UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
MEDELLÍN

Implementación de solución de Software Plaza Core para gestión empresarial y operativa de aplicaciones omnicanal

Propuesta presentada a:

Universidad Santo Tomás-Sede Medellín

Primer Claustro Universitario de Colombia

Elaborado por:

Germán Esteban Plata Rincón

31 de enero de 2019
Medellín

Tabla de contenido

Implementación de solución de Software Plaza Core para gestión empresarial y operativa de aplicaciones omnicanal

1. Introducción.

Las soluciones de software cada vez optimizan y mejoran los procesos de las compañías logrando así un modelo de operación eficiente que se traduce en aumento de los ingresos recibidos por la misma, para lograr este objetivo se busca tener una experiencia única a cada cliente de la organización por lo cual las soluciones de software deben estar preparadas ante esta exigente labor fidelizando al cliente y haciendo que él se sienta importante para la compañía.

A partir de esta premisa se construye una aplicación web enfocada al comercio electrónico la cual es desarrollada sobre una de las aplicaciones líderes en el mercado como lo es Woocommerce la cual cuenta con 32.5% de uso a nivel mundial en este gremio, el objetivo de este proyecto corresponde a la integración de dicha plataforma con un software ERP el cual es utilizado por aproximadamente 600 empresas a nivel nacional. Esta aplicación tiene dependencia de elementos como servidores, servicios web, entre otros, el proyecto de practica cursado esta está enfocado en el desarrollo de dicha aplicación planteando elementos como la gestión técnica y de software, gestión comercial, mantenimiento e implementación de esta.

En el mercado existen múltiples soluciones de Ecommerce las cuales son ajustables a los tamaños de las compañías permitiendo una escalabilidad a nivel de aplicación según el modelo organizacional y los procesos utilizados en la misma, la solución planteada corresponde a una gestión transversal de los macroprocesos estipulados como compras, inventarios, logística, entre otros.

2. Resumen

En este informe se presentan los resultados de la ejecución realizada con el proyecto de autoatención propuesto por CSI SAS donde se dio inicio al desarrollo de aplicaciones de software las cuales corresponden a la planeación estratégica realizada anteriormente, el producto aún no se encuentra completamente terminado pues únicamente se ejecutó la fase I de este para este informe de práctica, en los siguientes capítulos se detallarán las actividades realizadas en esta entrega.

Es completamente necesario contar con una aplicación central la cual permitirá tener control de las soluciones comercializadas a los clientes con el objetivo que la operación pueda gestionar temas como: licenciamiento, bloqueos, activaciones, inactivaciones, controles de cartera entre otros por tal razón se procede con el desarrollo de la arquitectura de software planteada.

El control de las soluciones es parte fundamental para la gestión de cartera y los ingresos de la compañía pues el modelo de distribución es sobre un contrato de alquiler mensual por tal razón todas las soluciones de software deberán estar registradas y autenticadas ante una central la cual tendrá el gobierno y control de estas, debido a esta situación es vital poder gestionar desde una aplicación central todo el control de licenciamiento y se hace necesario este aplicativo antes.

3. Breve Descripción de la Empresa:

3.1. Razón Social.

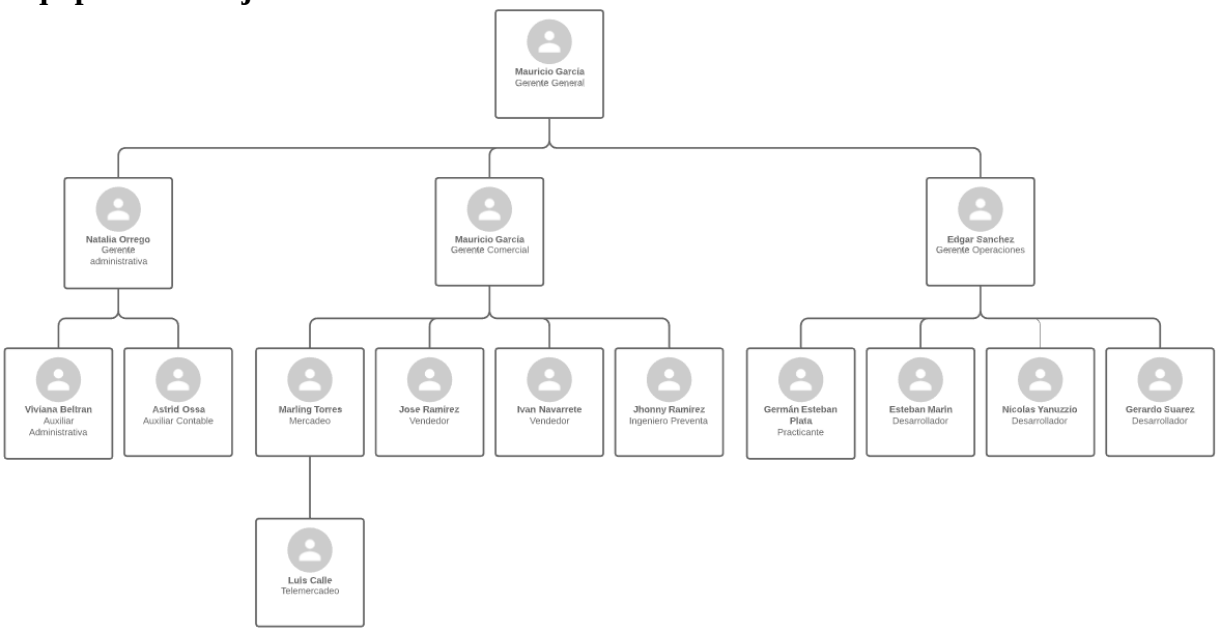
CIS S.A.S NIT 900.092.385-9.

3.2. Objeto Social

C.S.I. S.A.S, constituida en 2003. es una organización reconocida como aliada estratégica en procesos de desarrollo tecnológico, orientada a entregar a sus socios de negocios soluciones que ayuden a agilizar, controlar y administrar las operaciones en las áreas vitales. C.S.I. S.A.S es líder en la prestación de servicios de consultoría y en la integración de productos y servicios que incluyen tecnología de punta, con un excelente servicio al cliente, promoviendo siempre el crecimiento profesional y humano de sus colaboradores.

