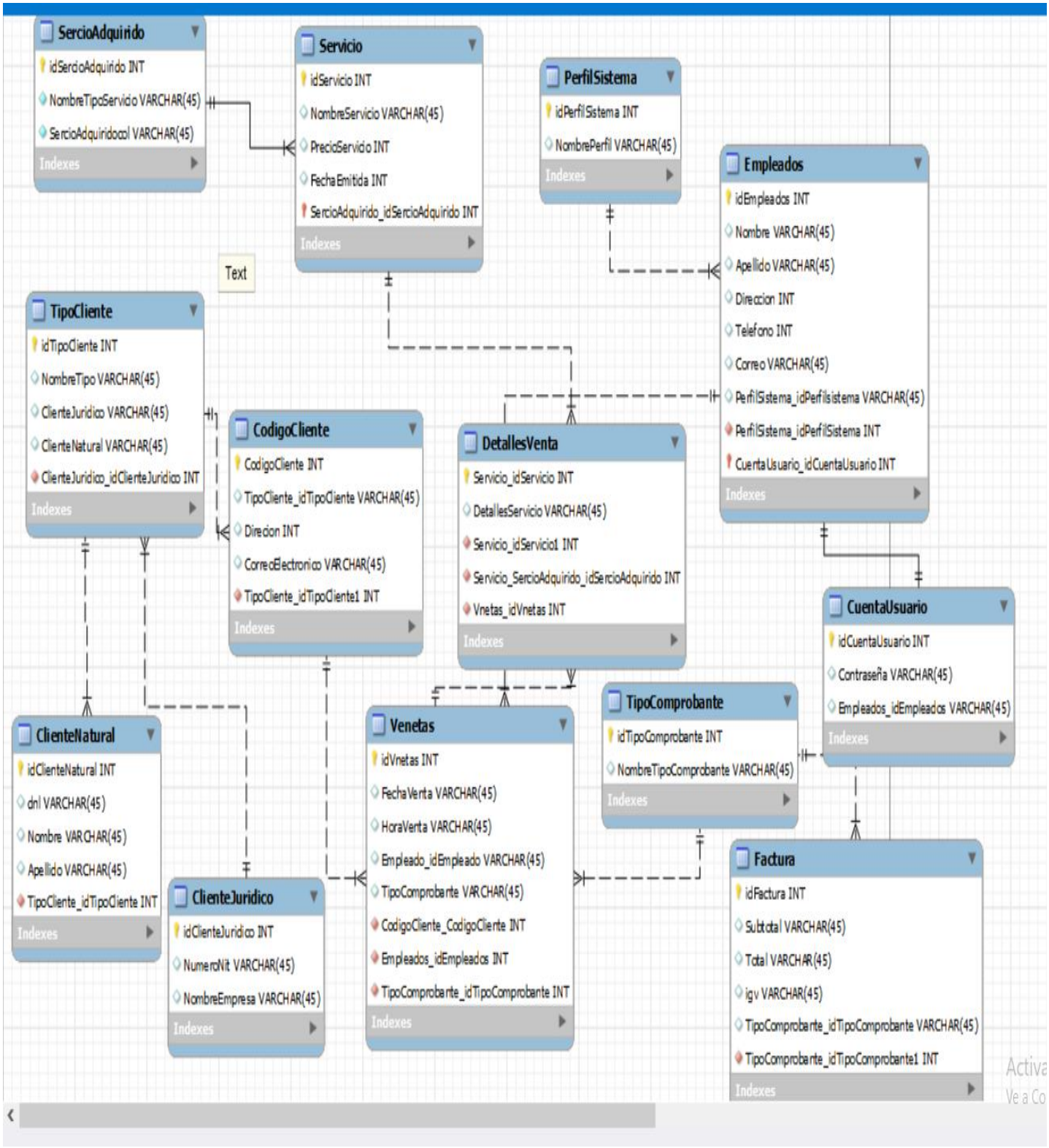
INTRODUCCIÓN

En el desarrollo de software, la creación de aplicaciones eficientes y funcionales requiere una cuidadosa planificación y diseño. Los modelos conceptuales y lógicos se erigen como pilares fundamentales en este proceso, sirviendo como mapas que ayudan a los desarrolladores a convertir sus ideas en realidad. El modelo conceptual se enfoca en lo básico y esencial de un sistema sin entrar en detalles técnicos, como una especie de esquema abstracto. Es la piedra angular sobre la cual se erige el edificio del software, trazando las líneas maestras de la solución que guían a los desarrolladores en el proceso de construir aplicaciones eficaces y útiles.

Por otro lado, el modelo lógico desciende a un nivel más detallado, estableciendo las reglas y relaciones que gobiernan el comportamiento interno del sistema. Aquí, se traducen las ideas abstractas en elementos tangibles, como entidades, atributos y relaciones, permitiendo una comprensión más profunda de la interconexión de los componentes. Ambos modelos, conceptual y lógico, colaboran armoniosamente para proporcionar una visión completa y coherente del software en gestación, facilitando no solo su desarrollo, sino también su mantenimiento y evolución a lo largo del tiempo. En este intrincado proceso de creación, la conjunción de estos modelos se convierte en la brújula que orienta a los desarrolladores hacia la consecución de soluciones robustas y efectivas.

MODELO LÓGICO



NORMALIZACIÓN DEL MODELO LÓGICO

1 Forma Normal

