

Instituto Tecnológico de Costa Rica

Carrera de Ingeniería en Computación

Campus Tecnológico Local

San Carlos

“Pernix Central”

Práctica Profesional para optar por el título de  
ingeniero en computación con grado académico de  
Bachillerato

Erwin Salas Mejías

## Campus Tecnológico Local, San Carlos

# Índice de contenido

Resumen	3
Abstract	4
Capítulo 1	4
1.1 Antecedentes:	5
1.2. Descripción de la Empresa	5
1.3 Problema:	6
1.4 Objetivos:	6
1.4.1 Objetivo general:	6
1.4.2 Objetivos específicos:	6
1.5 Justificación:	7
Capítulo 2	7
2.1 Marco teórico	8
2.1.1 Arquitectura cliente-servidor	8
2.1.2 Reglas de negocio	9
2.1.2 Tecnologías de desarrollo web	9
2.2 Trabajos relacionados	10
Capítulo 3	12
Solución planteada	12
3.1. Propuesta	12
3.2. Patrocinadores	13
3.3. Metodología	13
3.5. Análisis de los Riesgos:	15
Capítulo 4	16
Definición de requerimientos	16
4.1. Tareas realizadas para definir los requerimientos	16
4.2 Resultados obtenidos en la definición de los requerimientos	18
4.2.1 Historias de usuario	18
4.2.2 Requerimientos de interfaz:	18

Capítulo 5	19
Diseño de la plataforma de software	19
5.1. Tareas realizadas para definir el diseño	19
5.2 Resultados obtenidos en el diseño de la plataforma de software	19
Capítulo 6	29
Desarrollo de la plataforma de software	29
6.1. Tareas realizadas para el desarrollo de la plataforma de software.	30
6.2. Resultados obtenidos en el desarrollo de la plataforma de software.	31
Capítulo 7	33
7.1 Tareas realizadas para la evaluación de la plataforma de software.	33
7.2 Resultados obtenidos de la evaluación de la plataforma de software.	34
Capítulo 8	38
Conclusiones	38
Bibliografía	40

## Índice de figuras

Figura 5.1 Diagrama de arquitectura	20
Figura 5.3 Ejemplo de servicio	23
Figura 5.5 Barra de navegación	24
Figura 5.6 Tablas de registros	25
Figura 5.7 Vistas de componente tabla	26
Figura 5.9 Vistas con componentes lista	27
Figura 5.9 Formularios	27
Figura 5.10 Perfil de usuario	28
Figura 5.11 Tablero de posiciones	29
Figura 7.1 Resultados de pruebas	34
Figura 7.2 Ejemplo de test con rspec.	35
Figura 7.3 Ejemplo de test con rspec.	36
Figura 7.4 Ejemplo de test con rspec 2	37

# Resumen

El presente documento consiste en un informe detallado sobre el trabajo de graduación realizado para optar por el título de bachiller en computación, el propósito del proyecto realizado es el de contribuir con el manejo de la información de la empresa Pernix mediante una plataforma de software. Con la meta de hacer más fácil la administración de los proyectos de la empresa, así como sus clientes y equipos asociados, actividades realizadas y listado de empleados etiquetados con sus habilidades técnicas y blandas y otras funciones que se explican detalladamente a lo largo del documento. Durante el proceso de desarrollo del proyecto se abordaron varias etapas como recolección de requerimientos, diseño, desarrollo y evaluación de la plataforma utilizando metodologías como historias de usuario, diagramas de flujo, Maquetado de interfaces y scrum para el desarrollo dividido en varias etapas. Como resultado final se obtuvo un producto de calidad acorde con los requerimientos establecidos en las reuniones con los involucrados.

# Abstract

This document consists of a detailed report on the graduation work carried out to opt for the bachelor's degree in computing, the purpose of the project is to contribute to the management of the information of the Pernix company through a software platform.

With the goal of making easier the administration of the company's projects as well as its clients and associated teams, activities carried out and list of employees labeled with their technical and soft skills and other functions that are explained in detail throughout the document.

