Universidad de las Ciencias Informáticas
Facultad 8

Libro de Firmas

Sistema de Control de Presencia (SISCOP)

Trabajo de diploma para optar por el título
de Ingeniero en Ciencias Informáticas

Autores: Lisbet Torres Triana
Yoan Trujillo Reyna

Tutor: Ing. Yanio García Vidal
Co-Tutora: Ing. Celia Maria Soulary Reyes

Ciudad de La Habana, junio del 2010
"Año 52 de la Revolución"

"Todo nuestro trabajo debe estar orientado a lograr que la tarea administrativa, de control y dirección, se vaya convirtiendo en algo cada vez más simple..."

(La banca, el crédito y el socialismo, Ernesto Guevara. Marzo de 1964)

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores del trabajo Libro de Firmas y autorizamos a la Facultad 8 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

Autores:

_____

Lisbet Torres Triana

_____

Yoan Trujillo Reyna

Tutores:

_____

Ing. Celia María Soulary Reyes

_____

Ing. Yanio García Vidal

Dedicatoria

A nuestros padres, por brindarnos siempre amor y comprensión. Por no faltarnos nunca su apoyo, a ustedes atribuimos este éxito.

Agradecimientos Lisbet

A mi ***mamá***, gracias por tu amor y tu gota de cariño día a día. Te amo Kika.

A mi ***papá*** por dedicarme su amor y apoyarme siempre… te quiero.

A mis ***abuelos***, gracias por su amor y apoyo sin límites.

A mi tía ***Marlín*** y mi tío ***Dago***, gracias por estar pendientes de mí y ayudarme siempre.

A mis primos, ***Yilén***, ***Ramiro*** y ***Obadis***, por ayudarme y apoyarme.

A mi ***novio***, por ser el mejor guía en mi carrera, gracias por tu apoyo incondicional, por estar siempre a mi lado… te amo puchungo.

A mi hermanita ***Yeni*** y mi tía ***Donaida***, gracias por estar al tanto de mí, por apoyarme y ayudarme tanto.

A mi amiga ***Yenni***, titi gracias por estar siempre a mi lado, aún estando lejos.

A ***Yoan*****,** que es mi hermano, sin ti no hubiese podido llegar hasta aquí.

A ***Adrian***, mi amigo incondicional por siempre. Te quiero negro.

A mi amigo ***Andy*** por aconsejarme, ayudarme y tenerme presente, te quiero Andillito.

A mi amigo ***Mario***, gracias por cuidar de mí, por ayudarme y apoyarme.

A ***Leidy*** y ***Leisy***, por estar conmigo en buenas y malas. Por hacerme reír siempre.

A ***Yanly*** y ***Daileny***, por ser mis compañeras en todo momento, las quiero.

A ***Daily, Andy Abalarde***, ***Ailen***, ***Osbel*** y mis otros compañeros del ***8108*** por estar siempre dispuestos ayudarme y brindarme su amistad.

A mi azabache ***Yalida***, gracias por ser mí amiga.

A ***Marlon***, ***Julio***, ***Albin***, ***Addiel*** y ***Oigres***, gracias por ayudarme y apoyarme siempre.

A mis ***suegros*** y mi ***cuñada***, por estar pendientes de mis pasos, por su confianza y su apoyo.

A ***Juan Carlos***, impulsor de este sueño hecho realidad.

A ***Lucía***, ***Maidel*** y ***Amarilis***, por estar al tanto de mis pasos en la universidad.

A los ***tutores***, por su esfuerzo, apoyo y paciencia. Gracias de todo corazón.

A los profes ***Dania***, ***Marianne*** y ***Vislet***, por contribuir a mi formación como ingeniera.

A ***Yunaisy*** y ***Yadelis***, gracias por la ayuda para la culminación de este trabajo.

A ***Dios***, por darme la oportunidad de vivir y por no abandonarme nunca.

Agradecimientos Yoan

A mis ***padres*** por formarme y guiarme todo este tiempo para enfrentar todos los retos de la vida.

A mi hermano ***Yoimi*** por ser el mejor hermano mayor y siempre ser mi guardián, tanto de los estudios como en la vida cotidiana.

A mis ***abuelos*** por estar siempre pendientes de mí y preocuparse por mi alimentación gracias a ellos hoy tengo este pequeño cuerpo.

A mis tíos, ***Mayda, Camilo, Cesy*** y ***Eduardo***.

A mis primos ***Deglis, Ariel, Aylin, Eliane, Daniel, Cecilin*** y ***Julito***.

Mis vecinos ***Norma, Librada*** y ***Guachi*** por ser una extensión de la familia Trujillo Reyna.

A mi familia de Santa Clara Alfredo, Nancy, Abbie, Iana, Juan Pablo (El Gato), Lina, Marilín y Luisito.

A mi esposa ***Celia*** por estar siempre a mi lado y prestarme todo su apoyo año por año en la universidad.

A mi hermanita ***Lisbet*** por soportar mis berrinches y mis pesadeces durante estos 5 años y soportar siempre el cartelito que le puse en la biblioteca.

A ***Greisy*** y ***Yudisbel*** por su sabios consejos siempre oportunos.

A ***Raúl*** mi hermano desde la secundaria, por todo este tiempo de amistad y sus siempre oportunos viajes a Matanzas.

A ***Mario*** por siempre estar ahí para poder dejarle caer un pequeño chistecito.

A ***Irene*** primero que todo por haberme dado una hermanita y segundo por sus ricos limones.

A ***Rosi*** por sus buenas atenciones.

A mi ***suegra*** por su constante preocupación por mis resultados.

A los ***tutores*** por su gran apoyo durante está difícil etapa.

A la profesora ***Dania Reyna*** que me ayudó mucho durante ese siempre difícil primer año.

A la negrita ***Yalida*** por ser tan alegre y reírte de mis chistes, saludo a las gemelas.

A todos mis compañeros del ***grupo 8108*** que aunque nos separaran en tercer año siempre siguieron apoyándome y preocupándose por mis resultados académicos.

Resumen

En la actualidad son muchas las entidades a nivel mundial que necesitan mantener el control del cumplimiento de la jornada laboral de sus trabajadores. Esto contribuye a que se realice el pago salarial correspondiente a cada trabajador y del mismo modo les permite a los directivos de la entidad mantener un mejor registro sobre las horas de producción de los mismos.

Durante el desarrollo de la presente investigación se realizó el estudio de sistemas para el control de presencia en el ámbito nacional e internacional. Se justificaron además las tecnologías, herramientas, lenguajes y metodologías que fueron utilizadas para el desarrollo de un sistema de este tipo. Por otro lado, se especificaron los módulos desarrollados para obtener las funcionalidades requeridas que permitan el correcto funcionamiento del software.

Como resultado de este trabajo de diploma con título “Libro de Firmas” se desarrolló una aplicación web, que garantiza el control de la jornada laboral de los trabajadores y que además brinda servicios al portal de la intranet de la Universidad de las Ciencias Informáticas y al Sistema de Planificación y Control de Alimentos de la misma entidad. El mismo cuenta con características apropiadas para ser desplegado en cualquier institución.

Palabras clave: jornada laboral, control de presencia

Índice

Índice de tablas

Índice de imágenes

Introducción

Desde el surgimiento de la computación hasta la actualidad el mundo de la informática ha crecido vertiginosamente. La automatización de los procesos y actividades que realiza el hombre diariamente ha estado a la par de este desarrollo, un ejemplo es la informatización de los procesos de control de acceso y presencia, es decir, del control del horario laboral.

La Universidad de las Ciencias Informáticas (en lo adelante UCI) está conformada por una estructura compleja y extensa, con funciones diversas. La misma está compuesta por un conjunto de áreas administrativas, las cuales tienen su propio régimen de horarios y sistema de turnos de trabajo, ejemplo de esta diversidad se puede ilustrar analizando dos áreas, las cuales pueden ser Complejo Comedor 1 y Facultad 8, donde podemos encontrar tres tipos de horarios: horario fijo, abierto y por turnos. Del mismo modo, espacialmente la universidad ocupa un área geográfica extensa, con las áreas administrativas diseminadas por la misma. Esto dificulta el control de la jornada laboral de todos los trabajadores, haciéndose más compleja la contabilización de las horas trabajadas de cada uno de estos en el Departamento de Recursos Humanos, para el posterior pago salarial.

En la universidad existen sistemas para el control de acceso a laboratorios y algunos proyectos productivos han implementado sus propios sistemas de control, estos constituyen soluciones para problemas similares al control de entrada/salida de los trabajadores, pero carecen del enfoque de Gestión de Recursos Humanos amparado en las legislaciones vigentes y no contemplan el estudio de procesos de la Gestión de Recursos Humanos necesaria. Existe además en la Universidad una potente red de computadores y servicios que se extiende a toda su geografía la cual puede ser utilizada para distintos fines, de acuerdo con el objeto social del centro.

Debido a la situación problémica planteada surge el siguiente ***problema***: ¿Cómo controlar el cumplimiento del horario laboral de los trabajadores de la UCI?

Con el fin de solucionar el problema planteado se tiene como ***objetivo general*** desarrollar una solución informática que respalde el control del horario laboral de la UCI.

Se determinó como ***objeto de estudio:*** el proceso de informatización del sistema de control del horario laboral.

Para la presente investigación se trazaron los siguientes ***objetivos específicos:***

1. Investigar sobre la informatización del proceso de entrada y salida laboral.

2. Caracterizar el proceso de control de entrada/salida laboral de la UCI.
3. Elaborar el diseño y desarrollo de la gestión de horarios laborales de las áreas administrativas de la UCI, de conjunto con los servicios a ofrecer a la Universidad.

Se reconoce además como ***campo de acción*** el proceso de informatización de un sistema para el control del horario laboral en la UCI y se aspira a obtener como ***posible resultado*** una solución informática que controle el cumplimiento del horario laboral y brinde servicios al Portal Intranet 2.0 y al Sistema de Planificación y Control de Alimentos.

El documento está estructurado de la siguiente forma:

Capítulo I: Fundamentación Teórica: En este Capítulo se realizará el estado del arte de los sistemas de control de acceso y presencia. Se exponen los conceptos control de acceso y control de presencia, y otras definiciones relacionadas con el tema de investigación. Se describen los lenguajes, las herramientas y metodologías a utilizar para el desarrollo de la aplicación.

Capítulo II: Características del Sistema: En este Capítulo se realizará una descripción de las características del sistema a implementar. Se describe el proceso de negocio y las Historias de Usuarios (en lo adelante HU), además se especifican los requisitos funcionales y no funcionales.

Capítulo III: Implementación: Se desarrolla la Implementación, utilizando la metodología seleccionada y documentándose en este el modelo de datos, el modelo de despliegue y las tareas de ingeniería.

Capítulo I: Fundamentación Teórica.

1.1 Introducción

Las tecnologías en la actualidad están en constante evolución, esto trae consigo que rápidamente se pongan obsoletas para el desarrollo óptimo de aplicaciones informáticas. Teniendo en cuenta lo antes planteado, el presente capítulo está destinado a realizar una descripción de los Sistemas de Control de Presencia existentes en el mundo, en nuestro país y en especial en la Universidad. Además para poder realizar la solución informática con la calidad requerida, se han investigado las metodologías, herramientas, lenguajes y tecnologías para el desarrollo de la misma.

1.2 Sistemas de Control de Presencia

Alrededor del año 1906 el ingeniero francés Charles Eugene Bedaus introdujo por primera vez en la sociedad moderna el término de “salarios”, [1] desde entonces el hombre se ha visto en la necesidad de controlar el tiempo de trabajo de sus empleados con el objetivo de mantener un control sobre la producción y lo aportado por cada trabajador. Con el paso de los años se hace cada vez más compleja las formas de pagos y en conjunto con estas también han evolucionado los métodos de control de la jornada laboral, surgiendo términos como **control de acceso** y **control de presencia**.

Se acostumbra usualmente confundir los términos **control de acceso** y **control de presencia**. **El control de acceso** está basado en limitar los movimientos del personal de la empresa o externos a las diferentes áreas de la instalación, el acceso puede estar clasificado según la categoría laboral, departamentos, individualizado por empleados, por fechas, por horarios, entre otras, según los intereses y restricciones que desee imponer la administración del local.

Por otro lado el **control de presencia** consiste en controlar la presencia o el absentismo de los empleados, no solo se controla el tiempo de trabajo, sino también las incidencias permitidas, no permitidas y las disponibles para el disfrute del trabajador en el horario laboral. Para poder tener un cómputo de las horas trabajadas es necesario definir un calendario de trabajo asociado con la definición de horarios a realizar, además de planificar las vacaciones del personal en el calendario laboral. Estos horarios son los que llevan los parámetros necesarios para determinar si el tiempo presencial hay que contabilizarlo y contra qué tiempo de horas debe aplicarse. Los tipos de horas del trabajador son definidos por cada

empresa según los intereses de estas, mayormente se utilizan los términos de hora normal, hora festiva, hora extra, entre otros. Para obtener los marcajes de presencia de los empleados se debe tener una terminal de control de presencia (Sistemas de control de Huellas, Sistemas lectores de códigos barras, Sistemas lectores barras magnéticas, etc.) para que el trabajador pueda ser identificado y quede almacenado su hora de entrada/salida e información de ubicación. Para lograr esto, el empleado debe fichar con rigor la entrada y salida en los controladores adecuados. [2]

Los Sistemas de Control de Presencia (en lo adelante SCP) son un conjunto de elementos que permiten gestionar y medir la presencia de los empleados de una empresa, estos elementos son los siguientes:

1. **Relojes de fichar:** Estos terminales gestionan que sólo los usuarios registrados tengan acceso al sistema y además que lo hagan respetando un calendario y horario.
2. **Soporte de identificación:** Son los elementos que permiten identificar al empleado o usuario ante el SCP. Estos elementos fueron evolucionando en conjunto con los relojes, primero fueron tarjetas de cartón, después tarjetas magnéticas, tarjetas con códigos de barras, tarjetas de proximidad, tarjetas chips y en la actualidad lo que se está desarrollando es la biométrica. La biométrica se encarga de desarrollar dispositivos de identificación mediante rasgos o características humanas. Existen sistemas de identificación por huella digital, morfología de mano, mediante el iris del ojo, entre otros.
3. **Programa de gestión de control de presencia:** Es el programa informático encargado de gestionar la función de gestión en la empresa.
4. **Sistema de Comunicación:** Es el sistema que permite la comunicación entre el soporte de identificación y el reloj de fichaje.

Las tecnologías en el mundo avanzan a un nivel muy acelerado y de conjunto con estas los SCP, desarrollándose sistemas que apenas unos años atrás eran considerados ficción en la cinematografía mundial. En la actualidad existen empresas encargadas de realizar estos sistemas, destacándose IDS S.A y AM System. Ambas empresas se dedican a la comercialización de software y terminales para el control de presencia, presentando una amplia gama de productos que oscilan desde los más sencillos como los dispositivos que utilizan soporte de identificación a través de tarjetas de cartulina, hasta los más avanzados que son mediante identificación biométrica y detección de proximidad.

El sistema AM Presencia, es una aplicación destinada para el control de acceso y presencia en empresas a pequeña y gran escala, mediante el fichaje que son realizados en una terminal TCP-5[1] o LPC-4[2]. Este sistema es comercializado por la empresa AM System.

AM Presencia presenta las siguientes características:

- Asignación de horarios individualmente a cada empleado.
- Márgenes de entrada y salida en horarios.
- Remuneración de horas extraordinarias por empleado o según el cargo que ocupe en la empresa.
- Días de asistencia al trabajo.
- Ausencias al trabajo.
- Cálculo de horas que realmente ha trabajado cada empleado.
- Cálculo de horas teóricas.
- Cálculo de horas extras y su correspondiente remuneración.
- Llegadas y salidas fuera de horario.
- Control de fichajes impares.
- Seguimiento en tiempo real de las entradas y salidas que están realizando los empleados, se puede conocer en todo momento qué empleado está dentro o fuera de la empresa.
- Posibilidad de especificar una hora de salida distinta (salida real) a la que se especifique en el contrato de cada empleado: dando la posibilidad de contabilizar la diferencia entre la salida según contrato y la salida real como horas extras.