```
CREATE TABLE IF NOT EXISTS `sistemaISP`.`SercioAdquirido` (
  `idSercioAdquirido` INT NOT NULL AUTO_INCREMENT,
  `NombreTipoServicio` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`idSercioAdquirido`)
) ENGINE = InnoDB;


CREATE TABLE IF NOT EXISTS `sistemaISP`.`Servicio` (
  `idServicio` INT NOT NULL AUTO_INCREMENT,
  `NombreServicio` VARCHAR(45) NULL,
  `PrecioServicio` DECIMAL(10, 2) NULL,
  `FechaEmitida` DATE NULL,
  `SercioAdquirido_idSercioAdquirido` INT NOT NULL,
  PRIMARY KEY (`idServicio`),
  CONSTRAINT `fk_Servicio_SercioAdquirido`
    FOREIGN KEY (`SercioAdquirido_idSercioAdquirido`)
    REFERENCES `sistemaISP`.`SercioAdquirido` (`idSercioAdquirido`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION
) ENGINE = InnoDB;
```

2 Forma Nomal

```
CREATE TABLE IF NOT EXISTS `sistemaISP`.`SercioAdquirido` (
  `idSercioAdquirido` INT NOT NULL AUTO_INCREMENT,
  `NombreTipoServicio` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`idSercioAdquirido`)
) ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `sistemaISP`.`Servicio` (
  `idServicio` INT NOT NULL AUTO_INCREMENT,
  `NombreServicio` VARCHAR(45) NULL,
  `PrecioServicio` DECIMAL(10, 2) NULL,
  `FechaEmitida` DATE NULL,
  `SercioAdquirido_idSercioAdquirido` INT NOT NULL,
  PRIMARY KEY (`idServicio`),
  CONSTRAINT `fk_Servicio_SercioAdquirido`
    FOREIGN KEY (`SercioAdquirido_idSercioAdquirido`)
    REFERENCES `sistemaISP`.`SercioAdquirido` (`idSercioAdquirido`)
    ON DELETE NO ACTION
```

```
    ON UPDATE NO ACTION
) ENGINE = InnoDB;
```