During the project development process several stages were addressed such as requirements collection, design, development and evaluation of the platform using methodologies such as user

histories, flow diagrams, interface and scrummating for the development divided into several stages.

As a final result, a quality product was obtained according to the requirements established in the meetings with those involved.

# Capítulo 1

A continuación, se exponen los detalles

## 1.1 Antecedentes:

Las empresas que se dedican al desarrollo de software requieren realizar una eficiente gestión de los recursos con los que cuentan, en este ámbito específico del desarrollo de software es común que se maneje principalmente recurso humano. Para lograr realizar esto de una manera eficiente y eficaz las empresas grandes y medianas recurren a herramientas de software que les permiten agilizar su labor. Estos sistemas están diseñados para modelar y automatizar todos estos procesos de gestión de recursos, a la vez que posibilita la toma de decisiones dentro de la organización de una forma rápida y eficaz.

Hoy en día es habitual emplear el término ERP para referirnos a los programas de gestión que poseen las empresas para llevar a cabo sus procesos de negocio, esto incluye: Compras, Ventas, Almacén, Contabilidad, Recursos Humanos... son muy utilizados por grandes y medianas empresas. Algunas marcas populares que podemos mencionar son Navision, SAP R/3, Sage, Oracle, Ross, Axapta, IFS. Estos se comercializan por medio de licencias y software prefabricado con soporte para muchos procesos, no obstante, muchas veces con desventajas como bajo nivel de personalización y gran costo monetario por el precio de licencia. Aunque existen algunas opciones más accesibles como Odo.

Estos sistemas suelen estar divididos por módulos y submódulos de manera que el cliente pueda seleccionar cuáles de ellos requiere basado en los procesos de la empresa que se deseen cubrir.

## 1.2. Descripción de la Empresa

Pernix SA es una empresa de capital costarricense certificada por la marca país esencial Costa Rica y fundada en el 2009. Dedicada al desarrollo de software a la medida, soporte de aplicaciones, servicios de “outsourcing” y de más servicios, para clientes tanto nacionales e internacionales, dentro de las que pueden mencionar Flatirons, StarProg, LiveWatch security entre otros.

Representados bajo el slogan “Software crafters” o artesanos de software, Pernix siempre se ha enfocado en brindar calidad e innovación en sus producciones, acompañadas de un diseño de experiencias de usuario que convierten la experiencia de uso en algo comfortable.

## 1.3 Problema:

Actualmente el registro de los datos sobre empleados, proyectos, clientes y demás, son manejados utilizando sistemas de almacenamiento en la nube y gestores de archivos como google drive. Debido al crecimiento en la empresa esta información se ha visto cada vez más difícil de manejar, los archivos xls cada vez se llenan más de filas y columnas incrementando el tamaño de los registros. Esta situación dificulta la navegación y el manejo de la información, por lo que es común en los usuarios perderse entre toda la pila de documentos y archivos. Como consecuencia de lo mencionado anteriormente se complican las tareas de administración a la hora de llevar control de las actividades, empleados, proyectos, clientes entre otros...

Si se analiza este problema a futuro el tiempo que se gasta o se pierde utilizando la metodología de gestión actual representa pequeñas pérdidas que se van acumulando con los años. En

resumen, se necesita una gestión más efectiva de la información y abrir puertas a nuevas posibilidades y mejoras a realizar.

## 1.4 Objetivos:

### 1.4.1 Objetivo general:

Contribuir con la administración de Pernix SA mediante la creación de una plataforma centralizada de software para manejo de la información de empleados, clientes, proyectos y actividades.

### 1.4.2 Objetivos específicos:

1. Definir los requerimientos funcionales de la plataforma
2. Diseñar una plataforma de software para el manejo de la información de empleados, clientes, proyectos y actividades.
3. Desarrollar una plataforma de software para el manejo de la información de empleados, clientes, proyectos y actividades.
4. Evaluar una plataforma de software para el manejo de la información de empleados, clientes, proyectos y actividades.