---

[1] Terminal de control de presencia comercializada por la misma empresa. Presenta comunicación a través de Ethernet utilizando TCP/IP. Dispone además de conexión serie **RS232C** o **USB** y multinodo **RS485**. Opcionalmente incluye conectividad Bluetooth, lo que permite comunicación inalámbrica con distancias de hasta 10m.

[2] Terminal de control de presencia con características similares a la terminal **TCP-5**, incluyendo que su funcionamiento puede trabajar conectada permanentemente a un PC (**ON-LINE**) o de manera autónoma (**OFF-LINE**), requiriendo en este caso la conexión sólo para la descarga de datos o parametrización.

- Flexible y completa edición de fichajes: permite añadir, modificar y eliminar el fichaje de un empleado en determinado día de forma rápida, cómoda y sencilla utilizando solamente el ratón.
- Pantalla de revisión de fichajes.
- Posibilidad de añadir, modificar y eliminar los fichajes recogidos del terminal antes de grabarlos para su cálculo.

Nuestro país se encuentra sumergido en un proceso de informatización y debido a sus escasos recursos la automatización del control de presencia sólo se toma como práctica en pocos centros laborales. En la mayoría de los centros el control se realiza en formato duro a través de planillas, haciendo más engorrosa la contabilización de las horas trabajadas del personal de la empresa. Aquellos que han logrado informatizar parte del control del horario laboral, básicamente lo hacen a través de un sistema de control de acceso.

En la UCI el control de presencia está semi-informatizado, ya que en muchas áreas este proceso se registra en modelos impresos y el contenido de los mismos es almacenado manualmente a una hoja electrónica de cálculo por el jefe de área o un trabajador encargado de esta función, conformándose así la pre-nómina de los trabajadores de cada área. Existen sistemas en algunas áreas que registran la entrada y salida de los trabajadores, así como unas sencillas estadísticas. Debido a que ninguno de los métodos de control de presencia, existentes en la universidad, están conectados directamente con el sistema ASSET de recursos humanos, existe la necesidad de emplear tiempo y recursos para almacenar manualmente la información que contiene la pre-nómina generada en cada área.

1.3 Dispositivos de los SCP

1.3.1 Relojes

Inicialmente los relojes utilizados para el control de presencia eran mecánicos, usando para ello una ficha de cartulina en la cual marcaba la hora de entrada y salida de los trabajadores. Precisamente el hecho de usar este tipo de relojes es lo que ha provocado que los relojes de control de presencia se conozcan en el mundo como reloj de fichar. Con la aparición del microprocesador en los años ´70 se comenzó a confeccionar relojes basados en microprocesadores, significando así una nueva era de los relojes de fichaje. Otro gran paso de avance en estos mecanismos fue la aparición del Sistema Operativo DOS, esto permitió que el reloj ya tuviera una interfaz para que el operario del mismo pudiera interactuar con él.

Hasta la fecha han aparecido variantes y mejoras para los SCP, todas alrededor del hardware y el software incorporándose la interfaz TCP/IP. Con la aparición de la web se ha logrado generar reportes que el empleado puede consultar constantemente, esto no solo les ha ofrecido beneficio a Recursos Humanos para la gestión del control del horario y asistencia de los trabajadores, sino también a los propios empleados. [3]

En el mundo cada vez es más habitual que todo tipo de empresas usen un reloj de fichar; los fichajes mediante identificación por radiofrecuencia (RFID) o huella digital–biométricos, son los más utilizados ya que facilitan y agilizan el proceso de fichar. Las tarjetas de proximidad RFID son tarjetas con forma similar a las tarjetas de crédito pero con mayor grosor, son leídas por radio, sin que se produzca contacto físico con el reloj de fichar. Por su parte, la identificación mediante huella digital–biométricos se realiza fundamentalmente mediante cuatro características singulares, la biometría de huella dactilar que funciona por reconocimiento de la huella digital de alguno de los dedos de las manos, la biometría de perfil de mano la cual funciona por reconocimiento de la morfología de la mano, la biometría de iris que funciona por reconocimiento del iris del ojo y finalmente la biometría de voz la cual funciona por reconocimiento de la voz del usuario.

Entre los relojes más usados se encuentra el terminal Futura ID3.300, diseñado para tarjetas magnéticas y tarjetas de proximidad. Cuenta con un diseño moderno, compacto y de reducido tamaño, lo que le permite ajustarse fácilmente dentro de cualquier ambiente de trabajo: oficinas, talleres, fábricas, etc. Su funcionamiento está basado en la gestión de captura de marcajes de forma totalmente autónoma, almacenándolos en su memoria y descargándolos de forma periódica en el ordenador para su posterior gestión desde el programa. [4]

Otro de los relojes utilizados para el control de acceso y presencia es el Midman M-660, que utiliza el método tradicional con tarjetas de cartón. Permite hasta 6 impresiones diarias, entrada y salida mañana, entrada y salida tarde, entrada y salida extra o permisos. Cuenta con reserva de memoria de tres años de los datos programados en caso de tenerlo sin alimentación y tiene incluido el cambio de horario invierno verano de manera automática. [5]

Por su parte el terminal Kimaldi BioMax2 cuenta con una interfaz RS-232, Ethernet UDP, TCP/IP y de manera opcional Wi-Fi. Este tipo de reloj permite la autenticación por biometría combinada de manera opcional con proximidad, smart card (tarjeta inteligente), banda magnética o código de barra. Es usado

para el control de acceso a universidades, hoteles, centros de procesamiento de datos y además para el control de la producción en aplicaciones industriales.

Existen cuatro modelos de este tipo de terminal de huella dactilar, Kimaldi KBio 2 off-line el control de acceso se realiza con identificación biométrica off-line de huella dactilar y gestión de usuarios de la base de datos on-line, Kimaldi KBio 2 on-line donde el funcionamiento del sistema se controla desde el host, Kimaldi KBio 2 autónomo el cual no necesita de conexión a PC y por último el Kimaldi KBio 2 semi-autónomo, su funcionamiento está basado de manera autónoma con envío de eventos on-line. [6]

De manera general el objetivo de estos terminales y de otros que existen en el mundo es garantizar el registro del cumplimiento del horario de los trabajadores.

1.3.2 Código de barra

Anteriormente se hizo referencia a la utilización del código de barra como soporte de identificación en relojes de fichaje. El primero de estos surgió en 1952 por Norman Woodland y Bernard Silver, junto con la colaboración de los ingenieros Raymond Alexander y Frank Stietz, el mismo estaba hecho por una serie de círculos concéntricos y su objetivo era identificar los vagones de los ferrocarriles.

El año 1973 fue un hito para la historia del código de barra, fue creado en Estados Unidos el Universal Product Code (UPC) de 12 dígitos. Al ver el avance alcanzado por este soporte de identificación se crea entonces en Bélgica en 1977 la International Article Numbering Association (EAN Internacional, hoy día GS1), con el propósito de facilitar un lenguaje común para el comercio internacional.

Es necesario aclarar que el código de barra es una referencia que permite la identificación del bien, y es preciso acceder a una base de datos para obtener las características de lo identificado.

Como se puede apreciar, existen varios tipos de códigos de barra que aunque cumplan los mismos objetivos todos tienen una simbología diferente y además se clasifican en dos grandes grupos:

- Los lineales (1D), usados en los productos y permiten incluir mensajes pequeños. El almacenamiento de los datos se realiza usando una combinación lineal de barras y espacios sin importar el tamaño de las medidas.

  La estructura básica de un código de barras de una dimensión (1D) consiste de una zona de inicio y una zona de término en la que se incluye: un patrón de inicio, uno o más caracteres de datos, opcionalmente uno o dos caracteres de verificación y patrón de término. La información es leída

con lectores de código de barras los cuales envían la información a una computadora como si la información hubiese sido tecleada. [7]

- Los de dos dimensiones (2D) usado en documentos que necesitan tener incluidos mensajes grandes, como por ejemplo una historia clínica. En este caso los datos se almacenan a lo ancho o a lo alto, incrementando la seguridad de los mismos.

  Un código de barras de dos dimensiones (2D) es una representación en dos dimensiones de un conjunto de valores numéricos y alfanuméricos (letras, números y caracteres especiales) representados por un patrón predeterminado. [7]

Existen en el mundo varias simbologías que pueden ser usadas por los códigos de barras, ellas son Intercalado 2 de 5 (ITF), Código 39 (C39), Código 128 (C128), ISBN 13, PDF417 y los más usados a nivel mundial en el ámbito comercial son el UPC y el EAN.

El código de barra utilizado en la Universidad de las Ciencias Informáticas para la confección de las credenciales es el código 39. Uno de los motivos por los cuales es usado en la UCI es el conjunto de caracteres que lo componen, que incluye letras y números. Además que el mismo puede tener una longitud variable y puede cambiar en ancho y altura. Siendo también uno de los códigos de barras más antiguos y utilizados.

1.3.3 Lectores de código de barra

En 1973 Jerome Swartz y Sheldon Harrison fundan la empresa Symbol Technologies, dedicada a vender los primeros códigos de barra de acuerdo con los números del código UPC. Ambos aplicaron sus conocimientos sobre códigos de barra y desarrollaron el LS100, primer escáner láser de mano. Luego en 1982 se realiza el lanzamiento del LS7000 comenzando abrir paso al desarrollo de los venideros códigos de barra que conocemos en la actualidad. En esa misma década se crea el LS8000, siendo este el primer escáner láser robusto de mano. Proporcionando mejoras en la productividad de las empresas, en 1990 es lanzada la primera computadora móvil de mano, con la combinación del código de barra LRT3800 y la informática móvil e inalámbrica, lo que hoy en día es usado en diversidad de aplicaciones. [8]

Existen cuatro principales tipos de lectores, estos son: el llamado lápiz óptico o wand que funciona deslizándose por el ancho del código transmitiendo una señal digital de las barras y espacios a la misma velocidad con la que es deslizado, el láser de pistola que realiza una lectura mediante una luz láser y

genera una señal llamada Hand Held Laser Compatible (HHLC), el CCD (Charge Cuopled Device) el cual toma una “foto” del símbolo del código de barra captado y lo traduce a una señal, la cual puede ser como la del lápiz óptico o de tipo HHLC y el lector láser omnidireccional que lo que envía es un patrón de rayos láser y permite leer parte de la simbología del código de barra sin importar su ubicación. [9]

1.4 Metodologías y Lenguajes de programación

Se presentará en el presente epígrafe una descripción de las metodologías y lenguajes a utilizar para el desarrollo de la solución informática.

1.4.1 Metodologías de desarrollo de Software

Las metodologías de desarrollo de software surgen en los años ‘60s del pasado siglo producto a lo complejo que resulta el desarrollo de un software. A partir de esta fecha comienza a surgir un gran número de metodologías como Programación estructurada Jackson, Structured Systems Analysis and Design Methodology (SSADM), Scrum, entre otras. Esta última en la actualidad es una de las metodologías ágiles más utilizadas a nivel mundial. Las metodologías son clasificadas en dos grupos, ágiles y tradicionales, en dependencia del tiempo de vida y la complejidad del proyecto que se vaya a realizar.

1.4.1.1 Metodologías ágiles

En febrero del 2001 diecisiete expertos de la industria del software, donde se encontraban un gran número de los creadores e impulsores de las metodología de software se reúnen con el objetivo de esbozar los valores y principios que les permitiera a los equipos desarrollar software rápidamente y con flexibilidad a los cambios que pudieran surgir a lo largo del ciclo de vida del proyecto, se pretendía proponer alternativa a los procesos de desarrollo de software tradicionales, ya que estos solían ser rígidos y dirigidos por la documentación que se generaba a partir de las actividades del proceso. En esta reunión surge por primera vez el término “ágil” y se crea La Alianza Ágil (The Agile Aliance), una organización con el objetivo de promover los conceptos relacionados con el desarrollo ágil de software. El punto de partida de esta organización fue el Manifiesto Ágil, este documento refleja la filosofía ágil.

Históricamente las metodologías tradicionales han intentado abordar una mayor cantidad de situación en el contexto de proyectos, haciéndose considerable el esfuerzo para ser adaptadas a proyectos pequeños

y con requisitos cambiantes. Las metodologías ágiles ofrecen una solución a la medida para los proyectos con estas características. Una de las cualidades más destacadas de este tipo de metodología es su sencillez en el aprendizaje y su aplicación, reduciendo ampliamente los costos de implementación en un equipo de desarrollo, esto ha llevado a las metodologías ágiles a un punto de preferencia en la comunidad de desarrolladores.

Las metodologías ágiles son una agrupación de las prácticas tradicionales pero llevadas al extremo, tomando la esencia y aplicándola en el desarrollo desde el inicio, entregas oportunas y la entrega final del proyecto, teniendo en cuenta mantenimiento, soporte, auditoría y capacitación al usuario final. Dentro del amplio número de metodologías ágiles surgidas en las dos últimas décadas se destacan:

- **Dynamic Systems Development Method (DSDM):** Surge en 1994 con el objetivo de crear una metodología de desarrollo rápido de aplicaciones (RAD siglas en inglés). Sus principales características son: proceso iterativo e incremental y el equipo de desarrollo trabaja en conjunto con el usuario. Propone cinco fases: estudio viabilidad, estudio del negocio, modelado funcional, diseño y construcción, y finalmente implementación.
- **Feature - Driven Development (FDD):** Define un proceso iterativo que consta de 5 pasos. Las iteraciones son cortas (hasta 2 semanas). Se centra en las fases de diseño e implementación del sistema partiendo de una lista de características que debe reunir el software.
- **Scrum:** Fue creada por Ken Schwaber, Jeff Sutherland y Mike Beedle. Esta metodología es básicamente para gestión de proyecto, se emplea en proyecto con requisitos cambiables y presenta dos características fundamentales. La primera característica es la realización de reuniones a lo largo de la duración del proyecto y fundamentalmente la reunión diaria durante 15 minutos. La segunda característica es que se realiza el desarrollo del software mediante iteraciones denominas *sprints*. Estos *sprints* son incrementos ejecutables que se les muestran al cliente en la finalización de cada uno de estos.
- **Extreme Programming (XP) o Programación Extrema:** Es una metodología centrada en la compenetración interpersonal para el buen desarrollo del software, fomenta el trabajo en equipo, promoviendo el buen clima en los puestos de trabajo y todo lo relacionado con el bienestar del equipo de trabajo. Se basa en la retroalimentación entre el cliente y los desarrolladores y enfrenta

con coraje los cambios que desee el cliente. Sus características se engloban en tres apartados fundamentales: procesos y prácticas, historial de usuario y roles.

- **SXP:** Esta metodología será ampliada a continuación.

Metodología SXP

Debido a la dinámica de los proyectos de la UCI, en el 2007 se pone en práctica en seis proyectos del grupo UNICORNIO de la Facultad 10 una nueva metodología, SXP. La misma es creada en la UCI a partir de la fusión de dos metodologías ágiles fundamentales Scrum y XP.

En la actualidad está metodología se emplean en los proyectos a cargos de la Dirección de Informatización, básicamente se emplea Scrum para la Gestión de software y XP para el desarrollo del mismo.