En C.S.I. S.A.S estamos comprometidos con la satisfacción total de nuestros clientes en áreas de su negocio, contribuyendo al mejoramiento continuo de las operaciones de nuestros Socios de Negocios entregando soluciones de la más alta calidad y con el respaldo de fabricantes de gran reconocimiento a nivel mundial. Llegamos a varias ciudades de y algunos países para entregar la solución que tanto buscas para lograr la eficiencia en tus puntos de venía.

3.3. Equipo de Trabajo



Coordinador: Edgar Sanchez Espinoza

Ingenieros Desarrollo: Nicolas Yanuzzio, Gerardo Suarez, Esteban Marin, Ferney Peña, Germán Esteban Plata.

3.4. Área en la que se desempeñaba

Departamento de técnico y desarrollo de software.

4. Planteamiento del Problema de Investigación y su Justificación en Términos de Necesidades y Pertinencia.

Para las compañías del sector es completamente necesario contar con una aplicación central la cual permitirá tener control de las soluciones comercializadas a los clientes con el objetivo que la operación pueda gestionar temas como: licenciamiento, bloqueos, activaciones, inactivaciones, controles de cartera entre otros por tal razón se procede con el desarrollo de la arquitectura de software planteada.

El proyecto es importante porque responde a diferentes necesidades en el desarrollo de software a nivel corporativo donde se puso en práctica los conocimientos adquiridos en asignaturas como estructuras de datos, ingeniería del software, fundamentos de programación, sistemas operativos entre otros, el estudiante de ingeniería de Telecomunicaciones aplico y comprendió la metodología del desarrollo de software basado en buenas prácticas y tecnologías de punta.

La calidad de software generado permite su uso e implementación en compañías del sector como Comfandi, Juan Valdez, Universidad Javeriana entre otras, permitiendo así llevar a cabo procesos de calidad en procesos como desarrollo, implementación y soporte. Durante el proceso de practica el estudiante se vio en capacidad de afrontar múltiples arquitecturas de software como web, Cliente/servidor, entre otras así mismo afronto retos acordes a requerimientos basado en servicios REST y SOAP.

La empresa CSI S.A.S consciente de la necesidad de software de alta calidad, posee herramientas tecnológicas indispensables en los trabajos de desarrollo, consultoría, requerimientos, implementación

5. Plan de trabajo – Desarrollo práctica profesional

5.1 Objetivos General

Generación de una solución de software central para la gestión de la compañía en sus diferentes canales de venta y procesos operativos, permitiendo una operación centralizada con información de calidad y en tiempo real asó como el control centralizado de cada uno de los mismos

5.2 Objetivos Específicos

- Construcción de base de datos mediante el modelo entidad relación garantizando integridad en la base de datos.
- Desarrollo de solución Plaza Core mediante el uso de programación por capas.
- Desarrollo FullStack de Frontend y backend de la solución.
- Implementación y configuración de Servidor de aplicaciones y servidor web.
- Apoyo en configuración de Firewall.
- Construcción de servicios de integración sistema ERP central el cual es una aplicación de escritorio logrando que tuviera un servicio web de integración.
- Desarrollo de módulo CRM de Plaza Core.
- Implementación de esquemas de seguridad informática a nivel de aplicación y autenticación de usuario.
- Implementación y configuración de servicio web de integración con aplicativo ERP lo cual requirió un ajuste a nivel de Firewall donde se permite el tráfico de entrada y salida sobre protocolos TCP/UDP en un puerto especifico.

5.3 Cronograma de trabajo- Fases/Actividades y sus respectivos tiempos.

Cronograma de Actividades: 09 de junio de 2015 – 09 de diciembre de 2015

- **Desarrollo FrontEnd:** Se desarrollo diseño FrontEnd a partir de Bootstrap 4 donde se utilizó JavaScript para el comportamiento de algunos elementos de la aplicación.

- **Desarrollo del backend:** Se desarrollo backend donde se implementó un CMS para la gestión de aplicaciones las cuales podrán ser operadas de manera centralizada por el equipo de operaciones.

- **Diseño y desarrollo BD Plaza:** Se desarrolla base de datos Plaza bajo el modelo entidad relación aplicando dependencias y normalización de esta.

- **Implementación de servidores:** Se implementan servidores web y de aplicación donde se alojarán aplicativos, estos servidores son virtualizados con esquema de replicación el cual garantizará alta disponibilidad ante cualquier eventualidad.

- **Apoyo en configuración de Firewall:** Se apoya en configuración de Firewall el cual controla y permite la entrada y salida de datos de la aplicación, esto se aplica para protocolos TCP y UDP

- **Desarrollo Servicio Web:** Desarrollo de aplicación tipo web service para la integración con sistema ERP central de la compañía.

- **Desarrollo FullStack:** Desarrollo de Front y back para módulo de gestión comercial CRM en la solución Plaza Core.

- **Implementación seguridad de la información:** Implementación de seguridad perimetral a nivel de servidores de aplicación y servidor web, implementación de algoritmos de seguridad y encriptación para el login de usuario en aplicación Plaza Core.

- **Publicación de Servicio Web:** Publicación de servicio web construido para aplicación central ERP el cual permitió la comunicación en tiempo real mediante protocolo http consumido por direccionamiento IP en un puerto específico del servidor.

- **Seguimientos:** Se realizan reuniones de presentación y seguimiento a CSI SAS donde se compartes, comentan y reciben sugerencias y oportunidades de mejora sobre el proceso en curso.

6. Metodología Implementada

La metodología que se utilizó corresponde a Scrum, donde se planificaron los múltiples sprint de programación los cuales tuvieron un puntaje según la historia de usuario, esto permitió un desarrollo ágil de la solución garantizando los más altos estándares en el código fuente el cual se documentó en cada una de sus clases.