## 1.5 Justificación:

El proyecto ***Pernix Central*** viene a impactar de forma positiva la operatividad de la empresa Pernix, facilitando una gestión ágil y rápida de la información permitiendo ahorrar tiempo y dinero, agregando ventajas competitivas que favorecen el desempeño en general de las tareas administrativas y operativas mediante las funcionalidades como el registro de clientes de la empresa, el estatus de proyectos, habilidades del personal y algunas otras características las cuales se describen a fondo en el capítulo 4.

La aplicación contribuirá en gran parte a la ejecución, control y registro de las operaciones diarias de Pernix, facilitando el día a día del personal que administra el recurso humano. La implementación de una plataforma centralizada mejora en la experiencia de usuario a la hora de acceder a la información y gestionar los datos con una mejor trazabilidad. También se pretende impulsar la participación del personal en las actividades a nivel de la empresa. Esto mediante el registro de las mismas, así como cada participación de los colaboradores. De esta forma los colaboradores suman puntos y se incentivan a liderar el tablero de posiciones. Se espera que de esta forma el sistema impulse un mejor desempeño de los colaboradores ya que muchas de las actividades involucran capacitaciones o talleres que benefician a los profesionales por ello se busca lograr esa motivación a involucrarse en las diferentes actividades organizadas por la empresa.

# Capítulo 2

## Revisión de literatura

### 2.1 Marco teórico

En este capítulo se abordarán los conceptos teóricos de desarrollo de software, reglas de negocio, herramientas a utilizar y de más, necesarios para una mejor comprensión del proyecto.

#### 2.1.1 Arquitectura cliente-servidor

Este tipo de arquitectura es caracterizada por hacer una división marcada de responsabilidades, es decir cada componente cumple con una serie de funciones que definen su rol. Por un lado, tenemos al cliente, a nivel de capas este se encuentra en el punto más alto, algunos le llaman la capa de presentación. En esta capa se encuentra la parte frontal del sistema, es decir la interfaz de



usuario mediante la cual este puede interactuar con el sistema, un ejemplo de este podría ser un dispositivo móvil o navegador web. Dentro de las responsabilidades que posee esta capa están:

1. Mostrar información al usuario sobre lo que está sucediendo en el sistema.
1. Permitir la fácil interacción entre el usuario y el sistema, mediante controles ya sean botones o campos de texto y de más
1. Mostrar resultados, mensajes y notificaciones al usuario final.
1. Realizar peticiones de datos al servidor

El servidor, por otro lado, se encarga de gestionar la capa de datos que corresponde a la base de datos e interfaces de conexión creadas para comunicarse con el cliente. Esta capa se rige principalmente por las reglas de negocio, las cuales determinan cómo están relacionados los datos y de qué forma se envían a la capa de presentación. Dentro de las responsabilidades que posee esta capa están:

1. Control del flujo de datos
1. Conexión con el motor de base de datos
1. Conversión de los datos a un formato estándar (JSON, XML)
1. Responder a las peticiones de datos realizadas por el cliente

La razón por la cual se escogió esta arquitectura para el proyecto, radica en que el sistema será utilizado por varias personas a la vez. Esto permite explotar las ventajas del patrón arquitectónico, ya que este permite que múltiples clientes se conecten a un mismo servidor. Por ende, todos compartirán una misma base de datos y se pueden distribuir mejor las cargas de procesamiento en incluso realizar operaciones de forma paralela.

### 2.1.2 Reglas de negocio

Una regla de negocio es una condición que se debe satisfacer cuando se realiza una actividad de negocio. Una regla puede imponer una política de negocio, tomar una decisión o inferir nuevos datos de datos existentes. Estas reglas determinan el comportamiento del sistema, así como los requerimientos que este debe cumplir.

El primer paso que se debe hacer siempre que se va a desarrollar algún sistema, es entender sus reglas y lógica, teniendo esto bien claro se puede aplicar la ingeniería para poder dar con una solución certera al problema abordado.

En este caso particular, las reglas de negocio se basan en acciones como la gestión de proyectos, empleados, equipos de trabajo, y demás funcionalidades que se abordan a detalle en el capítulo 4.