SXP se orienta para proyectos con pequeños equipos de trabajo, con requisitos cambiables o imprecisos y de corto tiempo de duración, permitiendo en estos casos una rápida entrega del producto. Permite que el equipo esté unido y trabajen en una sola dirección, con un objetivo claro, facilitando el seguimiento y control de las tareas a realizar, de forma que los jefes puedan verificar día a día el avance el proyecto. No se rige estrictamente por la documentación y permite que el cliente se integre al equipo de trabajo. Posibilita que el tiempo de desarrollo sea de corto plazo. [10]

Esta metodología consta de 4 fases fundamentales:

1. **Planificación – Definición:** En esta fase se define la visión, expectativas y el aseguramiento del financiamiento del proyecto.
2. **Desarrollo:** Se realiza la implementación del producto hasta que esté listo para ser entregado.
3. **Entrega:** Se pone en marcha el producto y se hace la entrega al cliente del mismo.
4. **Mantenimiento:** Se le presta servicio de soporte al cliente.

Dentro de estas fases se realizan numerosas actividades, las cuales pueden ser, levantamiento de requisitos, definición de las Historias de Usuarios (HU), diseño e implementación, pruebas entre otras. De estas actividades se generan artefactos para documentar todo el proceso, estos son:

1. Concepción del sistema

2. Modelo de Historias de Usuarios del Negocio
3. Lista de Reserva del Producto
4. Historias de Usuarios
5. Lista de Riesgos
6. Modelo de Diseño
7. Tarea de Ingeniería
8. Plan de Releases
9. Estándar de Programación
10. Casos de Pruebas de Aceptación
11. Manual de Usuario
12. Manual de Identidad
13. Manual de Desarrollo
14. Gestión de Cambios [11]

Las actividades que son registradas en estos artefactos son llevadas a cabo por un conjunto de personas que cumplen un rol dentro del proyecto, estos roles son:

1. **Líder del Proyecto (Scrum Master):** Es un rol de administración que debe asegurar que el proyecto se está llevando a cabo de acuerdo con las prácticas y que todo funciona según lo planeado.
2. **Gerente (Management):** Es el responsable de tomar las decisiones finales, acerca de estándares y convenciones a seguir durante el proyecto.
3. **Especialistas:** Son los responsable del proceso global. Es necesario que conozca a fondo el proceso, ya sea de la metodología utilizada o cualquier otro proceso o elementos de gran importancia para el desarrollo de software.
4. **Consultor:** Es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto, en el que puedan sugerir problemas, además aportan ideas y experiencias para el beneficio del sistema en desarrollo.

5. **Cliente (Customer):** El cliente participa en las tareas que involucran la lista de reserva del producto.
6. **Miembros del Proyecto (Scrum Team):** Típicamente es un equipo de entre 5 y 10 personas cada una especializada en algún elemento que conforma los objetivos a cumplir, por ejemplo: Programadores, Diseñadores de Interfaz de usuario, etc.
    - **Programadores (Programmers):** Es el encargado de producir el código y escribir las pruebas unitarias. Debe existir una comunicación y coordinación adecuada entre los programadores y otros miembros del equipo.
    - **Analista (Analyst):** Es el encargado de escribir las historias de usuario y las pruebas funcionales para validar su implementación.
    - **Diseñadores (Designers):** Encargados del diseño del sistema; así como el de los prototipos de interfaces, máximos responsables de la realización del diseño de las metáforas y supervisan el proceso de construcción.
    - **Encargado de Pruebas (Tester):** Es el encargado de ayudar al cliente a escribir las pruebas funcionales. Ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas.
    - **Arquitecto (Architect):** Se vincula directamente con el analista y el diseñador debido a que su trabajo tiene que ver con la estructura y el diseño en grande del sistema. [10]

1.4.2 Business Process Management (BPM)

Es una metodología empresarial destinada a mejorar la eficiencia de los procesos de negocio que se deseen modelar, integrar, optimizar o informatizar a través de la gestión sistemática de estos. En el 2004 se crea una notación estándar para el modelado de los procesos de negocios, Business Process Modeling Notation (BPMN), dicha notación agrupó sus elementos en cuatro grupos principales, *objetos de flujo, objetos conectores, artefactos y swimlanes.*

- **Objetos de flujo:** Muestran la dinámica del proceso de negocio, esto es mediante *eventos, actividades y decisiones.*

- **Objetos conectores:** Los objetos de flujo se conectan entre ellos para crear el esqueleto básico de la estructura de un proceso de negocio, en este grupo se encuentran los *flujos de secuencia, flujos de mensajes y las asociaciones.*
- **Artefactos:** Los artefactos dentro del proceso de negocio pueden ser los *objetos de datos, los grupos y las anotaciones o comentarios.*
- **Swimlanes (canales):** Es un mecanismo para organizar actividades y está compuesto por *Pool y Lane.* [12]

Cada uno de estos elementos permite la representación gráfica del proceso de negocio formándose así el Diagrama de Proceso del Negocio (BPD, siglas en inglés).

1.4.3 Lenguajes de programación

Los lenguajes de programación son idiomas artificiales que tienen como objetivo crear programas que le den órdenes a una computadora, ya que ellas por sí mismas no ejecutan acciones. La estructura de los lenguajes consta de un léxico, una sintaxis y una semántica:

- **Léxico:** Conjunto de símbolos o vocabulario permitidos por el lenguaje.
- **Sintaxis:** Reglas que indican como realizar la construcción del lenguaje.
- **Semántica:** Reglas que permiten determinar el significado de la construcción del lenguaje.

Los lenguajes se clasifican en dos grupos atendiendo al número de instrucciones a colocar para realizar una determinada instrucción.

- **Bajo nivel:** Es el tipo de lenguaje que cualquier computadora pueda entender.
- **Alto nivel:** Son lenguajes muy similares al lenguaje humano, usando palabras y frases fáciles de entender para dictar determinadas instrucciones a una computadora.

1.4.3.1 Lenguajes del lado del servidor

Hypertext Preprocessor (PHP)

Fue creado en 1994 por Rasmus Lerdor, aunque en la actualidad es desarrollado por un grupo de personas pertenecientes a The PHP Group. Es un lenguaje de script (o de guiones), fue diseñado con el

objetivo de aumentar el dinamismo de la web, aunque inicialmente solo se empleaba en el mantenimiento de páginas web mediante un conjunto de macros.

Durante el transcurso de los años las características de PHP fueron creciendo hasta convertirse hoy en un lenguaje de programación completo, siendo capaz de manejar un entorno que integre grandes Bases de Datos como MySQL, Microsoft SQL Server, Firebird, PostgreSQL, SQLite y Oracle, permitiendo la creación de aplicaciones web muy robustas. Tiene la capacidad de ser ejecutado en la mayoría de los sistemas operativos, tales como los soportados por UNIX (Linux, Mac OS X) y Windows, además puede interactuar con los servidores web más populares.

Facilita la realización de grandes cosas en pocas líneas de código y trabaja en combinación con otras tecnologías como Perl, JavaScript, Phyton, entre otras. La combinación de PHP y Qt/GTK permite la creación de aplicaciones gráficas independientes al navegador, así como aplicaciones de escritorio en los sistemas operativos en los que están soportados.

Es un lenguaje del lado del servidor, por lo que se le hace invisible al navegador y al cliente el código fuente, ya que se encarga de ejecutar el código y enviarle al navegador un código HTML, lo que hace que sea más segura la aplicación web. Es muy legible y fácil de aprender, no soporta directamente punteros, de forma que no existen los problemas de depuración provocados por estos y es libre por lo que es de fácil acceso para todos.

1.4.3.2 Lenguajes del lado del cliente

HyperText Marckup Language (HTML)

Fue creado en 1986 por el físico nuclear Tim Berners-Lee y es producto de la unión del hipertexto (conocido como link o ancla) y el Lenguaje Estándar de Marcación General (SGML)[3]. No se puede considerar precisamente un lenguaje de programación como C++, Visual Basic, etc.; sino más bien es un sistema de etiquetas que carece de compiladores por lo que no se detecta errores, solo se muestra lo que el navegador interprete del código HTML. El entorno de trabajo es solo un procesador de texto, no importa cual, solo se debe guardar con extensión HTML, producto a esto puede ser desarrollado a partir de

[3] **SGML** son las siglas de ***S**tandard **G**eneralized **M**arkup **L**anguage,* consiste en un sistema para la organización y etiquetado de documentos, este lenguaje fue normalizado por la ISO en 1986.

cualquier sistema operativo y ser interpretado por una gran variedad de navegadores o browsers como, Netscape Navigator, Mosaic, Opera, Internet Explorer de Windows, Mozilla Firefox, Chrome de Google, entre otros. En la actualidad la mayoría de las páginas en la World Wide Web (www) están soportadas por códigos HTML porque es estandarizado y multiplataforma.

Cascading Style Sheets (CSS)

Las hojas de estilos o CSS es un leguaje utilizado para definir la presentación de un documento que esté escrito en HTML o XML, el objetivo fundamental del estilo en cascada es separar la estructura del documento a su apariencia, facilitando al desarrollador hacer cambios en la apariencia del documento sin necesidad de buscar mucho ya que no tiene por medio la estructura y contenido del documento.

Mantiene un control centralizado de la apariencia de la web, agilizando la actualización del mismo y posibilitándole al usuario seleccionar un estilo propio que le permita ver mejor el sitio. La página puede tener varios estilos por independiente, teniendo un estilo para la impresión y otro para la visualización del sitio web. Reduce considerablemente la codificación del documento HTML.

JavaScript

JavaScript es un lenguaje interpretado, es decir, prescinde de un compilador. Se utiliza básicamente en páginas web, permitiendo el desarrollo de interfaz de usuarios mejoradas y la realización de páginas web dinámicas.

En las páginas web se venía utilizando HTML para realizar operaciones por el lado del cliente sin necesidad de acceso a funciones del servidor, con la llegada de JavaScript se logró ampliar las operaciones a realizar en el cliente, descargándose en esta las sentencias JavaScript en conjunto con la codificación HTML.

Actualmente los navegadores modernos interpretan el código JavaScript integrado dentro de las páginas web. Para la interacción con una página web se provee al lenguaje JavaScript de una implementación del Document Object Model (DOM).

1.5 Herramientas y tecnologías

Se presentará en el presente epígrafe una descripción de las herramientas y tecnologías existentes en el mundo, posibles a utilizar para el desarrollo de la solución informática.

1.5.1 Herramientas

Visual Paradigm – UML 6.0 (VP – UML 6.0)

Es una herramienta CASE creada para el ciclo vital del desarrollo de software, permitiendo la captura de requisitos, análisis y diseño e implementación. Considerada como una herramienta muy completa y fácil de usar, con soporte multiplataforma.

Es compatible con todas las fases y roles en un proceso de desarrollo de software tales como el ingeniero de software, analistas del sistema, arquitecto del sistema, entre otros. VP– UML 6.0 es empleado para las aplicaciones grandes y complejas que presenten un enfoque orientado a objeto.

VP–UML 6.0 proporciona características tales como generación del código, ingeniería inversa y generación de informes. Tiene la capacidad de crear el esquema de clases a partir de una base de datos y crear la definición de base de datos a partir del esquema de clases. Incorpora el soporte para trabajo en equipo, que permite que varios desarrolladores trabajen a la vez en el mismo diagrama y vean en tiempo real los cambios hechos por sus compañeros.

Ventajas de VP – UML 6.0

- Multiplataforma y muy útil en la generación de códigos fuente en PHP.
- Integración completa con Microsoft Office.
- Incorpora soporte para el trabajo en equipo, permitiendo que varios desarrolladores trabajen en el mismo diagrama y vean en tiempo real los cambios realizados por el resto del equipo de trabajo.
- Permite invertir código fuente de programas y archivos ejecutables en modelos UML al momento, creando de forma simple la documentación.

NetBeans 6.7.1

Fue creado en 1996 en una universidad de Praga, República Checa, inicialmente se llamó Xelfi y era un proyecto estudiantil, la meta era escribir de desarrollo integrado (IDE) para Java, parecido a Delphi. Más tarde Roman Stanek se interesa en financiar el proyecto y poco después se lanza al mercado la primera versión de Netbeans.

La plataforma NetBeans permite que las aplicaciones sean programadas a partir de un conjunto de componentes de software llamados módulos. Un módulo es un archivo Java que contiene clases de Java escritas para interactuar con las APIs de NetBeans.

La plataforma brinda servicios comunes a las aplicaciones de escritorio, permitiéndole al desarrollador enfocarse en la lógica específica de su aplicación. Entre las características de la plataforma se encuentra:

- Administración de las interfaces de usuarios.
- Administración de las configuraciones de usuarios.
- Administración del almacenamiento.
- Administración de ventanas.
- Frameworks basados en asistentes.

NetBeans IDE es una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas, aunque esté escrito en Java se puede programar en otros lenguajes. Es usada para el desarrollo de aplicaciones de escritorio grandes y su licencia es libre y gratuita sin restricciones para su uso.

PgAdmin III

Es una interfaz gráfica para el gestor de bases de datos PostgreSQL, siendo la más completa con licencia OpenSource. Está escrito en C++ usando la librería gráfica multiplataforma wxWidgets, lo que permite que se pueda utilizar en cualquiera de los sistemas operativos más utilizados. Con PgAdmin III los usuarios pueden escribir tanto consultas SQL simples, como desarrollar bases de datos complejas. La interfaz gráfica soporta todas las características de PostgreSQL facilitando su administración, también incluye un editor SQL con resultados de sintaxis, un editor de código por la parte del servidor y más.

La conexión al servidor se puede hacer mediante el protocolo TCP/IP o Unix Domain Sockets y puede encriptarse en SSL para mayor seguridad.

GIMP

GIMP (GNU Image Manipulation Program) es un programa para la manipulación de imágenes, tanto de fotos como dibujos, es una herramienta libre comprendida bajo la licencia de GNU, es multiplataforma. Existen versiones al español y al lenguaje original de programación, el inglés. Es muy similar al Photoshop.

Axure RP

Axure RP es una aplicación ideal para crear prototipos y especificaciones muy precisas para páginas web. Se trata de una herramienta especializada en la tarea, así que cuenta con todo lo que se puede necesitar para crear los prototipos de forma más eficiente. Axure RP te permite componer la página web visualmente, añadiendo, quitando y modificando los elementos con suma facilidad.

1.5.2 Tecnologías

1.5.2.1 Servidores Web

Un servidor web es un programa que se encarga de atender y responder peticiones realizadas por un cliente mediante un navegador, este intercambio se realiza a través de dos protocolos fundamentales, el protocolo HTTP (Hyper Text Transfer Protocol) y el protocolo HTTPS que es una versión cifrada del protocolo HTTP.
Los servidores web simples presentan un funcionamiento básico que se explica a continuación, está ejecución se realiza de forma infinita hasta que se cierra la conexión entre servidor y cliente.

1. Espera peticiones en el puerto TCP indicado (el estándar por defecto para HTTP es el 80).
2. Recibe una petición.
3. Busca el recurso.
4. Envía el recurso utilizando la misma conexión por la que recibió petición.
5. Vuelve al segundo punto. [13]

Servidor Apache 2.2

El servidor Apache es un servidor de tecnología Open Sourse (código abierto) potente, para plataformas Unix, Windows y Mac que implemente el protocolo HTTP, además de ser multiplataforma, se tiene fácil acceso a la documentación del mismo. Su nombre se debe a Behelendorf, eligió este nombre en honor a las tribus Apache de las tierras norteamericanas, con el objetivo de darle una visión firme y enérgica, ya que esta tribu fue la última en rendirse a lo que pronto fuese el gobierno de los EE.UU.

Apache presenta bases de datos de autenticación y negociación de contenido, es fuertemente criticado por falta de una interfaz gráfica para su configuración, pero esto no frena que sea el servidor web más utilizado en el mundo desde 1996.

Apache es un software que está estructurado en módulos. La configuración de cada módulo se hace mediante la configuración de las directivas que están contenidas dentro del módulo. Los módulos del Apache se pueden clasificar en tres categorías: Módulos Base, Módulos Multiproceso y Módulos Adicionales, esta estructura modular es lo que lo hace extensible.