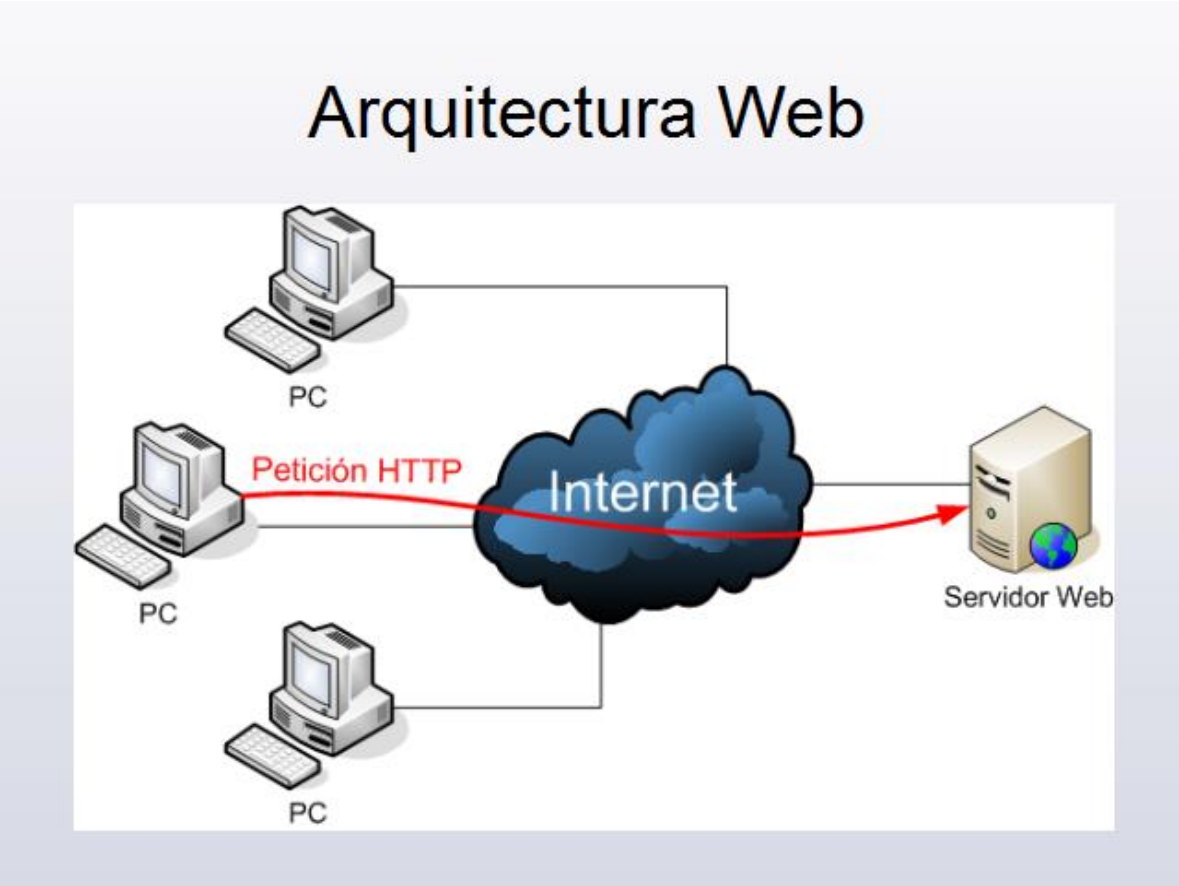
Scrum es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos.

En Scrum se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, Scrum está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales.

Scrum también se utiliza para resolver situaciones en que no se está entregando al cliente lo que necesita, cuando las entregas se alargan demasiado, los costes se disparan o la calidad no es aceptable, cuando se necesita capacidad de reacción ante la competencia, cuando la moral de los equipos es baja y la rotación alta, cuando es necesario identificar y solucionar ineficiencias sistemáticamente o cuando se quiere trabajar utilizando un proceso especializado en el desarrollo de producto.

6.1 Marco Teórico

Se desarrolla aplicación central definida como Plaza Core, dicha solución contará con funcionalidades del tipo, Software, soporte, CRM, entre otros, la aplicación es implementada mediante arquitectura web por lo cual se hace necesario la implementación de un servidor web la cual operará sobre tecnologías Microsoft a continuación se detalla la arquitectura web



El usuario accederá a través de un navegador web donde posterior a un proceso de autenticación está autenticado en la plataforma, cada usuario tiene niveles de seguridad lo cual garantiza el uso de funcionalidades especificas las cuales son determinadas por el administrador del sistema, el desarrollo de la solución Core fue realizado en la parte FrontEnd mediante Bootstrap 4, para el backend se utilizaron las mejores prácticas a nivel de software donde se programó por capas separando capa de acceso a datos, capa de lógica de negocio adicionalmente se implementó un service el cual fue publicado en una API Web la cual permite un control mayo de la solución y un arquitectura escalable por tanto el software podrá soportar nuevas funcionalidades y crecerá de manera controlada, fue necesario la configuración de reglas a nivel de Firewall para así permitir tráfico de datos para la entrada y salida de información de la aplicación adicionalmente con el propósito de proteger el servidor ante cualquier ataque informático, los servidores fueron virtualizados y se implementó una redundancia.

.

6.1.1. Arquitecturas del software, ¿Cómo han evolucionado las arquitecutas hasta hoy?

6.1.1.1 Cliente-Servidor

El modelo Cliente/Servidor es un modelo de aplicación distribuida en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes. Las aplicaciones Clientes realizan peticiones a una o varias aplicaciones Servidores, que deben encontrarse en ejecución para atender dichas demandas.

El modelo Cliente/Servidor permite diversificar el trabajo que realiza cada aplicación, de forma que los Clientes no se sobrecarguen, cosa que ocurriría si ellos mismos desempeñan las funciones que le son proporcionadas de forma directa y transparente. En esta arquitectura la capacidad de proceso está repartida entre los clientes y los servidores, aunque son más importantes las ventajas de tipo organizativo debidas a la centralización de la gestión de la información y la separación de responsabilidades, lo que facilita y clarifica el diseño del sistema. Tanto el Cliente como el Servidor son entidades abstractas que pueden residir en la misma máquina o en máquinas diferentes.

Una disposición muy común son los sistemas multicapa en los que el servidor se descompone en diferentes programas que pueden ser ejecutados por diferentes computadoras aumentando así el grado de distribución del sistema

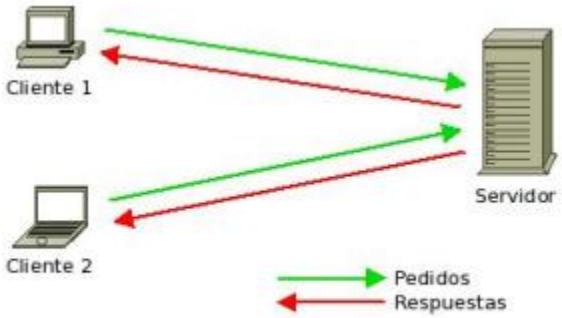


Figura 1. El Modelo Cliente/Servidor.