3 Forma Norma

```
CREATE TABLE IF NOT EXISTS `sistemaISP`.`SercioAdquirido` (
  `idSercioAdquirido` INT NOT NULL AUTO_INCREMENT,
  `NombreTipoServicio` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`idSercioAdquirido`)
) ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `sistemaISP`.`Servicio` (
  `idServicio` INT NOT NULL AUTO_INCREMENT,
  `NombreServicio` VARCHAR(45) NULL,
  `PrecioServicio` DECIMAL(10, 2) NULL,
  `FechaEmitida` DATE NULL,
  `SercioAdquirido_idSercioAdquirido` INT NOT NULL,
  PRIMARY KEY (`idServicio`),
  CONSTRAINT `fk_Servicio_SercioAdquirido`
    FOREIGN KEY (`SercioAdquirido_idSercioAdquirido`)
    REFERENCES `sistemaISP`.`SercioAdquirido` (`idSercioAdquirido`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION
) ENGINE = InnoDB;
```

1 Forma Normal

```
-- Tabla ClienteJuridico
CREATE TABLE IF NOT EXISTS `sistemaISP`.`ClienteJuridico` (
  `idClienteJuridico` INT NOT NULL,
  `NombreClienteJuridico` VARCHAR(100) NOT NULL,
  PRIMARY KEY (`idClienteJuridico`)
);

-- Tabla TipoCliente
CREATE TABLE IF NOT EXISTS `sistemaISP`.`TipoCliente` (
  `idTipoCliente` INT NOT NULL,
  `NombreTipo` VARCHAR(45) NOT NULL,
  `ClienteJuridico_idClienteJuridico` INT NOT NULL,
  PRIMARY KEY (`idTipoCliente`),
  FOREIGN KEY (`ClienteJuridico_idClienteJuridico`)
    REFERENCES `sistemaISP`.`ClienteJuridico` (`idClienteJuridico`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION
);
```

2 formal Normal

```
-- Tabla ClienteJuridico
CREATE TABLE IF NOT EXISTS `sistemaISP`.`ClienteJuridico` (
  `idClienteJuridico` INT NOT NULL,
  `NombreClienteJuridico` VARCHAR(100) NOT NULL,
  PRIMARY KEY (`idClienteJuridico`)
);

-- Tabla TipoCliente
CREATE TABLE IF NOT EXISTS `sistemaISP`.`TipoCliente` (
  `idTipoCliente` INT NOT NULL,
  `NombreTipo` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`idTipoCliente`)
);
```

3 Forma Normal

```
-- Tabla ClienteJuridico
CREATE TABLE IF NOT EXISTS `sistemaISP`.`ClienteJuridico` (
  `idClienteJuridico` INT NOT NULL,
  `NombreClienteJuridico` VARCHAR(100) NOT NULL,
  PRIMARY KEY (`idClienteJuridico`)
);

-- Tabla TipoCliente
CREATE TABLE IF NOT EXISTS `sistemaISP`.`TipoCliente` (
  `idTipoCliente` INT NOT NULL,
  `NombreTipo` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`idTipoCliente`)
);
```

Normalizacion adicional

```
-- Tabla TipoComprobante
CREATE TABLE IF NOT EXISTS `sistemaISP`.`TipoComprobante` (
  `idTipoComprobante` INT NOT NULL,
  `NombreTipoComprobante` VARCHAR(45) NULL,
  PRIMARY KEY (`idTipoComprobante`)
) ENGINE = InnoDB;
```

```sql
-- Tabla Comprobante
CREATE TABLE IF NOT EXISTS `sistemaISP`.`Comprobante` (
  `idComprobante` INT NOT NULL,
  `NumeroComprobante` VARCHAR(20) NOT NULL,
  `FechaEmision` DATE NOT NULL,
  `idTipoComprobante` INT NOT NULL,
  PRIMARY KEY (`idComprobante`),
  FOREIGN KEY (`idTipoComprobante`)
    REFERENCES `sistemaISP`.`TipoComprobante` (`idTipoComprobante`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION
) ENGINE = InnoDB;
```