Las funcionalidades más elementales se encuentran en el módulo base, siendo necesario un módulo multiproceso para manejar las peticiones. Se han diseñado varios módulos multiproceso para cada uno de los sistemas operativos sobre los que se ejecuta el Apache, optimizando el rendimiento y rapidez del código. El resto de las funcionalidades del servidor se consiguen por medio de módulos adicionales que se pueden cargar. Para añadir un conjunto de utilidades a este, simplemente hay que añadirle un módulo, de forma que no es necesario volver a instalar el software. [14]

1.5.2.2 Sistemas Gestores de Bases de Datos (SGBD)

Los Sistemas de Gestión de Bases de Datos tienen un peso fundamental dentro de los sistemas de información ya que estos se encargan de manipular del manejo de la misma, dígase, definición, construcción y manipulación de los datos. Permitiendo la eliminación y actualización de registros, combinación con otras bases de datos, generación de informes impresos, entre otras funcionalidades.

En el mundo existen un gran número de Gestores de Bases de Datos, dentro de los que se destacan Oracle, SQL Server por parte de los privativos y MySQL y PostgreSQL por parte del software libre.

PostgreSQL

PostgreSQL es un sistema de Gestión de Bases de Datos objeto-relacional (ORDBMS), creado en el Departamento de Investigación de la Universidad de California en Barkeley y aunque su licencia es propiedad de dicha universidad, es libre para utilizar, copiar, modificar y distribuir, sin importar para los fines que se aplique, tanto comercial como académico.

Soporta casi todas las sintaxis SQL y puede almacenar grandes volúmenes de información tales como imágenes, audios, videos, etc., manteniendo la integridad de los mismos. Cuenta con herramientas gráficas de administración y diseño de Bases de Datos tales como PgAdmin y phpPgAdmin, estas herramientas permiten que se haga sencilla la administración de las Bases de Datos.

Reduce costos considerables de operación, manifiesta estabilidad y confiabilidad legendaria, es extensible y multiplataforma.

1.5.2.3 Ajax

Ajax es el acrónimo de **Asynchronous JavaScript + XML**, en si no es una tecnología, sino que es la unión de varias tecnologías que juntas pueden lograr aplicaciones muy potentes.

Una aplicación AJAX elimina la naturaleza “arrancar-frenar- arrancar-frenar” de la interacción en la Web introduciendo un intermediario -un motor AJAX- entre el usuario y el servidor. [15]

El motor AJAX permite que la interacción del usuario con la aplicación se realice independiente de la comunicación con el servidor. De esta manera el usuario nunca estará mirando una ventana en blanco del navegador y esperando respuesta del servidor.

Ajax contiene presentación basada en estándares usando HTML y CSS, para la interacción dinámica utiliza el DOM. Mantiene el intercambio y la manipulación de los datos usando XML, utiliza HMLHttpRequest para la recuperación de datos asincrónica. La unión de estas la realiza en JavaScript.

1.5.2.4 Framework

Un framework es un producto que sirve como base para la programación avanzada de aplicaciones, este nos aporta una serie de funciones para realizar tareas habituales. Es una librería de código que contiene procesos ya listos para usar. Los programadores utilizan los frameworks para no tener que desarrollar

ellos las tareas más básicas, ya que estos contienen implementaciones que están probadas, que funcionan y que no se necesitan volver a programar.

CodeIgniter

CodeIgniter es un conjunto de herramientas para personas que construyen su aplicación web usando PHP. Su objetivo es permitirle desarrollar proyectos mucho más rápido, provee un rico juego de librerías para tareas comúnmente necesarias, así como una interface simple y estructura lógica para acceder a esas librerías.

CodeIgniter le permite creativamente enfocarse en su proyecto minimizando la cantidad de código necesaria para una tarea dada. Tiene una amplia compatibilidad con cuentas de hosting estándar que corren una variedad de versiones y configuraciones de PHP.

Además presenta un entorno de trabajo que no requiera usar línea de comando, es libre y multiplataforma.

Jquery

jQuery fue creado por John Resig, es un framework de Java Script que se utiliza para la implementación del lado del cliente, es decir, para la creación de las interfaces de usuario. Es una librería de JavaScript considerada para interactuar con los elementos de una web por medio del DOM, su sintaxis es sencilla de utilizar y el código a escribir es de reducido tamaño. Tiene gran variedad de plugins, los cuales nos permiten un desarrollo más avanzado de las validaciones, los efectos, las animaciones, etc.

Provee mecanismos para la captura de eventos y es compatible con varios navegadores web, como son: Firefox, Internet Explorer, Safari, Opera y Chrome.

Es un software que tiene licencia libre para usar, ya sea con fines personales o comerciales. Tiene muy buena aceptación por parte de los programadores y un alto grado de acierto en el mercado. Además es un producto serio, estable y con mucha documentación, a la cual se tiene fácil acceso. Contiene funcionalidades para trabajar con AJAX para el desarrollo web rápido.

1.6 Patrones de Arquitectura

1.6.1 Modelo Cliente Servidor

Esta arquitectura consiste básicamente en que un usuario realiza una petición a un programa (servidor) y este le emite una respuesta. Aunque esta idea la cumplen también sistemas que se encuentran en una misma computadora, es mucho más ventajoso en sistemas operativos multiusuario distribuidos en una red de computadoras. Se establece así un enlace de comunicación transparente entre los elementos que conforman la estructura.

En esta arquitectura el servidor no necesita potencia de procesamiento, parte del proceso se reparte con los clientes, un único servidor típicamente sirve a una multitud de clientes, ahorrando a cada uno de ellos el problema de tener la información instalada y almacenada localmente. Además reduce el tráfico de red de manera considerable. Idealmente, el cliente se conecta al servidor cuando es estrictamente necesario, obtiene los datos que necesita y cierra la conexión dejando la red libre.

Entre las principales características de la arquitectura Cliente/Servidor, se pueden destacar las siguientes:

1. El servidor presenta a todos sus clientes una interfaz única y bien definida.
2. El cliente no necesita conocer la lógica del servidor, sólo su interfaz externa.
3. El cliente no depende de la ubicación física del servidor, ni del tipo de equipo físico en el que se encuentra, ni de su sistema operativo.
4. Los cambios en el servidor implican pocos o ningún cambio en el cliente.

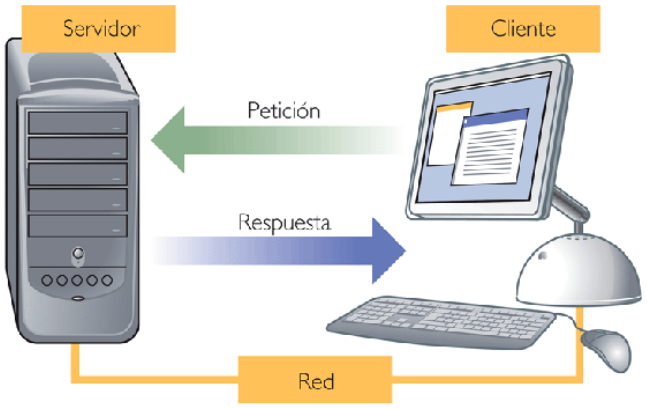


Figura 1.1: Modelo Cliente–Servidor

1.6.2 Modelo Vista Controlador (MVC)

Este modelo consiste en separar los componentes relacionados con los datos del sistema, los elementos relacionados con la interfaz y la lógica del control de estos elementos, cada uno de estos elementos se recogen en tres clases fundamentales:

Modelo: Administra y maneja todo lo relacionado con los datos del sistema, da respuesta a peticiones de información sobre el estado de la aplicación (normalmente desde la vista), y responde con instrucciones de cambio de estado (usualmente desde el controlador) a la vista.

Vista: Gestiona lo relacionado con mostrar la información al usuario.

Controlador: El controlador interpreta los eventos que son lanzados por la entrada estándar del usuario (normalmente mouse y teclado), informando de los mismos al modelo y/o la vista para que se ejecuten los cambios apropiadamente.

A continuación se muestra una representación gráfica del MVC

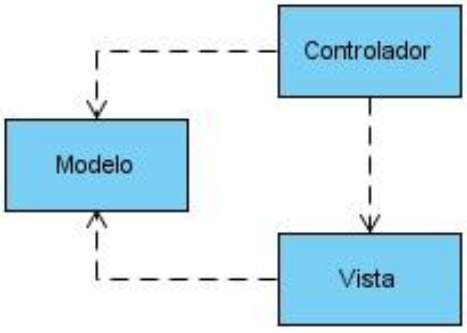


Figura 1.2: Modelo–Vista–Controlador

1.7 Patrones de Diseño

Los patrones de diseño proporcionan una serie de elementos reusables a problemas de desarrollo de software que tienen contextos similares. Nos permiten evitar la búsqueda de soluciones a problemas conocidos y solucionados anteriormente.

1.7.1 Patrones GRASP

GRASP es un acrónimo que significa General Responsibility Asignment Software Patterns. [6] UML y Patrones, Craig Larman. En orientación a objeto resulta complicado seleccionar las clases adecuadas y a

su vez la interacción entre ellas, aún cuando se utiliza una metodología rápida. Es preciso elegir de manera cuidadosa las responsabilidades de cada clase.
Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. [16]

1.7.1.1 Alta Cohesión

1. Una clase de alta cohesión posee una funcionalidad importante y poco trabajo por hacer.
2. En caso de ser necesario comparte el esfuerzo de las tareas con otros objetos.
3. Cada elemento dentro del diseño debe realizar una labor única no realizada por otro elemento.

1.7.1.2 Experto

Propone como solución asignar una responsabilidad al experto en información. La responsabilidad de realizar una labor es de la clase de objetos que contiene la información necesaria para cumplir el compromiso que tiene encomendado.

1.7.1.3 Bajo Acoplamiento

- Plantea la existencia de pocas dependencias entre clases.
- Soporta el diseño de clases independientes, lo que hace posible la reutilización.
- Una herencia poco profunda es uno de los principales síntomas de bajo acoplamiento.

1.7.1.4 Creador

Guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientado a objetos. [15] Su principal intención es encontrar la clase responsable de de crear una nueva instancia de determinada clase.

1.8 Servicios web

Un Servicio Web es una colección de protocolos y estándares que sirve para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los Servicios Web para intercambiar datos en redes de ordenadores como Internet. La interoperabilidad se consigue mediante la adopción de estándares abiertos. [17]

Web Services Description Language (WSDL)

Lenguaje de Descripción de Servicios Web (sus siglas en inglés WSDL) es un lenguaje basado en XML que permite la descripción de los servicios desplegados por una aplicación. Se define un servicio WSDL utilizando los siguientes elementos XML:

- **<portType>** para las operaciones que proporciona el servicio web
- **<message>** para los mensajes que utiliza por el servicio web
- **<types>** para los tipos de datos que utiliza el servicio web
- **<binding>** para los protocolos de comunicaciones que utiliza el servicio web

El documento WSDL tiene una estructura semejante a la que se les muestra a continuación:

```
<definitions>
      <types>
            los tipos de datos…
      </types>

      <message>
            las definiciones del mensaje...
      </message>

      <portType>
            las definiciones de operación ...
      </portType>
```

```
      <binding>
            las definiciones de protocolo...
      </binding>

</definitions>
```

1.9 Conclusiones parciales

En el capítulo se ha demostrado las facilidades que brindan los Sistemas de Control de Presencia para la contabilización de las horas trabajadas y control de los recursos humanos de una determinada empresa, demostrándose así la necesidad que presenta la universidad de un sistema de este tipo.

Para desarrollar la solución informática se ha decidido utilizar como metodología de desarrollo SXP perteneciente al grupo de metodologías ágiles y como metodología para el modelado del negocio BPM. Como lenguaje del lado del servidor se utilizará PHP y del lado del cliente los lenguajes a utilizar serán JavaScript, HTML y CSS. La solución estará hosteada en un servidor Apache 2.2 y el Gestor de BD será PostgreSQL. Se fundamentó la elección de patrones de diseño y arquitectura, framework y herramientas a utilizar para el desarrollo de la solución. Con el objetivo de prestarle servicios a la Intranet 2.0 y al Sistema de Planificación y Control de Alimentos, se empleará el lenguaje WSDL.

Es importante destacar que esta selección se debe a que el cliente es el centro de informatización de la UCI y al ser este un productor de software con un estilo de trabajo propio, la solución informática propuesta se tiene que ajustar a la estrategia de trabajo del centro.

Capítulo II: Características del Sistema

2.1 Introducción

Antes de realizar el desarrollo de un software es importante tener conocimiento de todos los procesos que intervienen en el mismo, para alcanzar así una mayor comprensión del problema a resolver y un mejor entendimiento entre el cliente y el desarrollador.

Para lograr esto, se realiza la modelación del negocio en el presente capítulo. Se plantean en el mismo los requisitos de software mediante la Lista de Reserva del Producto, la cual recoge tanto los requisitos funcionales como los no funcionales, se describe la propuesta de solución para la implementación del sistema y se documentan las HU, estas en función de los requisitos funcionales y se incluye el mapa conceptual del sistema.

2.2 Modelo del negocio

Para el modelado del negocio se empleará la metodología BPM; la utilización de esta permite modelar el negocio de una forma estructurada y sencilla, logrando que el cliente y el desarrollador obtengan una visión clara de los procesos del negocio.

2.2.1 Proceso del negocio

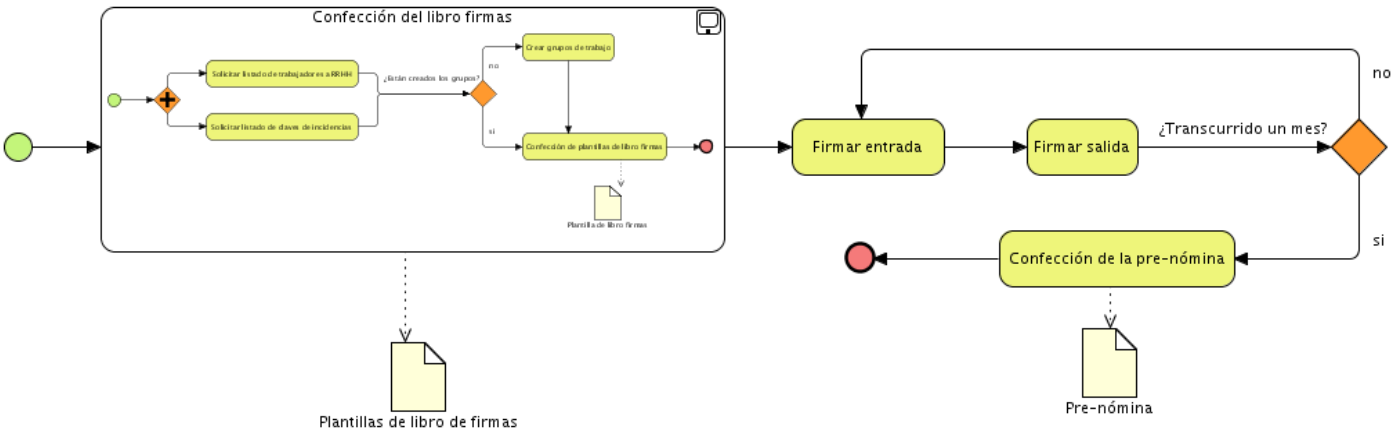


Figura 2.1: Proceso Confeccionar Pre-nómina

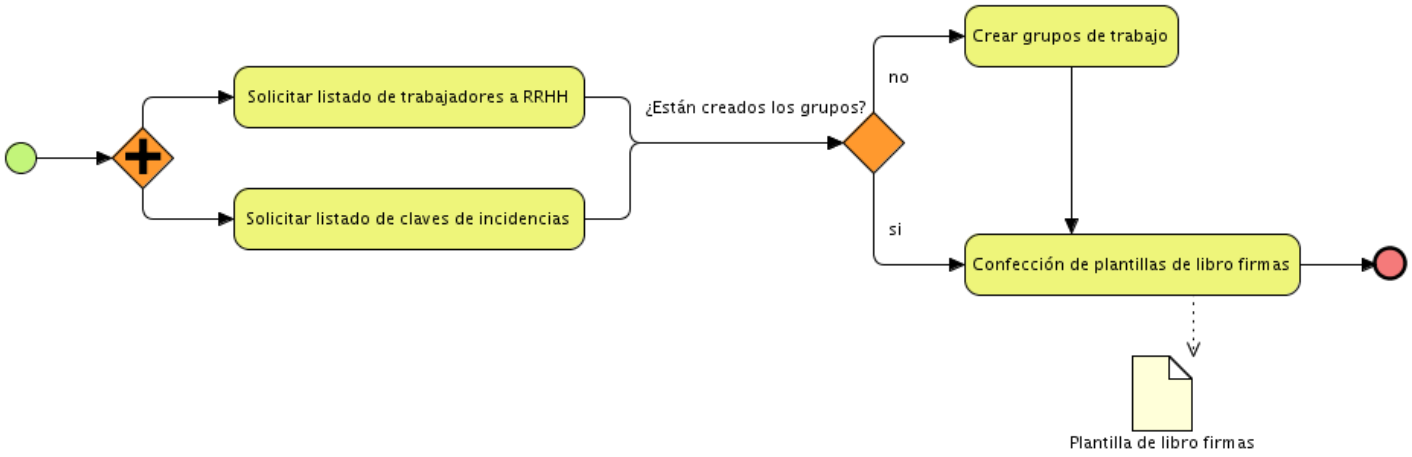


Figura 2.2: Proceso Confeccionar Planillas del Libro de Firmas

2.2.2 Descripción de los procesos del negocio

Ficha de Proceso	
Proceso:	Confección de pre-nómina.
Entradas:	Datos básicos pertenecientes al personal.
Salidas:	Hoja de cálculo que contiene la pre-nómina.
Involucrados:	Especialista de RRHH. Trabajador de RRHH del área. Trabajadores.
Descripción: La confección de la pre-nómina es un proceso que permite automatizar la recopilación de datos de los trabajadores, para el departamento de RRHH.	
1. Confeccionar el libro de firma	El trabajador de RRHH solicita el listado de los trabajadores al departamento de RRHH, a su vez solicita el listado de claves de incidencias previendo actualizaciones. En el caso que los grupos de trabajo, el encargado de RRHH organiza las planillas en las cuales