6.1.1.2 Blackboard (Pizarrón)

Fuentes de Conocimiento

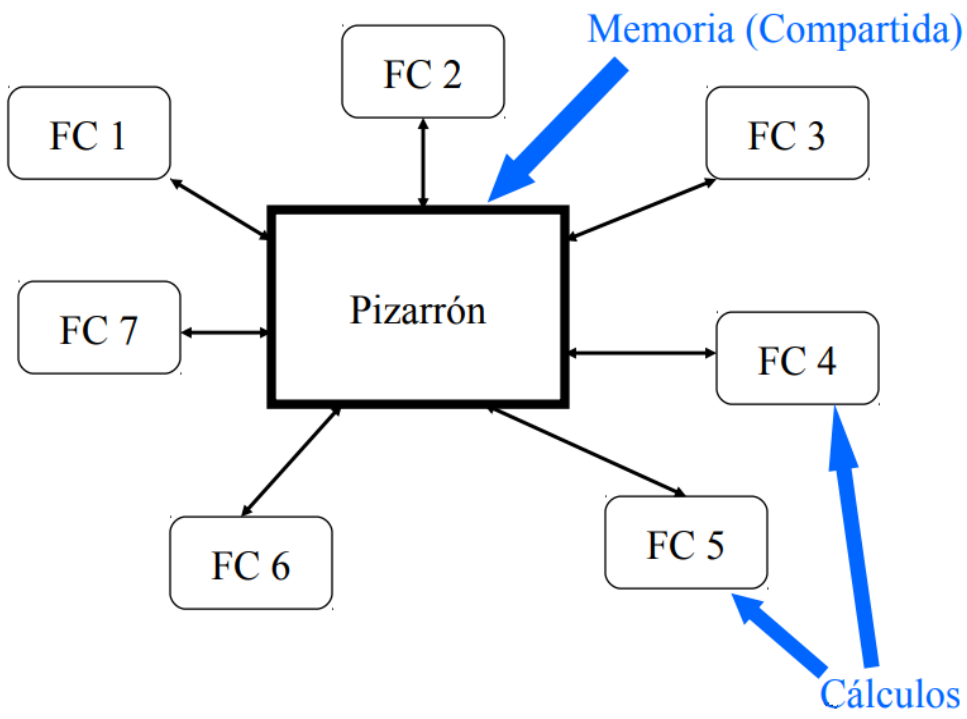
- Procesos independientes que corresponden a particiones del conocimiento del mundo y del dominio dependientes de la aplicación
- Responden a cambios en el pizarrón

Estructura de datos del Pizarrón

- Estado completo de la solución del problema
- único medio por el cual las Fuentes de conocimiento interactúan para llegar a la solución

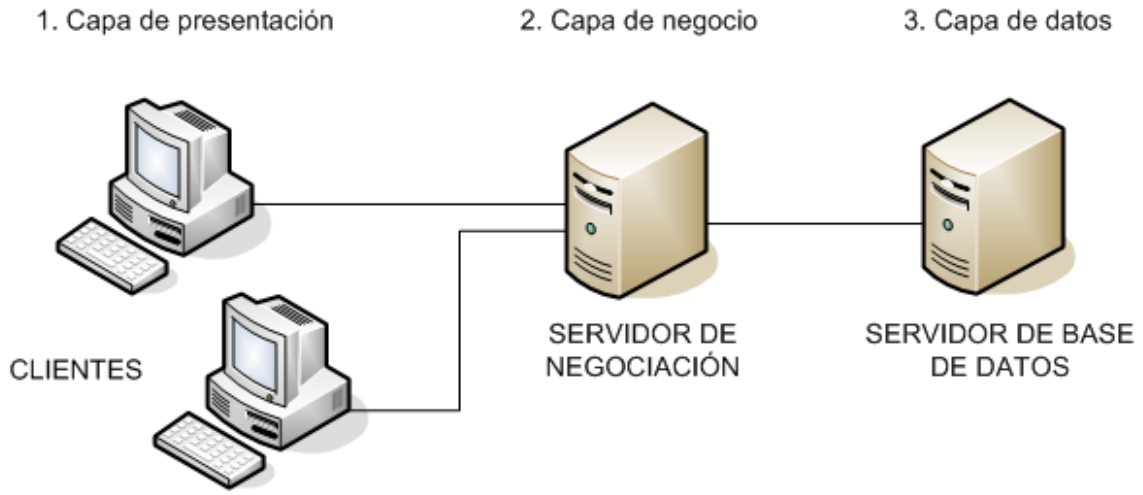
Control

- Guiado enteramente por el estado del pizarrón
- Las Fuentes de conocimiento responden oportunamente cuando los cambios en el pizarrón aplican
- Puede implementarse en las FC, en el pizarrón, en un componente separado o cualquier combinación de éstos.



6.1.1.2.1 Entre capas

La programación por capas es un estilo de programación en el que el objetivo primordial es la separación de la lógica de negocios de la lógica de diseño; un ejemplo básico de esto consiste en separar la capa de datos de la capa de presentación al usuario. La ventaja principal de este estilo es que el desarrollo se puede llevar a cabo en varios niveles y, en caso de que sobrevenga algún cambio, sólo se ataca al nivel requerido sin tener que revisar código entremezclado.



6.1.1.3 Interpretes

Un lenguaje interpretado es un lenguaje de programación que está diseñado para ser ejecutado por medio de un intérprete, en contraste con los lenguajes compilados. Cualquier lenguaje puede ser compilado o interpretado, así que esta denominación es aplicada debido a la práctica de funcionamiento común y no a alguna característica subyacente de un lenguaje en particular. Sin embargo, hay lenguajes que son diseñados para ser en concreto interpretativos, por lo tanto, un compilador causará una carencia de la eficacia. Muchos autores rechazan la clasificación de lenguajes de programación entre interpretados y compilados, considerando que el modo de ejecución del programa escrito en el lenguaje es independiente del propio lenguaje. A ciertos lenguajes interpretados también se les conoce como lenguajes de script.