DICCIONARIO DE DATOS SEGÚN EL MODELO LÓGICO

NOMBRE	TABLA REGISTRO		
FECHA	15- 12-2023		
DESCRIPCIÓN	REGISTRO USUARIOS		
CAMPO	**TIPO DE DATO**	**TAMAÑO**	**DESCRIPCIÓN**
Nombre	*Varchar*	*20*	*Nombre de los usuarios*
Apellido	*Varchar*	*20*	*Apellido de los usuarios*
Cedula	*int*	*10*	*Identificación única del cliente*
Correo	*Varchar*	*50*	*Correo electrónico del usuario*
Celular	*int*	*10*	*Celular del usuario*
Dirección	*Varchar*	*50*	*Dirección del usuario*
Tipo de usuario	*int*	*1*	*Rol del usuario*
Contraseña	*Varchar*	*16*	*Contraseña del usuario*

NOMBRE	FINANCIAMIENTO		
FECHA	15- 12-2023		
DESCRIPCIÓN	FINANCIAMIENTO CLIENTES		
CAMPO	**TIPO DE DATO**	**TAMAÑO**	**DESCRIPCIÓN**
Financiamiento id	*int*	*1*	*Financiamiento cliente*
N° contrato	*int*	*5*	*Contrato del cliente*
Cedula	*Entero*	*10*	*Identificación única del cliente*
Valor de cuota	*Float*	*20*	*Valor a pagar cuota*
N° de cuotas	*int*	*2*	*Numero cuotas*
Monto por pagar	*Float*	*20*	*Valor a pagar*

NOMBRE	TABLA DE FACTURACIÓN MEGAS		
FECHA	15- 12-2023		
DESCRIPCIÓN	REGISTRAR TIPO DE MEGAS, PLAN ADQUIRIDO Y SERVICIOS		
CAMPO	**TIPO DE DATO**	**TAMAÑO**	**DESCRIPCIÓN**
Financiamiento id	*int*	*5*	*Numero único de facturación*
N° Contrato	*int*	*5*	*Serial único*
Cedula	*Entero*	*10*	*Identificación única del cliente*
Total	*Float*	*20*	*Total a pagar*
Impuesto	*Float*	*13*	*Recaudo IVA*
Fecha	*Date*	*10*	*Fecha de la compra*

NOMBRE	TABLA DE CARTERA		
FECHA	15- 12-2023		
DESCRIPCIÓN	SEGUIMIENTO PLANES Y GESTION RECAUDACIÓN MORA		
CAMPO	**TIPO DE DATO**	**TAMAÑO**	**DESCRIPCIÓN**
Cartera id	*int*	*5*	*Numero único de facturación*
N° Caso	*int*	*5*	*Serial único*
Cedula	*Entero*	*10*	*Identificación única del cliente*
Estado de deuda	*Float*	*20*	*Paz y salvo*
Valor de la deuda	*Float*	*13*	*Total a pagar*

NOMBRE	TABLA DOCUMENTOS LEGALES		
FECHA	15- 12-2023		
DESCRIPCIÓN	CONTIENE DOCUMENTO A GENERAR ,DATOS DEL CLIENTE Y SERVICIO ADQUIRIDO		
CAMPO	**TIPO DE DATO**	**TAMAÑO**	**DESCRIPCIÓN**
Tipo de documento legal	*Entero*	*2*	*Tipo de documento a generar*
Cedula	*Entero*	*10*	*Identificación única del cliente*
Remitente	*Varchar*	*50*	*Persona que genera el documento*
Destinatario	*Varchar*	*50*	*Persona que va dirigido el documento*
Fecha id	*Date*	*10*	*Fecha en la que se general el documento*

NOMBRE	TABLA CASO		
FECHA	15- 12-2023		
DESCRIPCIÓN	CONTIENE TIPO DE CASO , NUMERO DE CASO Y DATOS CLIENTE		
CAMPO	**TIPO DE DATO**	**TAMAÑO**	**DESCRIPCIÓN**
Numero de caso	*Entero*	*5*	*Contiene Numero de caso*
Cedula	*Entero*	*10*	*Identificación única del cliente*
Caso id	*Entero*	*2*	*Tipo de caso*