	firman los trabajadores y confecciona el libro de firmas. Si no están creados los grupos, los crea y luego procede a la confección del libro de firmas. Los datos que contienen estas planillas son: - Nombre y apellidos. - Fecha y hora que el trabajador firmó la entrada. - Fecha y hora que el trabajador firmó la salida. - Ausencia y motivo (justificada o no). - Horas nocturnidad.
2. Firmar entrada	El trabajador firma la hora de entrada en la panilla que le corresponde.
3. Firmar salida.	El trabajador firma la hora de salida en la panilla que le corresponde.
4. Confeccionar la pre-nómina	El trabajador de RRHH del área confecciona la pre-nómina a partir de los datos recopilados en el libro de firmas y otras operaciones. Las operaciones están dadas por: - Nombre y apellidos. - Número de solapín. - Incidencias del tiempo no laborado (clave). - Días feriados (por horas). - Pago a realizar. - Evaluación mensual. - Nocturnidad.
Reglas del negocio: Al cumplirse un mes, es generada la pre-nómina en cada área.	

Tabla 2.1: Ficha de proceso de confección de pre-nómina

2.3 Requerimientos del software

Los requerimientos de software son las condiciones o capacidades que debe cumplir el sistema para que se ajuste a las necesidades del cliente, estos son registrados en un documento o contrato formal llamado Lista de Reserva del Producto (LRP); este documento es un artefacto propuesto por la metodología SXP. Los requisitos de un software son clasificados en dos grupos, Requisitos Funcionales (RF) y Requisitos no Funcionales (RNF).

- **Requerimientos Funcionales (RF)**

  Los requerimientos funcionales definen el comportamiento de un software, estos pueden ser la necesidad de realizar algún cálculo, manipulación de datos, detalles técnicos, etc. La descripción de estos requisitos se especifica en las historias de usuario.

  Estos requisitos son clasificados muy altos, altos y medios, en correspondencia a su importancia en el funcionamiento de la aplicación, además están agrupados por 6 módulos según sus funcionalidades. Estos módulos son:

  - Registro
  - Planificación
  - Personal
  - Configuración
  - Seguridad
  - Estadísticas

  La implementación de los mismos se realizará en dos iteraciones, en la primera iteración un total de 49 que estarán comprendidos en los módulos Personal, Configuración y Seguridad, en la segunda iteración se desarrollarán 17 requisitos que estarán agrupados en los módulos Registro, Planificación y Estadística.

- **Requerimientos no Funcionales (RNF)**

  Los requerimientos no funcionales son propiedades o condiciones que el producto debe tener. Estas características son las que hacen al producto más atrayente, usable, rápido y confiable.

Una vez que se conozca lo que el sistema debe hacer será posible determinar cómo ha de comportarse, qué cualidades debe tener, cuán rápido y grande debe ser el mismo. Dichas propiedades son importantes para los clientes y usuarios a la hora de valorar cuán aceptable está el producto con respecto a usabilidad, seguridad, y cumplimiento de la funcionalidad requerida.

No.	Descripción
Requisitos Funcionales (RF)	
Prioridad Muy Alta	
1.	Crear grupo.
2.	Modificar grupo.
3.	Eliminar grupo.
4.	Mostrar grupo.
5.	Asociar miembro a grupo.
6.	Crear clave de incidencia.
7.	Modificar clave de incidencia.
8.	Mostrar clave de incidencia.
9.	Crear evento.
10.	Modificar evento.
11.	Eliminar evento.
12.	Mostrar evento.
13.	Crear punto de acceso.
14.	Modificar punto de acceso.
15.	Eliminar punto de acceso.
16.	Mostrar punto de acceso.
17.	Asociar punto de acceso.
18.	Crear estructura.
19.	Crear rol.
20.	Modificar rol.
21.	Eliminar rol.
22.	Mostrar rol.

23.	Asociar miembros del rol.
24.	Crear grupo de usuarios.
25.	Modificar grupo de usuarios.
26.	Eliminar grupo de usuarios.
27.	Asociar miembros de grupo de usuarios.
28.	Eliminar miembros de grupo de usuarios.
29.	Mostar grupos de usuario.
30.	Crear usuario.
31.	Modificar usuario.
32.	Eliminar usuario.
33.	Mostrar usuario.
34.	Crear funcionalidad.
35.	Modificar funcionalidad.
36.	Eliminar funcionalidad.
37.	Mostrar funcionalidad.
38.	Crear dominio.
39.	Modificar dominio.
40.	Eliminar dominio.
41.	Mostrar dominio.
42.	Crear tipo de dominio.
43.	Modificar tipo de dominio.
44.	Eliminar tipo de dominio.
45.	Mostrar tipo de dominio.
46.	Agregar trabajador.
47.	Modificar trabajador.
48.	Mostrar trabajador.
49.	Autenticar usuario.
Prioridad Alta	
50.	Crear calendario.
51.	Modificar calendario.

52.	Eliminar calendario.
53.	Mostrar calendario.
54.	Asignar calendario a grupo.
55.	Asignar calendario a trabajador.
56.	Registrar firma de entrada.
57.	Registrar firma de salida.
58.	Mostrar ausentes.
59.	Mostrar estadísticas generales de trabajador.
60.	Mostrar estadísticas de ausencia del trabajador.
61.	Mostrar estadísticas de entrada/salida del trabajador.
62.	Mostrar estadísticas generales del grupo.
63.	Mostrar estadísticas de ausencia del grupo.
64.	Ver estadísticas de ausencia de trabajador.
65.	Mostrar pre-nómina.
Prioridad Media	
66.	Brindar servicios a otros sistemas.
Requisitos No Funcionales (RNF)	
Usabilidad	
1.	*Facilidad de uso por parte de los usuarios:* el sistema debe presentar una interfaz amigable que permita la fácil interacción con el mismo y llegar de manera rápida y efectiva a la información buscada. Debe, además, ser una interfaz de manejo cómodo que posibilite a los usuarios sin experiencia una rápida adaptación.
2.	*Especificación de la terminología utilizada:* el sistema debe adaptarse al lenguaje y términos utilizados por los clientes en la rama abordada con vista a una mayor comprensión por parte del cliente de la herramienta de trabajo.
3.	*Emplear perfiles de usuario:* diferenciar las interfaces y opciones para los usuarios que accedan al sistema según los diferentes roles que estos tengan dentro del sistema (administrador, jefe de área, jefe de grupo, RRHH entidad (UCI), encargado de RRHH y usuarios).
4.	*Menús:* el sistema debe presentar una serie de menús tanto laterales como en barra de iconos

	flotantes que permitan el acceso rápido a la información por parte de los usuarios, aprovechando así las potencialidades de estas estructuras.
Fiabilidad	
5.	*Seguridad de la base de datos:* la base de datos deberá estar fraccionada en esquemas que permitan un mejor uso de la información y la división de forma lógica de las funcionalidades del sistema, trayendo consigo además la protección de la información al ocurrir un incidente sobre una parte de la base de datos. El SGBD escogido debe presentar facilidades de administración de roles y usuarios restringiendo el acceso a los datos.
6.	*Servicios Web restringidos:* los servicios Web que brinde el sistema deben estar restringidos a grupos de usuarios definidos y aprobados previamente.
7.	*Políticas de seguridad por usuarios y roles:* el sistema debe contar con un grupo de políticas de accesibilidad a las diferentes funcionalidades del mismo en dependencia del nivel de autorización que presente un usuario determinado.
8.	*Registro sistemáticos de incidencias:* el sistema debe ser capaz de registrar el accionar del usuario, así como permitir auditorías y exámenes de las trazas tanto en tiempo real como en históricos. Se precisa un monitor de incidencia para la visualización y tratamiento de las mismas.
9.	*Alta protección de los datos:* al estar trabajando con información sensible, se hace necesario una alta protección de los datos a nivel de aplicación y de tráfico de red, para tal fin se ha definido además la seguridad en varios niveles dentro de la aplicación: nivel de interfaz, nivel de acceso a datos y nivel de base de datos.
Eficiencia	
10.	El sistema debe soportar un tiempo de respuesta menor o igual a 5 segundos.
11.	El sistema debe soportar una conexión simultanea de más de 10 000 usuarios.
Soporte	
12.	Grupo de soporte y asesoría: el sistema contará con un grupo de soporte y asesoría al cliente del producto, destinado a brindar asesoría y soporte técnico al mismo.

Restricciones de diseño	
13.	Lenguaje de programación: PHP 5.1 o superior.
14.	El marco de trabajo base de desarrollo que se utilizará es: CodeIgniter 1.7.2
15.	Como IDE se empleará NetBeans 6.7. 1
16.	Como servidor Web se explotará Apache 2.2.2
17.	El SGDB deberá ser PostgreSQL 8.4.1
18.	El diseño de la base de datos se realizará con Visual Paradigm 3.0
19.	El sistema operativo a utilizar en el entorno de desarrollo deberá ser: GNU Linux.
20.	El repositorio principal, el entorno de prueba y el servidor de base de datos estarán montados sobre Ubuntu Server 9.04 o superior.
Requisitos para la documentación de usuarios en línea y ayuda del sistema	
21.	*Documentación actualizada del grupo de desarrollo:* se precisa que la documentación del sistema esté actualizada en todos los aspectos, fases de trabajo y ciclos de desarrollo del mismo, permitiendo con ello un respaldo tanto ingenieril como legal del desarrollo de dicho sistema.
Interfaz	
22.	*Interfaz Web:* la interfaz deberá ser sencilla con colores suaves a la vista y sin cúmulo de imágenes u objetos que distraigan al cliente del objetivo de su empleo.
23.	*Interfaz interna:* la interfaz interna estará determinada por los desarrolladores, construyendo así una vista escalable de las clases o agrupaciones de clases que permitirán un mejor encapsulamiento de las funcionalidades y una mayor abstracción modular del sistema.
Interfaces de Hardware	
24.	Para el desarrollo: PC Intel Pentium 4 o superior, CPU 3GHZ o superior, 512 MB RAM o superior, 160 GB HDD o superior.
25.	Para explotación del cliente: PC Pentium 3 o superior, CPU 133 MHZ o superior, 256 RAM mínimo 512 RAM recomendada o superior.

26.	Para explotación del servidor: CPU Dual Core 2.0 GHZ o superior, memoria RAM de 4 GB (recomendado 6 GB), 250 GB HDD.
Requisitos Legales, de Derecho de Autor y otros	
27.	El sistema debe ser sometido a una evaluación y certificación por parte del cliente del producto.

Tabla 2.2: Lista de Reserva del Producto

2.4 Historias de usuario (HU)

Las HU es un artefacto propuesto por la metodología SXP con el objetivo de describir los requisitos del sistema. El cliente es el encargado de la realización de la misma aunque preferentemente con la presencia de un desarrollador para que exista una mejor comprensión entre ambas partes. A las HU se les asigna un número único, un nombre, el programador encargado de desarrollar el requisito, una breve descripción de su funcionamiento, entre otros datos.

Las historias de usuario es un artefacto a generar en la metodología SXP, cada una corresponde a un requisito funcional y guarda en ella la descripción, iteración a la que está asignada, desarrollador encargado de programar la misma, prioridad que tiene y otros aspectos de la misma.

Historia de Usuario	
Número: 46	**Nombre Historia de Usuario:** Agregar trabajador.
Modificación de Historia de Usuario Número: 1	
Referencia: Agregar trabajador.	
Programador: Lisbet Torres Triana	**Iteración Asignada:** 1 iteración
Prioridad: Muy Alta	**Puntos Estimados:** 0.2
Riesgo en Desarrollo: Medio	**Puntos Reales:** 0.2
Descripción: La historia de usuario permite agregar un trabajador. Para agregar un nuevo trabajador se selecciona en el área de iconos flotantes, la acción adicionar. Se puede agregar mediante dos opciones:	

De manera manual: el usuario define Nombre y Apellidos, Carnet de identidad y solapín.

De manera automática: el usuario selecciona la URL desde donde lo importará y los atributos que desee importar.

El trabajador puede estar activo o no.

Una vez agregado el trabajador, se actualiza el listado de trabajadores y se muestra el mensaje: ¨El elemento ha sido agregado satisfactoriamente¨.

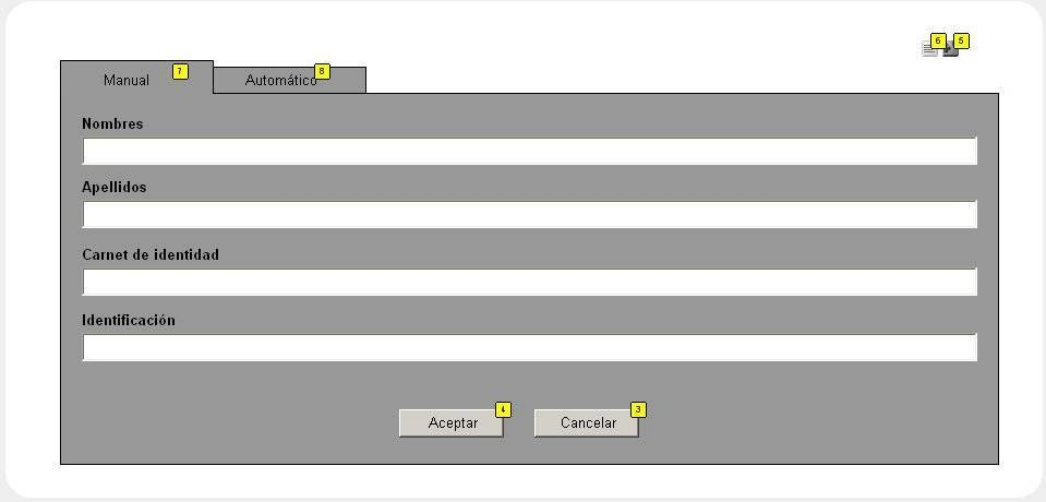
Observaciones:

En caso de que el elemento exista se muestra un mensaje de error “El elemento ya existe”.