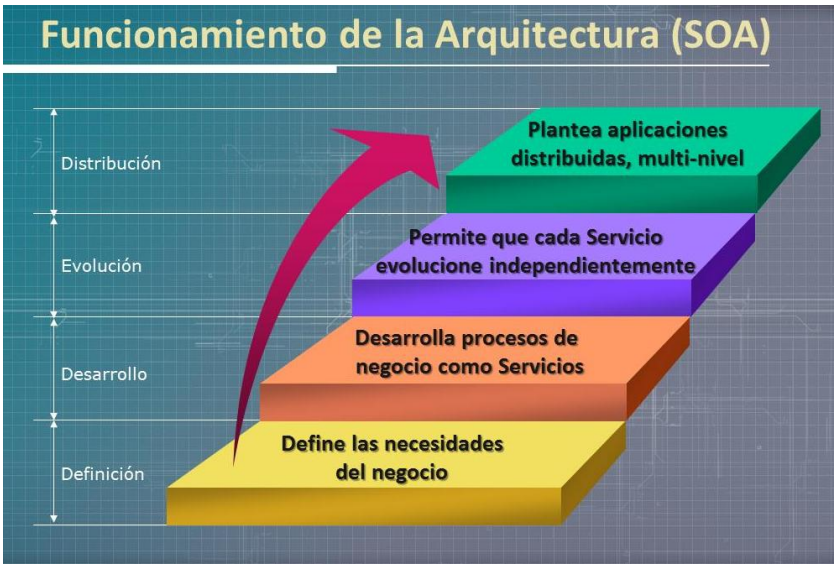
6.1.1.4 Orientada a Servicios

La Arquitectura Orientada a Servicios (SOA en inglés), es un concepto de arquitectura de software que define la utilización de servicios para dar soporte a ciertos requisitos del negocio. Esta arquitectura permite crear sistemas altamente escalables, que pueden ayudar a las organizaciones a impulsar el rendimiento y, al mismo tiempo, reducir costos de TI y mejorar la flexibilidad en los procesos del negocio.

SOA proporciona una metodología y un marco de trabajo para documentar las capacidades de negocio y da soporte a las actividades de integración y consolidación de los datos de cualquier organización.

Iniciativas:

- Servicios de datos escalables y flexibles.
- Servicios de infraestructura para la autentificación, el control de acceso y el registro.
- Servicios CRUD para crear, leer y actualizar datos de todas las formas de sistemas back-end, incluidos datos estructurados, no estructurados, semi-estructurados, de mainframe y en la nube.
- Servicios de integración para ofrecer todas las funciones de integración de datos, como el acceso, el perfilado, la transformación, la calidad y la entrega de datos, así como la federación
- Servicios de metadatos para gestionar y utilizar metadatos técnicos y de negocio para la detección, la auditoría, el linaje y el análisis de impacto



6.1.2. Servidores web

Es un programa que gestiona cualquier aplicación en el lado del servidor realizando conexiones bidireccionales y/o unidireccionales y síncronas o asíncronas con el cliente generando una respuesta en cualquier lenguaje o aplicación en el lado del cliente. El código recibido por el cliente suele ser compilado y ejecutado por un Navegador Web. Para la transmisión de todos estos datos se utiliza algún protocolo. Generalmente se utiliza el protocolo HTTP para estas comunicaciones, perteneciente a la capa de aplicación del Modelo OSI. El término también se emplea para referirse al ordenador que ejecuta el programa.

6.1.2.1 Historia

Los Servicios Web surgieron ante una necesidad de estandarizar la comunicación entre distintas plataformas (PC, Mainframe, Mac, etc.) y lenguajes de programación (PHP, C, Java, etc.). Anteriormente se habían realizado intentos de crear estándares pero fracasaron o no tuvieron el suficiente éxito, algunos de ellos son DCOM y CORBA, por ser dependientes de la implementación del vendedor DCOM – Microsoft, y CORBA – ORB (a pesar que CORBA de múltiples vendedores pueden operar entre sí, hay ciertas limitaciones para aplicaciones de niveles más altos en los cuales se necesite seguridad o administración de transacciones).

Otro gran problema es que se hacía uso de RPC (Remote Procedure Call) para realizar la comunicación entre diferentes nodos. Esto, además de presentar ciertos problemas de seguridad, tiene la desventaja de que su implementación en un ambiente como es Internet es casi imposible (muchos Firewalls bloquean este tipo de mensajes, lo que hace prácticamente imposible a dos computadoras conectadas por Internet comunicarse). Los Servicios Web surgieron para finalmente poder lograr la tan esperada comunicación entre diferentes plataformas. En la actualidad muchos sistemas legacy están pasando a ser servicios web. Es por esto que en 1999 se comenzó a plantear un nuevo estándar, el cual terminaría utilizando XML, SOAP, WSDL, y UDDI.

A pesar de mucho limitar el uso de los servicios Web al Protocolo http, los servicios Web no fueron pensados para un protocolo en particular, es decir, nada impidió utilizar SOAP sobre algún otro protocolo de Internet (SMTP, FTP, etc.). Se utiliza principalmente HTTP por ser un protocolo ampliamente difundido y que se encuentra menos restringido por firewalls

(generalmente se bloquean puertos como el FTP, pero el HTTP es muy probable que no este bloqueado). La década de los 80's fue marcada por el surgimiento de la PC y de la interfase gráfica. Entre 1988 y 1993, NeXT fabricó una estación de trabajo de altas prestaciones para la época de la que hablamos que impulsó con el Sistema operativo de la casa, el NeXTSTEP. Contaba con un micro de la serie (68040) de Motorota capaz de trabajar a 25 MHz, una memoria de 8 MB ampliables a 64 MB y un monitor de 17. La máquina, que costaba 6500 dólares, recibió el nombre de NeXT Computer, aunque se le acabó conociendo como NeXTcube o, simplemente, "The Cube". Más allá de su Hardware y sus posibilidades técnicas, la NeXT Computer ha pasado a formar parte de la pequeña gran historia de la informática por ser el ordenador que Tim Berners-Lee, el "inventor" de Internet, se utilizó por primera vez como un Servidor Web.

En la década de los 90's Internet permitió conectar computadoras en una escala global. En principio la conexión fue entre PCs y servidores por medio del explorador de Internet. A comienzos de este siglo es clara la necesidad de permitir a las computadoras conectadas a Internet comunicarse entre ellas. Desde entonces se va dando forma al nuevo modelo de computación distribuida llamado servicios Web basados en XML. El objetivo es permitir comunicarse entre sí a sistemas heterogéneos dentro y fuera de la empresa. Esta comunicación es independiente del Sistema Operativo, lenguaje o modelo de programación. Para conseguir esto se desarrollaron estándares. El consorcio de Internet http://www.w3c.org fue el encargado de crear y mantener estos estándares.

