implementaciones de sistemas para gestionar el proceso de Scrum, que van desde notas amarillas "post-it" y pizarras hasta paquetes de software. Una de las mayores ventajas de Scrum es que es muy fácil de aprender, y requiere muy poco esfuerzo para comenzarse a utilizar. Así, si se utiliza una pizarra con notas autoadhesivas cualquier miembro del equipo podrá ver tres columnas: trabajo pendiente ("backlog"), tareas en proceso ("in progress") y hecho ("done"). De un solo vistazo, una persona puede ver en qué están trabajando los demás en un momento determinado.
Software System Safety	En la ingeniería de software, la seguridad del software optimiza la seguridad del sistema en el diseño, desarrollo, uso y mantenimiento de sistemas informáticos y su integración con los sistemas de hardware de seguridad crítica en un entorno operativo. Introducción Para la seguridad del software del sistema, ningún elemento del programa de desarrollo total seguridad y software, se puede permitir que funcione de forma independiente. Tanto sistemas simples como sistemas altamente integrados están experimentando un extraordinario crecimiento en el uso de computadoras y software para supervisar y / o controlar los subsistemas o funciones críticas para la seguridad. Un error de especificación de software, un fallo de diseño, o la falta de requisitos genéricos de seguridad críticos pueden contribuir a causar un fallo del sistema o una decisión humana errónea. Para lograr un nivel aceptable de seguridad para el software utilizado en aplicaciones críticas, la ingeniería de seguridad del software del sistema pone énfasis principal en la definición de los requisitos el proceso de diseño conceptual. El software crítico debe entonces recibir atención continua de gestión y análisis de ingeniería en todo el ciclo de vida de desarrollo y operacionales del sistema. La seguridad del software del sistema está directamente relacionada con los aspectos de diseño y atributos de seguridad más críticos en la funcionalidad del software y del sistema, mientras que los atributos de calidad de software son intrínsecamente diferentes y requieren un examen estándar y rigor en el desarrollo. El Nivel de Rigor (LOR) es un enfoque diferente a la calidad del software y la garantía de diseño de software es un pre-requisito de que un proceso de software adecuado es confiable. Los conceptos y estándares de LOR como DO-178C LOR no

	son un sustituto para la seguridad del software. El Software de seguridad según la norma IEEE STD-1228 y MIL-STD-882E se centra en garantizar que se cumplan los requisitos de seguridad explícitas y se verifique utilizando enfoques funcionales a partir de un análisis de los requisitos de seguridad y la perspectiva de prueba.

(vi) Anexo Herramientas para Pruebas

El siguiente es un listado de algunas de las herramientas encontradas en Internet.

Se dividen en gratuitas y comerciales y a su vez en:

1. Herramientas de Gestión de pruebas
2. Herramientas para pruebas funcionales
3. Herramientas para pruebas de carga y rendimiento.

Herramientas Gratuitas

1) Herramientas de gestión de pruebas

Herramienta	Sitio
Bugzilla Testopia	http://www.mozilla.org/projects/testopia/
FitNesse	http://fitnesse.org/
qaManager	http://sourceforge.net/projects/qamanager
qaBook	http://www.qabook.com/
RTH (open source)	https://sourceforge.net/projects/rth/
Salome-tmf	https://wiki.objectweb.org/salome-tmf/
Squash TM	http://www.squashtest.org/index.php
Test Environment Toolkit	http://tetworks.opengroup.org/Products/tet.htm
TestLink	http://www.teamst.org/
Testitool	http://majordojo.com/testitool/
XQual Studio	http://www.xqual.com/
Radi-testdir	http://sourceforge.net/projects/radi-testdir/
Data Generator	http://www.generatedata.com/

Figura 7.3.4 Herramientas Gratuitas

2) Herramientas para pruebas funcionales

Herramienta	Sitio
Selenium	http://docs.seleniumhq.org/download/
Soapui	http://www.soapui.org/
Watir (aplicaciones web en Ruby)	http://wtr.rubyforge.org/
WatiN (aplicaciones web en .Net)	http://watin.sourceforge.net/
Capedit	http://www.packetsquare.com/
Canoo WebTest	http://webtest.canoo.com/webtest/
Solex	http://solex.sourceforge.net/
Imprimatur	http://imprimatur.wikispaces.com/
SAMIE	http://samie.sourceforge.net/
ITP	http://www.incanica.com/itp.html
WET	http://wet.qantom.org/
WebInject	http://www.webinject.org/

Figura 7.3.5 Herramientas Pruebas funcionales

3) Herramientas para pruebas de carga y rendimiento

Herramienta	Sitio
FunkLoad	http://funkload.nuxeo.org/
FWPTT load testing	http://fwptt.sourceforge.net/
loadUI	http://www.loadui.org/
jmeter	http://jmeter.apache.org/

Figura 7.3.6 Herramientas Carga y Rendimiento

Herramientas Comerciales:

1) Herramientas de gestión de pruebas

Herramienta	Sitio
HP Quality Center/ALM	http://www8.hp.com/us/en/software/enterprise-software.html
QA Complete	http://www.testmanagement.com/qacomplete.html
qaBook	http://www.qabook.com/
T-Plan Professional	http://www.t-plan.com/product_trials.html
SMARTS	http://www.testworks.com/
QAS.Test Case Studio	http://www.objentis.com/index.php
PractiTest	http://www.practitest.com/product/
SpiraTest	http://www.inflectra.com/SpiraTest/Default.aspx
TestLog	http://www.testlog.com/download.htm
ApTest Manager	http://www.aptest.com/tools.html
Zephyr	http://www.getzephyr.com/

Figura 7.3.7 Herramientas Gestión de Pruebas

2) Herramientas para pruebas funcionales

Herramienta	Sitio
QuickTest Pro	http://www8.hp.com/us/en/software/enterprise-software.html
Rational Robot	http://www-01.ibm.com/software/rational/
Sahi	http://sahi.co.in/sahi-pro/
SoapTest	http://www.parasoft.com/jsp/products.jsp?itemId=13
Test Complete	http://smartbear.com/products/qa-tools/automated-testing-tools
QA Wizard	http://www.seapine.com/qawizard.html
Squish	http://www.froglogic.com/
vTest	http://www.verisium.com/products/vTest/index.html
Internet Macros	http://www.iopus.com/imacros/

Figura 7.3.8 Herramientas de Pruebas Funcionales II

3) Herramientas para pruebas de carga y rendimiento

Herramienta	Sitio
HP LoadRunner	http://www8.hp.com/us/en/software-solutions/software.html?compURI=1175451#.UXI1xKLTx2c
LoadStorm	http://loadstorm.com/
NeoLoad	http://www.neotys.com/
WebLOAD Professional	http://www.radview.com/product/product.aspx
Forecast	http://www.facilita.co.uk/products/
ANTS – Advanced .NET Testing System	http://www.red-gate.com/products/dotnet-development/
Webserver Stress Tool	http://www.paessler.com/webstress
Load Impact	http://loadimpact.com/

Figura 7.3.9 Herramientas de Carga y Rendimiento

Herramientas Todo en Uno

Herramienta	Sitio
Test Studio	http://www.telerik.com/automated-testing-tools

Figura 7.3.10 Herramientas Todo en uno

4. Índice de Ilustraciones