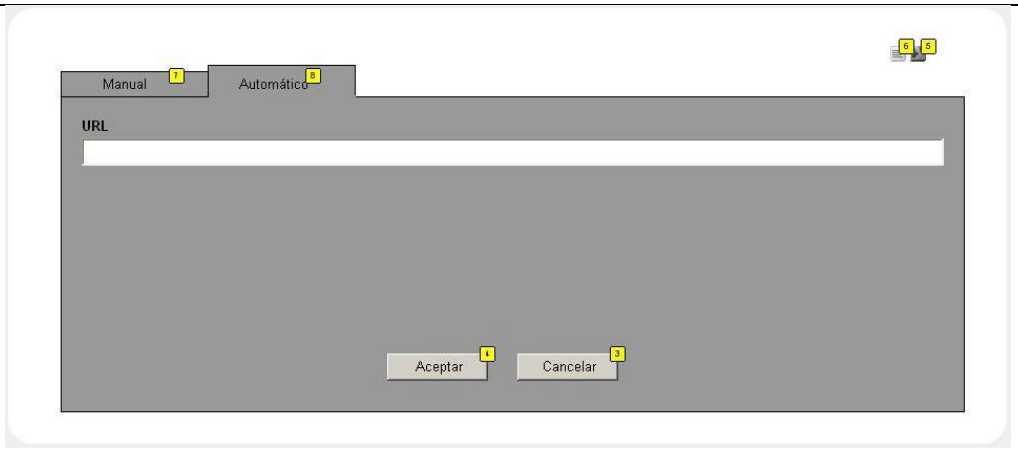
En caso de cancelar la acción se muestra el listado de elementos.

Interactúa con esta acción RRHH entidad (UCI).

Prototipo de interfaz:



IU50. Agregar trabajador de manera manual



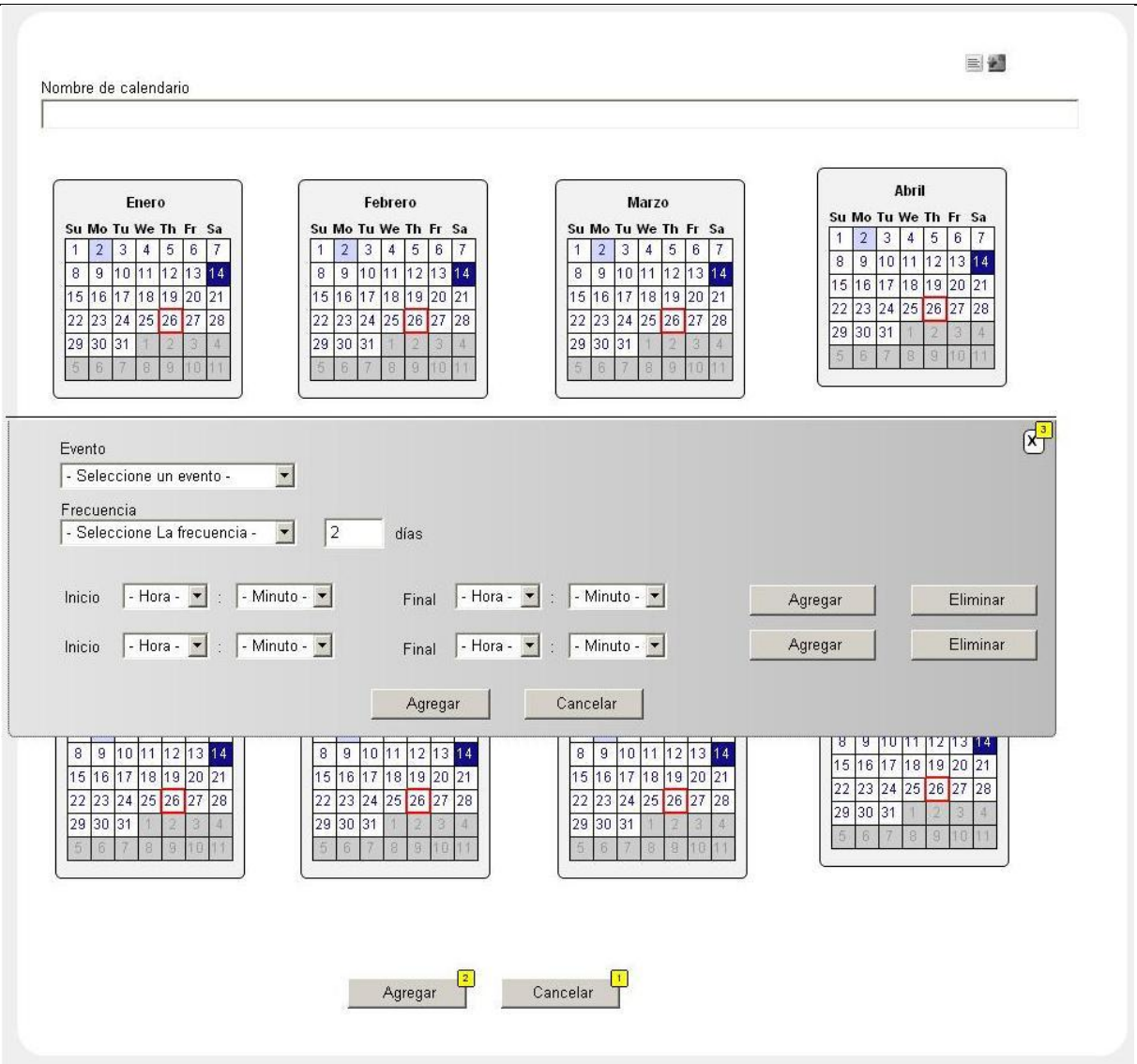
IU51. Seleccionar URL

IU52. Agregar trabajador de manera automática

Tabla 2.3: HU Agregar trabajador

Historia de Usuario	
Número: 50	**Nombre Historia de Usuario:** Crear calendario.
Modificación de Historia de Usuario Número: 1	

Referencia: Crear calendario.	
Programador: Yoan Trujillo Reyna	**Iteración Asignada:** 2 iteración
Prioridad: Alta	**Puntos Estimados:** 0.6
Riesgo en Desarrollo: Medio	**Puntos Reales:** 0.6
Descripción: La historia de usuario permite crear un calendario. Para crear un nuevo calendario se selecciona en el área de iconos flotantes, la acción adicionar. El usuario define Nombre, selecciona un Día del mes, le asocia a este día un Evento, tiene la opción de seleccionar la Frecuencia (todos los días, cada # días, este mes, todo el año) con que se aplicará el evento y define la Duración del evento (fecha inicio, fecha fin). Una vez agregado el calendario, se actualiza el listado de calendarios y se muestra el mensaje: ¨El elemento ha sido creado satisfactoriamente¨.	
Observaciones: Este calendario se realiza para el año fiscal. En caso de que el elemento exista se muestra un mensaje de error “El elemento ya existe”. En caso de cancelar la acción se muestra el listado de elementos. Interactúa con esta acción el Jefe de área y encargado de RRHH.	
Prototipo de interfaz:	

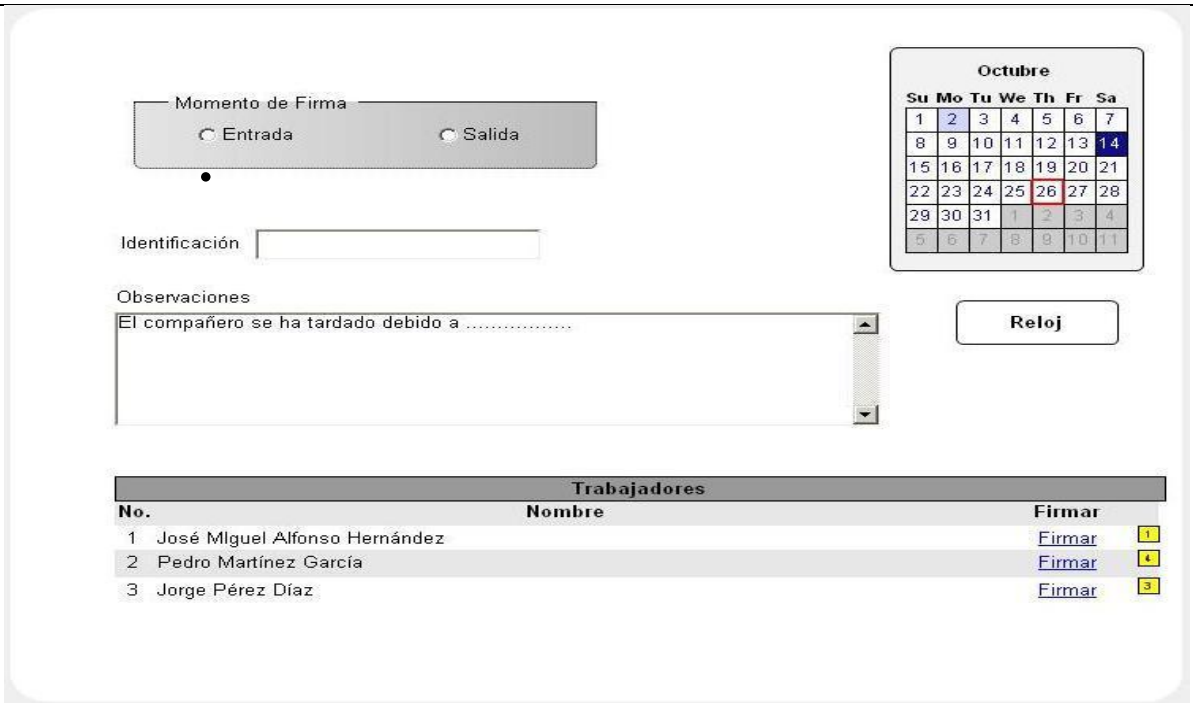


IU56. Crear calendario

Tabla 2.4: HU Crear calendario

Historia de Usuario	
Número: 56	**Nombre Historia de Usuario:** Registrar firma entrada.
Modificación de Historia de Usuario Número: 1	

Referencia: Registrar firma entrada.	
Programador: Yoan Trujillo Reyna	**Iteración Asignada:** 2 iteración
Prioridad: Alta	**Puntos Estimados:** 0.4
Riesgo en Desarrollo: Medio	**Puntos Reales:** 0.4
Descripción: La historia de usuario permite registrar la firma de entrada, que puede ser mediante identificación o manual. Y en la que se indica en caso de tardanza una observación. *Mediante credencial:* marca el solapín. *De manera Manual:* se muestra un listado de los trabajadores para registrar la asistencia. En ambos casos, al firmar, se muestra un mensaje mostrándose los datos del trabajador durante 3 segundos. Estos datos son: Nombre y apellidos, Grupo al que pertenece, hora de fichaje y si tiene permiso (por colores).	
Observaciones: Si un trabajador firma mediante credencial, su nombre desaparecerá automáticamente del listado de los trabajadores mostrado. El listado contiene los trabajadores que se encuentran activos y los que tengan asignado un calendario. Interactúa con esta acción el Jefe de área y encargado de RRHH.	
Prototipo de interfaz:	

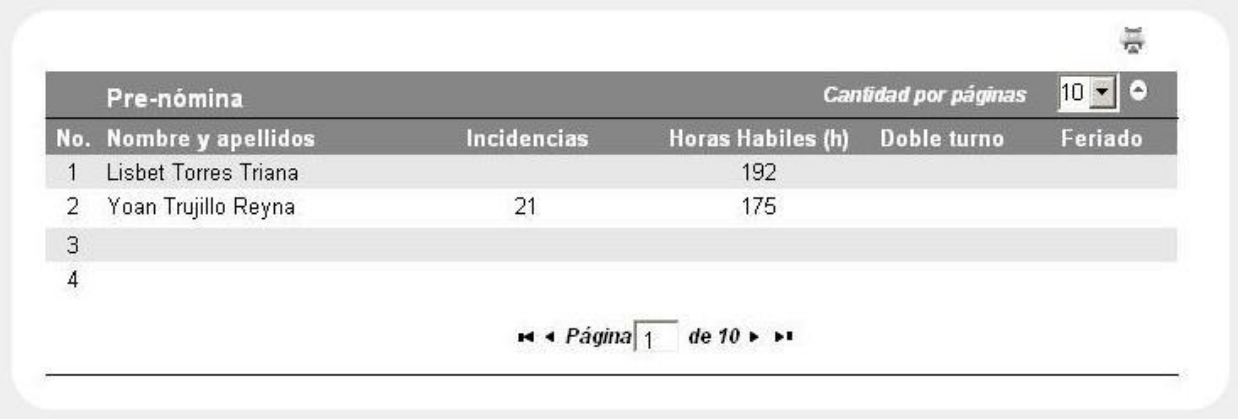


IU62. Registrar firma entrada

Tabla 2.5: HU Registrar firma entrada

Historia de Usuario	
Número: 65	**Nombre Historia de Usuario:** Mostrar pre-nómina.
Modificación de Historia de Usuario Número: 1	
Referencia: Mostrar pre-nómina.	
Programador: Lisbet Torres Triana	**Iteración Asignada:** 2 iteración
Prioridad: Alta	**Puntos Estimados:** 0.2
Riesgo en Desarrollo: Medio	**Puntos Reales:** 0.2
Descripción: La historia de usuario muestra la pre-nómina del mes con los datos: - Solapín - Nombre y Apellidos	

- Incidencias - Nocturnidad El usuario puede imprimir el reporte.
Observaciones: Interactúa con esta acción el Jefe de área.
Prototipo de interfaz:



IU69. Mostrar pre-nómina

Tabla 2.6: HU Mostrar pre-nómina

2.4 Mapa Conceptual

Los mapas conceptuales son representaciones gráficas que permiten transmitir con claridad mensajes conceptuales. Su objetivo es representar las relaciones existentes entre conceptos.

La metodología usada en el presente trabajo, no propone un artefacto que represente los conceptos básicos del sistema, pero el cliente (Centro de Informatización) si lo emplea tanto para esto, como para facilitar luego la confección del diseño de la base de datos.