Desde los inicios de Internet, fueron surgiendo diferentes demandas por los usuarios y se dieron soluciones mediante lenguajes estáticos. A medida que paso el tiempo, las tecnologías fueron desarrollándose y surgieron nuevos problemas a dar solución. Esto dio lugar a desarrollar lenguajes de programación para la Web dinámicos, que permitieran interactuar con los usuarios y utilizaran sistemas de Bases de Datos.

Antes de la adopción del modelo de Servicios Web basados en XML los datos eran “islas” que se encontraban dentro de las aplicaciones. Era muy difícil y costoso implementar soluciones para acceder a la información desde afuera de la aplicación. Las aplicaciones

pueden ahora, comunicarse entre sí y con los sistemas de sus socios, proveedores y clientes gracias a los Servicios Web y XML.

6.1.2.2 Protocolos del Servidor Web

Un servidor web es un programa que sirve datos en forma de Páginas Web, hipertextos o páginas HTML (HyperText Markup Language): textos complejos con enlaces, figuras, formularios, botones y objetos incrustados como animaciones o reproductores de sonidos. La comunicación de estos datos entre cliente y servidor se hace por medio un protocolo, concretamente del protocolo Http. Con esto, un servidor Web se mantiene a la espera de peticiones HTTP, que son ejecutadas por un cliente HTTP; lo que solemos conocer como un Navegador Web. A modo de ejemplo: al teclear (http://www.cnice.mec.es) en un navegador, éste realizará una petición HTTP al servidor que tiene asociada dicha URL.

El servidor responde al cliente enviando el código HTML de la página; el navegador cuando recibe el código, lo interpreta y lo muestra en pantalla. El Cliente es el encargado de interpretar el código HTML, es decir, de mostrar las fuentes, los colores y la disposición de los textos y objetos de la página. El servidor se encarga de transferir el código de la página sin llevar a cabo ninguna interpretación de esta.

Servidor Web Local

Tener un servidor Web local debería ser fundamental para todas aquellas personas que no disponen de un Sitio Web online. ¿Por qué? La respuesta es muy sencilla. El Servidor local nos va a permitir comprobar que todas las modificaciones que se van a realizar en nuestro diseño Web, no provoquen algún error que pueda afectar a todos aquellos usuarios que naveguen por la red. Otro aspecto positivo de un Servidor local es que no hará falta tener que subir ficheros al servidor Ftp para hacer nuestras pruebas con lo cual, la espera se hace más corta.

Como es de suponer, también se permite el acceso a nuestro servidor a cualquier usuario de Internet. Para ello, se configura correctamente el Router y sería de mucha ayuda tener una dirección IP estática, mediante la cual cualquier usuario podría conectarse a nuestro servidor desde un navegador.

Protocolo: Conjunto de reglas que gobiernan el intercambio de datos entre entidades dentro de una red. Es el lenguaje común “que utilizan” los ordenadores para “hablar” y entenderse entre sí. Existen muchos tipos de protocolos cada uno con sus reglas bien definidas, como, por ejemplo: FTP, POP3, SMTP, ICMP, etc.

Protocolo HTTP: Una de las características del Protocolo Http es que no es permanente, es decir, cada operación HTTP implica una conexión con el servidor, que es liberada al término de la misma. Por ejemplo, un documento HTML con 10 imágenes son necesarias 11 conexiones distintas (10 imágenes más la página HTML en sí).

Servidores de aplicaciones

Un Servidor de Aplicaciones no es más que un cambio de nombre, para algunos Servidores Web de nueva generación que proporcionan la lógica de negocio sobre la que construir aplicaciones. Suelen asociarse con servidores de alto rendimiento pensados para dar servicio a sitios Web (Web Sites) con grandes necesidades: afluencia de visitas, movimiento de datos, atención de transacciones hacia bases de datos, etc. Generalmente los fabricantes del sector tienen a disposición del público un servidor Web básico y otro con multitud de extensiones fuertemente integradas al que llaman Servidor de Aplicaciones.

6.1.2.2.1. Funcionamiento de un Servidor Web

La Web funciona siguiendo el Modelo cliente-servidor. Un Servidor se encarga de prestar el servicio, y un cliente que es quien recibe dicho servicio.

Cliente Web: Es un programa mediante el cual el usuario solicita a un Servidores Web el envío de información. Esta información se transfiere mediante el Protocolo HTTP. Información que recibe: La información que se recibe es un conjunto de documentos de texto codificados en lenguaje HTML.

El Cliente Web debe interpretar estos documentos para mostrárselos al usuario en el formato correspondiente. Cuando la información recibida no es un documento de texto, sino un objeto multimedia que el cliente no sabe interpretar, el propio cliente Web debe activar una aplicación externa encargada de gestionarlo. Clientes Web más habituales: Los clientes Web más habituales son Microsoft Internet Explorer, Mozilla Firefox y Netscape Navigator.

6.1.2.3. Arquitectura Modelo Cliente – Servidor

Diversas aplicaciones se ejecutan en un entorno Cliente/servidor. Esto significa que los equipos clientes (equipos que forman parte de una red) contactan a un servidor, un equipo generalmente muy potente en materia de capacidad de entrada/salida, que proporciona servicios a los equipos clientes. Estos servicios son programas que proporcionan datos como la hora, archivos, una conexión, etc.

Los servicios son utilizados por programas denominados programas clientes que se ejecutan en equipos clientes. Por eso se utiliza el término "cliente" (cliente FTP, cliente de correo electrónico, etc.) cuando un programa que se ha diseñado para ejecutarse en un equipo cliente, capaz de procesar los datos recibidos de un servidor (en el caso del cliente FTP se trata de archivos, mientras que para el cliente de correo electrónico se trata de correo electrónico).

En la manera de describir la forma de trabajo entre los clientes y los ordenadores se define: Cliente: Es el ordenador que pide información a otro, mediante la aplicación de un programa llamado cliente. Este contacta con el servidor y Da formato a la petición de la información y da formato a la respuesta.

Servidor: Es el ordenador que ofrece la información mediante la aplicación de un programa llamado servidor que: Recibe la información y la procesa y responde enviando la petición al cliente.