NOMBRE	TABLA TIPOS DE CASO		
FECHA	15- 12-2023		
DESCRIPCIÓN	CONTIENE TIPO DE CASO,DESCRIPICIÓN TIPO DE CASO Y DATOS CLIENTE		
CAMPO	**TIPO DE DATO**	**TAMAÑO**	**DESCRIPCIÓN**
Descripción	*Entero*	*5*	*Descripción del caso*
Cliente	*Entero*	*10*	*Identificación única del cliente*
Caso id	*Entero*	*2*	*Tipo de caso*

POLITICAS DE SEGURIDAD

Normas para la gestión y control de acceso y perfiles:

- El rol de administrador tiene la posibilidad de insertar, eliminar y editar cualquier tipo de información.
- El rol de cliente solo puede modificar sus datos personales y consultar la información relacionada con los casos que tiene a su nombre.
- Las cuentas de usuario solo se emitirán después de un proceso de verificación.
- Los usuarios acceden al sistema mediante un formulario de ingreso (Login) donde se verifican su nombre de usuario y contraseña para comprobar si tiene autorización y permitir el acceso al sistema.
- Las contraseñas deben tener una longitud máxima de 16 caracteres y debe ser una combinación de letras, números y símbolos.
- Se desactivará el usuario de las personas que se retiren de la empresa.
- La contraseña tendrá una validez de 3 meses.
- Se guardará registro de intentos fallidos con la intención de ingresar al sistema.
- Después de 1 hora de uso, el software pedirá al usuario validar su cuenta nuevamente.
- Los usuarios se pueden deshabilitar o desactivar, pero nunca eliminar

- Normas básicas para la protección de datos:

- Realizar periódicamente copias de seguridad de las bases de datos y los sitios web de la empresa.
- Proteger el correo electrónico de la empresa.
- Utilizar software para prevención de pérdida de datos
- Instalar y mantener actualizado el software antivirus en todos los computadores que utilicen el software
- No descargar archivos de fuentes dudosas.
- No abrir correos electrónicos de dudosa procedencia.

CONCLUSIONES

NETOPANFINDER es un software que se utiliza para encontrar y gestionar redes de manera eficiente. Al desarrollar este tipo de software, la implementación de modelos conceptuales y lógicos proporciona diversos beneficios clave:

- Los modelos conceptuales, como diagramas de casos de uso y diagramas de clases, permiten una representación visual clara de la estructura y las interacciones del sistema. Esto facilita la comprensión tanto para los desarrolladores como para los usuarios finales.

- Los modelos lógicos, como diagramas de secuencia y diagramas de actividad, ayudan a definir el flujo de trabajo del software. Esto proporciona una guía paso a paso sobre cómo interactúan los diferentes componentes del sistema, mejorando la claridad en el diseño.

- Los modelos conceptuales sirven como una herramienta eficaz para la comunicación entre los desarrolladores, diseñadores y los interesados en el proyecto. Al utilizar un lenguaje visual, se reducen las ambigüedades y se mejora la comunicación.

- Los modelos lógicos son útiles para coordinar las actividades de desarrollo entre los miembros del equipo. Al comprender cómo se interconectan los diferentes módulos del software, los desarrolladores pueden colaborar de manera más efectiva.

- Los modelos conceptuales y lógicos proporcionan una base sólida para el desarrollo de software al definir claramente la arquitectura y las relaciones entre los componentes. Esto facilita las actualizaciones y mejoras futuras, ya que los desarrolladores tienen una comprensión completa de la estructura del sistema.

- Al emplear modelos lógicos, es posible anticipar y planificar la escalabilidad del software. Esto es crucial para adaptarse a cambios en los requisitos y asegurar que el software pueda crecer de manera eficiente a medida que las necesidades del usuario evolucionan.
- En resumen, la aplicación de modelos conceptuales y lógicos en el desarrollo de software como NETOPANFINDER ofrece ventajas significativas al proporcionar claridad en el diseño, facilitar la comunicación y colaboración entre los equipos, y mejorar la mantenibilidad y escalabilidad del sistema.