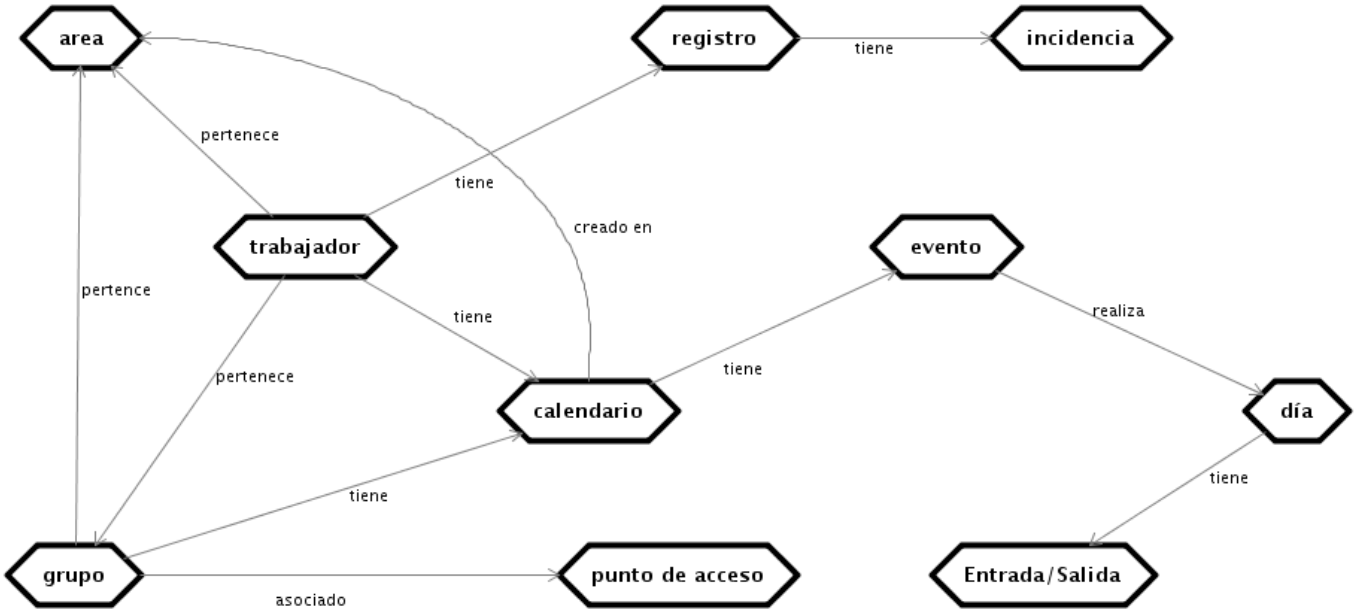


Figura 2.3: Mapa conceptual

2.5 Conclusiones parciales

Para la realización de un software con calidad es necesario tener un conocimiento basto de los principales procesos que intervienen en la investigación y además una amplia visión del desarrollo del proyecto. Por lo cual en este capítulo se reflejaron los principales procesos del negocio, los requerimientos funcionales y no funcionales que representan las características fundamentales que requiere la aplicación a desarrollar, la descripción de las historias de usuarios y el mapa conceptual del sistema.

Capítulo III: Implementación

3.1 Introducción

En el presente capítulo se estructura la información que se desea persista, a través del diseño de la base de datos. Se realiza la descripción de las tareas de ingeniería, las cuales indican al programador como se procederá a desarrollar cada funcionalidad.

3.2 Modelo de Datos

El modelo de datos es el enfoque utilizado para la representación de las entidades y sus características dentro de la base de datos. El objetivo fundamental de este modelo es definir las estructuras permitidas y las restricciones, lo cual constituye un elemento básico para el posterior diseño de la base de datos.

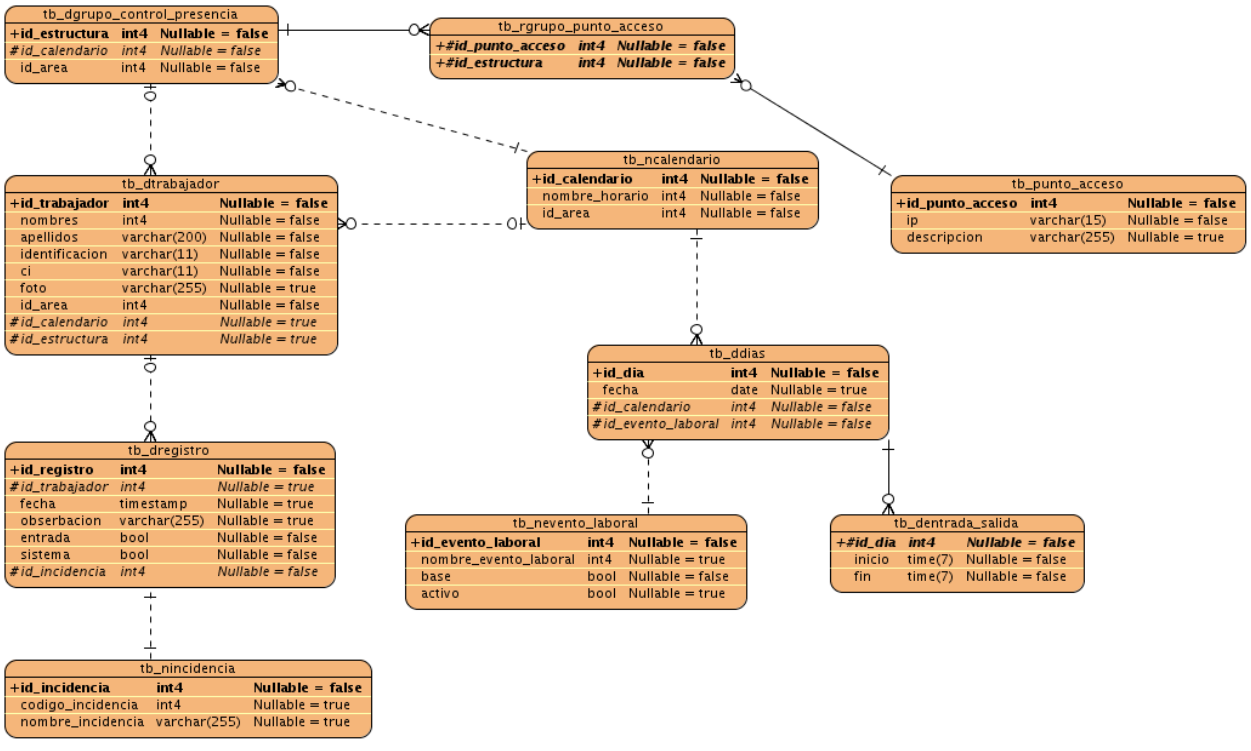


Figura 3.1: Modelo de Datos

3.3 Tareas de Ingeniería

Las tareas de ingeniería son un artefacto que le indican al programador como implementar la funcionalidad correspondiente. Ellas comprenden datos como el tipo de tarea que es, el programador responsable de realizarla y la descripción de lo que se debe hacer.

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** 1
Nombre Tarea: Diseñar Base de datos.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 0.1
Fecha Inicio: 25/02/2010	**Fecha Fin:** 25/02/2010
Programador Responsable: Yoan Trujillo Reyna Lisbet Torres Triana	
Descripción: Diseñar el modelo de datos que se usará para almacenar la información.	

Tabla 3.1: TI Diseñar Base de datos

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** 1
Nombre Tarea: Generar las clases modelos.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 0.1
Fecha Inicio: 26/02/2010	**Fecha Fin:** 26/02/2010
Programador Responsable: Yoan Trujillo Reyna Lisbet Torres Triana	
Descripción: Se generan las clases modelos utilizando el script *php_gen.php*, especificándole el servidor de base de datos, el nombre de la base de datos, el puerto a usar, el usuario y la contraseña para autenticar.	

Tabla 3.2: TI Generar las clases modelos

Tarea de Ingeniería	
Número Tarea: 131	**Número Historia de Usuario:** 46
Nombre Tarea: Implementar manager.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 27/03/2010	**Fecha Fin:** 27/03/2010
Programador Responsable: Yoan Trujillo Reyna	
Descripción: Se implementa la clase *personal_mng* que agrupa por funcionalidad las clases modelos generadas.	

Tabla 3.3: TI Implementar manager (Módulo Personal)

Tarea de Ingeniería	
Número Tarea: 132	**Número Historia de Usuario:** 46
Nombre Tarea: Confeccionar la interfaz de usuario.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 27/03/2010	**Fecha Fin:** 27/03/2010
Programador Responsable: Yoan Trujillo Reyna	
Descripción: Crear la interfaz de usuario *crear trabajador* con los componentes: • *Cuadro de texto* para el *nombre del trabajador.* • *Cuadro de texto* para el *solapín del trabajador.* • *Cuadro de texto* para el *carnet de identidad del trabajador.* • *Cuadro de texto* para el *área a la que pertenece del trabajador.* • *Cuadro de selección simple* para activar o desactivar al trabajador.	

Tabla 3.4: TI Confeccionar la interfaz de usuario (HU Agregar trabajador)

Tarea de Ingeniería	
Número Tarea: 133	**Número Historia de Usuario:** 46
Nombre Tarea: Implementar los métodos en la controladora.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 27/03/2010	**Fecha Fin:** 27/03/2010
Programador Responsable: Yoan Trujillo Reyna	

Descripción:

Se implementan en la controladora *trabajador* los métodos *crear* y *crearTrabajador.*

- *crear():* Se realiza una llamada a la interfaz *crear grupo.*
- *crearTrabajador():* Valida los datos recibidos de la interfaz y realiza la llamada al método *crearTrabajador* de la librería *trabajador_lib* y le pasa los atributos validados. Espera el resultado de la acción y crea el mensaje que envía a la interfaz.

Validación:

- El atributo *nombre del trabajador* no admite caracteres extraños como: @#$%^&* y no puede tener un valor nulo.
- El atributo *solapín del trabajador* no admite caracteres extraños como: @#$%^&* y no puede tener un valor nulo.
- El atributo *carnet de identidad del trabajador* no admite caracteres extraños como: @#$%^&*, tiene que tener 11 dígitos (no más, no menos) y no puede tener un valor nulo.
- El atributo *área del trabajador* no admite caracteres extraños como: @#$%^&* y no puede tener un valor nulo.

Tabla 3.5: TI Implementar los métodos en la controladora (HU Agregar trabajador)

Tarea de Ingeniería	
Número Tarea: 134	**Número Historia de Usuario:** 46
Nombre Tarea: Implementar método en la librería.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1

Fecha Inicio: 27/03/2010	**Fecha Fin:** 27/03/2010
Programador Responsable: Yoan Trujillo Reyna	
Descripción: Se implementa el método *crearTrabajador* en la librería *trabajador_lib* que recibe los datos, crea el arreglo a insertar y hace una llamada al método de la modelo desde el manager, pasándole dicho arreglo.	

Tabla 3.6: TI Implementar métodos en la librería (HU Agregar trabajador)

Tarea de Ingeniería	
Número Tarea: 145	**Número Historia de Usuario:** 50
Nombre Tarea: Implementar manager.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 2/04/2010	**Fecha Fin:** 2/04/2010
Programador Responsable: Yoan Trujillo Reyna	
Descripción: Se implementa la clase *planificacion_mng* que agrupa por funcionalidad las clases modelos generadas.	

Tabla 3.7: TI Implementar manager (Módulo Planificación)

Tarea de Ingeniería	
Número Tarea: 146	**Número Historia de Usuario:** 50
Nombre Tarea: Confeccionar la interfaz de usuario.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 2/04/2010	**Fecha Fin:** 2/04/2010
Programador Responsable: Yoan Trujillo Reyna	
Descripción: Crear la interfaz de usuario *crear calendario* con los componentes:	

• *Calendario* (12) para seleccionar días. • *Cuadro de selección simple* para seleccionar el evento. • *Cuadro de selección simple* para seleccionar la frecuencia del evento. • *Cuadro de selección simple (5)* para seleccionar la fecha de entrada/salida.

Tabla 3.8: TI Confeccionar la interfaz de usuario (HU Crear calendario)

Tarea de Ingeniería	
Número Tarea: 147	**Número Historia de Usuario:** 50
Nombre Tarea: Implementar los métodos en la controladora.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 03/04/2010	**Fecha Fin:** 05/04/2010
Programador Responsable: Yoan Trujillo Reyna	
Descripción: Se implementan en la controladora *planificacion* los métodos *crear* y *crearCalendario.* • *crear():* Se realiza una llamada a la interfaz *crear planificacion.* • *crearCalendario():* Valida los datos recibidos de la interfaz y realiza la llamada al método *crearCalendario* de la librería *calendario_lib* y le pasa los atributos validados. Espera el resultado de la acción y crea el mensaje que envía a la interfaz. Validación: • Los atributos no pueden tener valor nulo. • La fecha de inicio tiene que ser menor que la fecha de fin.	

Tabla 3.9: TI Implementar los métodos en la controladora (HU Crear calendario)

Tarea de Ingeniería	
Número Tarea: 148	**Número Historia de Usuario:** 50
Nombre Tarea: Implementar método en la librería.	

Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 03/04/2010	**Fecha Fin:** 05/04/2010
Programador Responsable: Yoan Trujillo Reyna	
Descripción: Se implementa el método *crearCalendario* en la librería *calendario_lib* que recibe los datos, crea el arreglo a insertar y hace una llamada al método de la modelo desde el manager, pasándole dicho arreglo.	

Tabla 3.10: TI Implementar métodos en la librería (HU Crear calendario)

Tarea de Ingeniería	
Número Tarea: 163	**Número Historia de Usuario:** 56
Nombre Tarea: Implementar manager.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 16/04/2010	**Fecha Fin:** 16/04/2010
Programador Responsable: Yoan Trujillo Reyna	
Descripción: Se implementa la clase *registro_mng* que agrupa por funcionalidad las clases modelos generadas.	

Tabla 3.11: TI Implementar manager (Módulo Registro)

Tarea de Ingeniería	
Número Tarea: 164	**Número Historia de Usuario:** 56
Nombre Tarea: Confeccionar la interfaz de usuario.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 16/04/2010	**Fecha Fin:** 16/04/2010
Programador Responsable: Yoan Trujillo Reyna	
Descripción:	

Crear la interfaz de usuario *crear registro entrada* con los componentes: • *Calendario* general. • *Selección simple* para seleccionar el momento de firma (entrada). • *Cuadro de texto* donde se introduce la credencial. • *Área de texto* para indicar observaciones. • Rejilla que contiene la información de los trabajadores y la opción firmar.

Tabla 3.12: TI Confeccionar la interfaz de usuario (HU Registrar firma entrada)

Tarea de Ingeniería	
Número Tarea: 165	**Número Historia de Usuario:** 56
Nombre Tarea: Implementar los métodos en la controladora.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 16/04/2010	**Fecha Fin:** 17/04/2010
Programador Responsable: Yoan Trujillo Reyna	
Descripción: Se implementan en la controladora *registro* los métodos *crear* y *crearFirmaE.* • *crear():* Se realiza una llamada a la interfaz *crear registro.* • *crearFirmaE():* Valida los datos recibidos de la interfaz y realiza la llamada al método *crearFirmaE* de la librería *registro_lib* y le pasa los atributos validados. Espera el resultado de la acción y crea el mensaje que envía a la interfaz. Validación: • Debe estar seleccionado el momento de la firma para poder efectuarse la misma. • Los atributos no pueden tener valor nulo.	

Tabla 3.13: TI Implementar los módulos en la controladora (HU Registrar firma entrada)

Tarea de Ingeniería	
Número Tarea: 166	**Número Historia de Usuario:** 56
Nombre Tarea: Implementar método en la librería.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 16/04/2010	**Fecha Fin:** 17/04/2010
Programador Responsable: Yoan Trujillo Reyna	
Descripción: Se implementa el método *crearFirmaE* en la librería *registro_lib* que recibe los datos, crea el arreglo a insertar y hace una llamada al método de la modelo desde el manager, pasándole dicho arreglo.	

Tabla 3.14: TI Implementar los módulos en la librería (HU Registrar firma entrada)

Tarea de Ingeniería	
Número Tarea: 170	**Número Historia de Usuario:** 65
Nombre Tarea: Confeccionar la interfaz de usuario.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 07/04/2010	**Fecha Fin:** 07/04/2010
Programador Responsable: Lisbet Torres Triana	
Descripción: Crear la interfaz de usuario *mostrar prenomina* con los componentes: • Rejilla que contiene, cuadro de selección simple para mostrar la cantidad de contenido por página e iconos para exportar a Excel / PDF.	

Tabla 3.15: TI Confeccionar la interfaz de usuario (HU Mostrar pre-nómina)

Tarea de Ingeniería	
Número Tarea: 171	**Número Historia de Usuario:** 65
Nombre Tarea: Implementar método en la controladora.	

Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 07/04/2010	**Fecha Fin:** 07/04/2010
Programador Responsable: Lisbet Torres Triana	
Descripción: Se implementan en la controladora *estadisticas* el método *mostrar.* • *mostrar():* Obtiene los datos del método *obtenerPrenomina()* de la librería *estadisticas_lib,* se pasan estos datos a la interfaz y realiza la llamada a la interfaz mostrar prenomina.	