El servidor Web se ejecuta en un ordenador manteniéndose a la espera de peticiones por parte de un cliente (un navegador Web) y que responde a estas peticiones, mediante una Página Web que se exhibirá en el navegador o mostrando el respectivo mensaje si se detectó algún error. A modo de ejemplo, al escribir la siguiente dirección en el navegador, [www.ecured.cu], éste realiza una petición al servidor de dicha dirección. El servidor responde al cliente enviando el código HTML de la página; el cliente, una vez recibido el código, lo interpreta y lo muestra en pantalla. Como observamos en este ejemplo, el cliente es el encargado de compilar y ejecutar el código HTML, es decir, de mostrar las fuentes, los colores y la disposición de los textos y objetos de la página Web; el servidor tan sólo se limita a transferir el código de la página sin realizar ninguna interpretación de la misma. Además de la transferencia de código HTML, los servidores Web pueden entregar aplicaciones Web. Éstas son bloques de código que se ejecutan cuando se realizan ciertas peticiones o respuestas. Se distinguen entre:

- **Aplicaciones en el lado del cliente:** el cliente Web es el encargado de ejecutarlas en la máquina del usuario. Son las aplicaciones tipo Java "applets" o Javascript. El servidor proporciona el código de las aplicaciones al cliente y éste las ejecuta mediante el navegador Web. Por tanto, es necesario que el cliente disponga de un navegador con capacidad para ejecutar aplicaciones (también llamadas Scripts). Generalmente, los navegadores permiten ejecutar aplicaciones escritas en lenguaje Javascript y Java, aunque pueden añadirse más lenguajes mediante el uso de plugins.

- **Aplicaciones en el lado del servidor:** el servidor Web ejecuta la aplicación; ésta, una vez ejecutada, genera código HTML; el servidor envía al cliente este código recién creado por medio del protocolo HTTP.

Las aplicaciones en el lado del servidor mayormente suelen ser la mejor opción para desarrollar aplicaciones Web. La razón es que, al ejecutarse ésta en el servidor y no en la máquina del cliente, éste último no necesita ninguna capacidad añadida para ejecutar la aplicación, como sí ocurre en el caso de querer ejecutar aplicaciones que incluyan scripts con javascript o java. Así pues, cualquier cliente que disponga de un navegador Web básico puede utilizar este tipo de aplicaciones.

6.1.2.4. Aplicaciones en el lado del servidor

Una aplicación en el lado del servidor es cualquier programa o conjunto de instrucciones diseñadas con la finalidad de que un servidor Web las procese para realizar alguna acción. Las aplicaciones del lado del servidor están escritas mediante un lenguaje de programación, entre los que más se utilizan están los siguientes:

- PHP
- ASP
- Perl
- Python
- Ruby

Servidores basados en procesos

Este diseño es el predecesor de todos los demás. Se basa en la obtención de paralelismo mediante la duplicación del proceso de ejecución. Existen varios diseños basados en procesos. El más simple es en el que el proceso principal espera la llegada de una nueva conexión y en ese momento, se duplica creando una copia exacta que atenderá esta

conexión. Sobre esta opción de diseño caben optimizaciones importantes, como las que incluyó Apache con la técnica de Pre-fork.

Técnica pre-fork: Consiste en la creación previa de un grupo de procesos y su mantenimiento hasta que sea necesaria su utilización. Las principales ventajas de este diseño residen en su simplicidad de implementación y su seguridad. La gran desventaja de este diseño es el bajo rendimiento. La creación o eliminación de un proceso son tareas pesadas para el sistema operativo y consumen una gran cantidad de tiempo.

Servidores basados en hilos

Este tipo de diseño hoy en día es mucho más común que el basado en procesos. Los conceptos básicos respecto al funcionamiento de un servidor basado en procesos son aplicables también a este modelo. Las principales diferencias de los dos modelos residen en el propio concepto de hilo. La ventaja es que la creación de un hilo no es tan costosa como la de un proceso. Varios hilos de un mismo proceso pueden compartir datos entre ellos, ya que comparten el mismo espacio de memoria.

El modelo de servidor basado en hilos hereda muchas de las características de los Servidores basados en procesos, entre ellas la de la simplicidad en su diseño e implementación. Por otro lado, el compartir el espacio de memoria implica un riesgo de seguridad que no tienen los servidores basado en procesos.

Hilos y procesos.Proceso: Es una ocurrencia o instancia de un programa en ejecución. Además, un proceso es propietario de una serie de recursos como: un espacio de direcciones en memoria, ficheros, hilos, etc.

Hilo: Es un proceso totalmente aislado es un proceso inerte, es decir, para que un proceso sea capaz de hacer algo, el proceso debe ser propietario de al menos un hilo (thread). El hilo es el responsable de ejecutar el Código contenido en el espacio de direcciones del proceso. De hecho, un proceso puede contener varios hilos y todos ellos ejecutando código "simultáneamente" en el espacio de direcciones del proceso y compartiendo recursos comunes.

Al compartir todos los hilos de un proceso la misma zona de memoria, si un hilo toca una variable, todos los demás hilos del mismo proceso verán el nuevo valor de la variable. Si no hay hilos ejecutando código en el espacio de direcciones del proceso no hay ninguna razón para que el proceso continúe existiendo y el sistema destruirá automáticamente el proceso y su espacio en memoria.

Servidores basado en sockets no bloqueantes o dirigidos por eventos

Estos servidores basan su funcionamiento en la utilización de lecturas y escrituras asíncronas sobre Sockets. Normalmente, estos servidores utilizan una llamada al sistema que examine el estado de los sockets con los que trabaja. Cada sistema operativo implementa una o más funciones de examen de sockets. El objetivo de estas funciones es inspeccionar el estado de un grupo de sockets asociados a cada una de las conexiones.

- La ventaja de este diseño es principalmente su velocidad.
- La desventaja es que la concurrencia es simulada; es decir, existe un sólo proceso y un sólo hilo, desde el cual se atienden todas las conexiones.

Socket: No son más que puntos o medios de comunicación entre dos aplicaciones que permiten que un proceso hable (emita o reciba información) con otro proceso estando los dos en distintas máquinas. Lo vemos mejor con un dibujo:

Si extrapolamos el concepto a la comunicación entre personas, un socket es al sistema de comunicación entre ordenadores lo que un teléfono es al sistema de comunicación entre personas: un punto de comunicación entre dos agentes (procesos o personas respectivamente) por el cual se puede emitir o recibir información.

Servidores implementados en el kernel

Este diseño es un poco especial. Se trata de un intento de acelerar la velocidad de un servidor Web mediante el movimiento de su código de espacio de usuario a espacio de kernel. En teoría este modelo se muestra muy eficiente, pero de cara al mundo real, los problemas e inconvenientes son muy grandes. Hay que tener en cuenta que cualquier problema que se produzca a nivel de kernel puede ocasionar la caída de todo el sistema completo.

6.2.1. Procesos a Ejecutar

6.2.1.1. Diseño y maqueteado de la solución.

El Product Owner desarrolla todo el maquetado de la solución y sus módulos donde define la historia de uso a desarrollar con sus respectivos caso, parámetros y condiciones así mismo

se involucra el analista de bases de datos donde alineado a los requerimientos e historias del Producto Owner se realiza el diseño del modelo de base de datos.

Responsables: Producto Owner - Estudiante de Ingeniería de Telecomunicaciones en práctica empresarial.

Documentación Requerida: Historias de usuario, cronograma de trabajo

6.2.1.2 Sprint

Una vez planteada la historia de usuario, se realiza el sprint de desarrollo donde es prioridad la entrega de un MVP es importante tener claro en este sprint

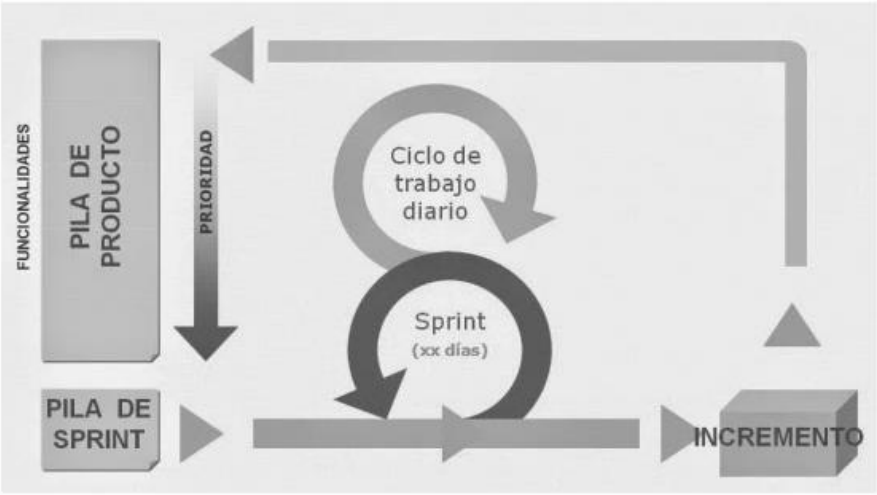
- Las iteraciones que se tendrán.
- El entregable que debe realizarse.

Responsables: Estudiante de Ingeniería de Telecomunicaciones en práctica empresarial.

Documentación Requerida: Historia de usuario y manual de usuario.

6.2.1.3 Ciclo de trabajo del sprint

Se realiza el desarrollo del sprint donde se debe cumplir el siguiente flujo.



Responsables: Coordinador – Scrum Manager - Estudiante de Ingeniería de Telecomunicaciones en práctica empresarial

Documentación Requerida: n/a.

7. Resultados

Con la participación del estudiante de practica se logra generar la primera versión de aplicativo Plaza Core cumpliendo con cada uno de los puntos anteriormente citados logrando de esta manera resultados como:

- Se dejaron operativos los servidores web vitualizados para Server de aplicaciones, Server de publicación y Server de base de datos.
- Se implementa control centralizado de las soluciones a implementar posteriormente.
- Se puso en marcha la versión 1.0.1 del aplicativo Plaza Core.
- Integración en tiempo real con la parte comercial de sistema ERP de CSI SAS
- Se implementan criterios de seguridad y seguridad perimetral en servidores.

El scope del proyecto está definido a 1 año por lo cual en esta primera etapa el apoyo del estudiante de telecomunicaciones en temas de infraestructura y comunicaciones fue de alto impacto en la compañía pues aporto e implemento conocimientos en seguridad informática mediante la implementación de un esquema de seguridad perimetral para el acceso a servidores así como las respectivas reglas de configuración para los firewall protegiendo los servidores pero permitiendo que estos operen acorde al lineamiento y desarrollo del mismo.

A nivel de desarrollo de software se construye una arquitectura de aplicación robusta y escalable la cual permitirá el crecimiento controlado de la solución en múltiples tecnologías pues la interconexión entre sistemas permite la integración de aplicaciones de software en diferentes sistemas operativos como Android, Windows y MAC permitiendo el control central de la solución.

La orientación del producto owner en el proyecto le ayudo al estudiante a afianzar sus conocimientos en planificación y ejecución del desarrollo de software lo cual se traduce en aplicaciones estables y confiables.

8. Conclusiones

- Se logro construir una base sólida para la escalabilidad de la solución de software a futuro.

- Con las buenas prácticas, se logró una versión beta del producto escalable con una construcción robusta y que podrá soportar múltiples años

- Se continuará con el desarrollo de la aplicación pues esta primera beta se pondrá en pruebas por parte de la operación quien detallará novedades y/o requerimientos adicionales.

- El aporte destacado desde el punto técnico e ingenieril del estudiante a la empresa está dado principalmente en la fusión de los conocimientos teóricos y prácticos (estos últimos aprendidos con rapidez en la misma práctica empresarial)

- La comprensión de las arquitecturas de software y la metodología scrum facilito llevar a cabo las diferentes actividades de una manera mas eficiente.

- En la ejecución de la práctica empresarial fortalecí las relaciones interpersonales al llegar a acuerdos en trabajo de equipo uniendo teoría y práctica en este caso particular en el área de mediciones para alcanzar el éxito en cada uno de las ordenes de trabajo solicitadas por los clientes, hecho que posibilitó también el desarrollo de competencias técnicas y tecnológicas.

9. Referencias

https://www.academia.edu/9357315/Arquitectura_basada_en_capas

https://www.ecured.cu/Servidor_Web#Protocolos_del_Servidor_Web

https://itm201530.webnode.es/archivos-del-sistema/lenguajes/interpretes/

https://www.ecured.cu/Arquitectura_en_Capas