Tabla 3.16: TI Implementar los módulos en la controladora (HU Mostrar pre-nómina)

Tarea de Ingeniería	
Número Tarea: 172	**Número Historia de Usuario:** 65
Nombre Tarea: Implementar método en la librería.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 07/04/2010	**Fecha Fin:** 07/04/2010
Programador Responsable: Lisbet Torres Triana	
Descripción: Se implementa el método *mostrarPrenomina* en la librería *estadisticas_lib* que recibe los datos, crea el arreglo a mostrar y hace una llamada al método de la modelo desde el manager, pasándole dicho arreglo.	

Tabla 3.17: TI Implementar los módulos en la librería (HU Mostrar pre-nómina)

3.4 Servicios

La aplicación web brinda servicios tales como:

- Obtener horas trabajadas dado el id del trabajador.
- Obtener incidencias dado el id del trabajador.
- Obtener Entradas/Salidas dado el id del trabajador.
- Obtener días trabajados dado el id del trabajador.

- Obtener pre-nomina dado el grupo.
- Obtener trabajadores dado su identificación.
- Obtener trabajadores dado registro de firma.

3.5 Modelo de despliegue

El modelo de despliegue define la arquitectura física del sistema. Se usa para modelar de manera detallada los nodos físicos y las asociaciones de comunicación que existen entre ellos. Del mismo modo queda especificado qué hardware, sistemas operativos, software de interfaces y soporte conformarán el nuevo sistema.

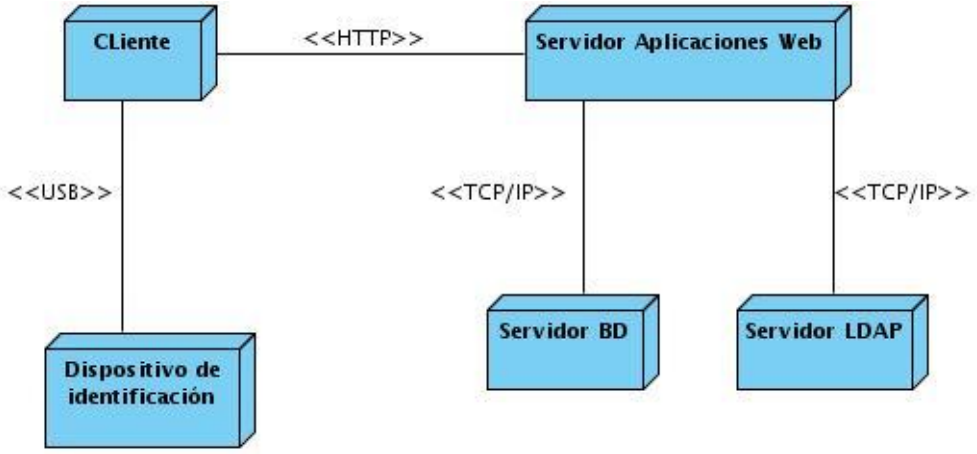


Figura 3.2: Modelo de despliegue

3.6 Conclusiones parciales

A lo largo de este capítulo se han obtenido los artefactos que dan lugar al modelo de implementación final. El modelo de datos, con el objetivo de facilitar posteriormente el diseño de la base de datos. Las tareas de ingeniería, las cuales detallan el procedimiento a llevar a cabo para programar las funcionalidades del software y el modelo de despliegue que describe las configuraciones sobre las cuales deberá ser implementado el sistema.

Conclusiones

El desarrollo de este trabajo de tesis está orientado a la concepción de una herramienta informática para la gestión del control del horario laboral de los trabajadores. El valor fundamental del sistema se expresa en la contribución a simplificar el trabajo y la demora que produce el procesamiento manual de la información y mejorar la gestión de las actividades en esta esfera.

Se demostró la efectividad de los lenguajes y tecnologías utilizadas para el desarrollo del sistema. Además se alcanzó, satisfactoriamente, el objetivo propuesto: desarrollar una solución informática que respalde el control del horario laboral de la UCI; reafirmando así la utilidad y validez de emplear las tecnologías informáticas para apoyar las labores que se desarrollan en cualquier tipo de esfera. Se implementaron en la misma los servicios al portal de la intranet de la Universidad de las Ciencias Informáticas y al Sistema de Control de Alimentos de la misma entidad.

Recomendaciones

Para facilitar la obtención de mejores resultados, una vez concluido el desarrollo de este trabajo investigativo, se recomienda:

- Poner a prueba el sistema durante un período de tiempo significativo, para comprobar su desempeño y que las funcionalidades del sistema se correspondan con la actividad que se está gestionando.
- Hacer extensivo el sistema, para el control de los trabajadores con horario abierto con el objetivo de aplicarlo en todas las áreas de la universidad.
- Continuar el estudio para añadir nuevas funcionalidades y de esta manera pueda ser empleado para el control de actividades establecidas para los estudiantes de la universidad.
- Realizar un estudio sobre los dispositivos biométricos para el desarrollo de futuras aplicaciones de control de acceso y presencia en la universidad.

Referencias bibliográficas

[1] **Levant, Yves y Nikitin, Marc.** Palissy - Fac. Lettres & Sc. Humaines - Nantes. Palissy - Fac. *Lettres & Sc. Humaines - Nantes.* [En línea] [Citado el: 23 de enero del 2010] http://palissy.humana.univ-nantes.fr/msh/wcah/textes/levant-nikitin.pdf.

[2] **Empresa Pablo Huc, S.A.** Empresa Pablo Huc, S.A. [En línea] Empresa Pablo Huc, S.A. [Citado el: 23 de enero del 2010] http://www.huelladactilar.info/phuc/control_presencia/phucipedia_control_presencia.htm.

[3] **WebClocks.** WebClocks. [En línea] WebClocks. [Citado el: 23 de enero del 2010] http://www.agw.es/index-5.html.

[4] **IDC S.A.** CalzaSano. *CalzaSano.* [En línea] [Citado el: 25 de enero del 2010] http://codeconet.com/relojeria-industrial/**id3300.pdf** .

[5] **Ingeniería LERP Ltda.** Ingeniería LERP Ltda. [En línea] Ingeniería LERP Ltda. [Citado el: 23 de enero del 2010] http://www.lerp.cl/reloj%20control%20asistencia%20tradicional%20con%20tarjetas%20de%20carton.htm

[6] **Kimaldi.** Kimaldi. [En línea] Kimaldi. [Citado el: 23 de enero del 2010] http://www.kimaldi.com/productos/control_de_acceso/control_de_acceso_on_line/terminal_kimaldi_biomax2.

[7] **codigodebarra.com.** codigodebarra.com. [En línea] codigodebarra.com. [Citado el: 23 de enero del 2010] http://www.codigodebarras.com/tema.php?ID=para_que_sirve

[8] **Frontline.** Frontline. [En línea] Frontline. [Citado el: 23 de enero del 2010] http://www.frontlineco.net/?p=126.

[9] **Azerty.** Azerty. [En línea] Azerty. [Citado el: 23 de enero del 2010] http://www.area.com.mx/azerty/lectorestec.html .

[10] **Meneses Abad, Abel; Marsi Peñalver Romero, Gladys**. *SXP, Metodología ágil para proyectos de software libre*. La Habana, 2009.

[11] **Leyva Samada, Lisandra Isabel**. *Flujo de investigación para la metodología ágil*. La Habana, 2009.

[12] **degerencia.com.** *degerencia.com.* [En línea] degerencia.com. [Citado el: 27 de enero del 2010] http://www.degerencia.com/articulo/business_process_management_bpm_articulando_estrategia_procesos_y_tecnologia.

[13] **Cibernética.** *Cibernética*. [En línea] Cibernética. [Citado el: 27 de enero del 2010] http://www.cibernetia.com/manuales/instalacion_servidor_web/1_conceptos_basicos.php

[14] **desarrolloweb.com.** *desarrolloweb.com.* [En línea] desarrolloweb.com. [Citado el: 2 de febrero del 2010] http://www.desarrolloweb.com/articulos/1112.php

[15] **James Garret, Jesse**. Maestros del web. *Maestros del web*. [En línea]. [Citado el: 2 de febrero del 2010] http://www.maestrosdelweb.com/editorial/ajax/

[16] **Larman, Craig.** *UML y Patrones.* La Habana: Félix Varela, 2004.

[17] La Oficina del W3C en España. *Guía Breve de Servicios Web.* [En línea]

Bibliografía

1. **Levant, Yves y Nikitin, Marc.** Palissy - Fac. Lettres & Sc. Humaines - Nantes. Palissy - Fac. *Lettres & Sc. Humaines - Nantes.* [En línea] [Citado el: 23 de enero del 2010] http://palissy.humana.univ-nantes.fr/msh/wcah/textes/levant-nikitin.pdf.
2. **Empresa Pablo Huc, S.A.** Empresa Pablo Huc, S.A. [En línea] Empresa Pablo Huc, S.A. [Citado el: 23 de enero del 2010] http://www.huelladactilar.info/phuc/control_presencia/phucipedia_control_presencia.htm.
3. **WebClocks.** WebClocks. [En línea] WebClocks. [Citado el: 23 de enero del 2010] http://www.agw.es/index-5.html.
4. **IDC S.A.** CalzaSano. *CalzaSano.* [En línea] [Citado el: 25 de enero del 2010] http://codeconet.com/relojeria-industrial/**id3300.pdf** .
5. **Ingeniería LERP Ltda.** Ingeniería LERP Ltda. [En línea] Ingeniería LERP Ltda. [Citado el: 23 de enero del 2010] http://www.lerp.cl/reloj%20control%20asistencia%20tradicional%20con%20tarjetas%20de%20carton.htm
6. **Kimaldi.** Kimaldi. [En línea] Kimaldi. [Citado el: 23 de enero del 2010] http://www.kimaldi.com/productos/control_de_acceso/control_de_acceso_on_line/terminal_kimaldi_biomax2.
7. **codigodebarra.com.** codigodebarra.com. [En línea] codigodebarra.com. [Citado el: 23 de enero del 2010] http://www.codigodebarras.com/tema.php?ID=para_que_sirve
8. **Frontline.** Frontline. [En línea] Frontline. [Citado el: 23 de enero del 2010] http://www.frontlineco.net/?p=126.
9. **Azerty.** Azerty. [En línea] Azerty. [Citado el: 23 de enero del 2010] http://www.area.com.mx/azerty/lectorestec.html .
10. **Meneses Abad, Abel; Marsi Peñalver Romero, Gladys**. *SXP, Metodología ágil para proyectos de software libre*. La Habana, 2009.

11. **Leyva Samada, Lisandra Isabel**. *Flujo de investigación para la metodología ágil*. La Habana, 2009.
12. **degerencia.com.** *degerencia.com.* [En línea] degerencia.com. [Citado el: 27 de enero del 2010] http://www.degerencia.com/articulo/business_process_management_bpm_articulando_estrategia_procesos_y_tecnologia.
13. **Jacobson, Ivar; Booch, Grady; Rumbauch, James.** *El Proceso Unificado de Desarrollo de Software*. La Habana: Felix Varela, 2004.
14. **Leyva Samada, Lisandra Isabel**. *Flujo de investigación para la metodología ágil*. La Habana, 2009.
15. **Larman, Craig.** *UML y Patrones.* La Habana: Felix Varela, 2004.
16. **Canós, José; Letelier, Patricio y Penadés, María Carmen.** Libro PDF. *Libro PDF.* [En línea] [Citado el: 25 de enero del 2010] http://www.willydev.net/descargas/prev/TodoAgil.pdf.
17. **DesarrolloWeb.** Qué es PHP. [En línea] [Citado el: 10 de febrero del 2010] http://www.desarrolloweb.com/articulos/392.php
18. **desarrolloweb.com.** *desarrolloweb.com.* [En línea] desarrolloweb.com. [Citado el: 2 de febrero del 2010] http://www.desarrolloweb.com/articulos/1112.php
19. **microsoft**. *microsoft.* [En línea] Microsoft. [Citado el: 10 de febrero del 2010] http://www.microsoft.com/spain/windowsserver2003/technologies/webapp/iis.mspx
20. **codigodebarra.com.** codigodebarra.com. [En línea] codigodebarra.com. [Citado el: 23 de enero del 2010] http://www.codigodebarras.com/tema.php?ID=para_que_sirve
21. **Cibernetica.** *Cibernetica*. [En línea] Cibernetica. [Citado el: 27 de enero del 2010.] http://www.cibernetia.com/manuales/instalacion_servidor_web/1_conceptos_basicos.php
22. **James Garret, Jesse.** Maestros del web. *Maestros del web*. [En línea]. [Citado el: 2 de febrero del 2010] http://www.maestrosdelweb.com/editorial/ajax/
23. La Oficina del W3C en España. *Guía Breve de Servicios Web.* [En línea]

Glosario de términos

Administrador: Es la persona que tiene privilegios para determinadas funcionalidades del sistema.

AJAX: **A**synchronous **J**avaScript **A**nd **X**ML.

Arquitectura Cliente/Servidor: Es un modelo para el desarrollo de sistemas de información, en el que las transacciones se dividen en elementos independientes que cooperan entre sí para intercambiar información, servicios o recursos.

Artefactos: Elementos de entrada y salida de las actividades. Productos tangibles del proyecto.

APACHE: Es un servidor HTTP de código abierto para plataformas Unix (BSD, GNU/Linux, etcétera), Windows y otras, que implementa el protocolo HTTP/1.1.

BPM: Business Process Management. Metodología empresarial destinada a mejorar la eficiencia de los procesos de negocio.

Control de acceso: Limita los movimientos del personal a las diferentes áreas de la entidad.

Control de presencia: Controla la presencia o el absentismo de los empleados.

CSS: Hojas de Estilo en Cascada (Cascading Style Sheets), es un mecanismo simple que describe cómo se va a mostrar un documento en la pantalla.

DOM: Document Object Model (Modelo de Objetos de documentos).

FrameWork: Estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

GIMP: GNU Image Manipulation Program (programa para manipular imágenes).

Herramientas CASE: (**C**omputer **A**ided **S**oftware **E**ngineering): Herramientas CASE: Herramientas utilizadas para el desarrollo de proyectos de Ingeniería de Software.

HTML: HyperText Markup Language. Lenguaje usado para escribir documentos para servidores World Wide Web.

HTTP: HyperText Transfer Protocol. Protocolo de Transferencia de Hipertextos. Modo de comunicación para solicitar páginas Web.

HU: Historias de usuario.

Interfaz de usuario: Interfaz a través de la cual un usuario interactúa con un sistema.

Linux: Es el nombre de un núcleo, pero se suele denominar con este nombre a un sistema operativo de libre distribución software libre (y de código abierto)

Microsoft: Compañía que manufactura los sistemas de operación DOS y Windows.

MVC: Modelo Vista Controlador.

PHP: PHP: Hypertext Preprocessor. Es un ambiente script del lado del servidor que permite crear y ejecutar aplicaciones Web dinámicas e interactivas.

Plugin: Es una aplicación informática que interactúa con otra aplicación para aportarle una función o utilidad específica, generalmente muy específica.

PostgreSQL: Es un Sistema de Gestión de Bases de Datos Objeto-Relacionales (ORDBMS) libre.

SXP: Metodología para el desarrollo de Software.

Software: Programas de sistema, utilerías o aplicaciones expresados en un lenguaje de máquina.

SGBD: Sistema de Gestión de Bases de Datos. Es el software que permite la utilización y/o la actualización de los datos almacenados en una (o varias) base(s) de datos por uno o varios usuarios desde diferentes puntos de vista y a la vez.

TI: Tareas de ingeniería.

UCI: Universidad de las Ciencias Informáticas.

WEB (WWW): Red de documentos HTML intercomunicados y distribuidos entre servidores del mundo entero.

XML: Extensible Markup Language. Es un lenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium. Orientado principalmente al almacenamiento, procesamiento y transmisión de mensajes.