### 2.1.2 Tecnologías de desarrollo web

**Redux:** Como los requisitos para las aplicaciones JavaScript se han vuelto cada vez más complicados, nuestro código debe administrar más estados que nunca. Este estado puede incluir respuestas del servidor y datos almacenados en caché, así como datos creados localmente que aún no se han conservado en el servidor.

Estado UI también está aumentando en complejidad, ya que necesitamos administrar rutas activas, pestañas seleccionadas, giradores, controles de paginación, etc. Redux viene a complementar y a solucionar este tipo de situaciones en las que el estado de nuestras aplicaciones se vuelve difícil de administrar. Popularmente se suele combinar esta herramienta como librerías como react, debido a que como tal es solo una librería y no un framework, por lo que necesita de otros complementos para ganar robustez.

**Saga:** redux-saga es una biblioteca que tiene como objetivo hacer que los efectos secundarios de la aplicación (es decir, cosas asíncronas como la obtención de datos y cosas impuras como acceder a la memoria caché del navegador) sean más fáciles de administrar, más eficientes de ejecutar, fáciles de probar y mejor para manejar los fallos.

Básicamente una saga es como un hilo separado en tu aplicación que es el único responsable de los efectos secundarios. redux-saga es un middleware redux, lo que significa que este hilo se puede iniciar, pausar y cancelar desde la aplicación principal con acciones de redux normales, tiene acceso al estado de aplicación de redux completo y también puede enviar acciones de redux.

**Active Record:** Es un componente desarrollado en ruby que al integrarse con el framework rails, puede ser utilizado para interactuar directamente con la base de datos. A este concepto se le conoce popularmente como Mapeo Objeto-Relacional o como se conocen comúnmente, ORM

(del inglés Object Relational Mapping). Un ORM permite convertir los datos de tus objetos en un formato correcto para poder guardar la información en una base de datos (mapeo) creándose una base de datos virtual donde los datos que se encuentran en nuestra aplicación, quedan vinculados a la base de datos (persistencia). De esta forma se logra desacoplar el motor de base de datos, es decir podemos declarar el esquema de la base en este caso utilizando Ruby. Evitando así tener crear la base de datos con otro lenguaje como SQL o Pgsql. Lo anterior trae ventajas con como por ejemplo una mayor portabilidad, ya que se crean migraciones que generan las tablas a partir de los modelos en Ruby. Permitiendo montar la base de datos de forma fácil y rápida independientemente del motor utilizado.

## 2.2 Trabajos relacionados

El mundo empresarial actual, afectado por la recesión económica mundial, problemas fiscales actuales y demás situaciones, desafía a la administración y a los propietarios de empresas con difíciles decisiones estratégicas. ¿Cómo responder o incluso prever cambios en el entorno empresarial?

Muchas veces sin siquiera llevar registros formales de los datos ni tener herramientas. Los procesos de gestión se tornan complicados para la mayoría. Como soluciones para mitigar dichos problemas, las empresas usualmente se apoyan en la tecnología, principalmente en sistemas de información que les ayuden en sus labores cotidianas. Los procesos corporales están sujetos a turbulencias y caos que tienden a ser más llevaderos mediante herramientas tecnológicas.

Los autores Wexing Song y Dengdao Li se expresan sobre los recursos de información como una ventaja competitiva y conductor dominante para llevar una empresa a ser exitosa [6].

El cambio en el paradigma de gestión representa grandes oportunidades competitivas para el ecosistema empresarial. Cabe destacar que no existe un software 100% estándar con el cual las empresas puedan abordar todos sus procesos administrativos a la medida. Sin embargo, hay sistemas pre fabricados que cuentan con funcionalidades y módulos bastante completos que se pueden adaptar a los negocios y distintas empresas. Por otro lado, existen aplicaciones

empresariales de integración que se definen como la implementación de varios modelos de software que se integran en aplicaciones empresariales existentes [4]. Esta metodología permite conectar varias aplicaciones y centralizarse en una sola aplicación, lo cual es muy útil en empresas que poseen muchos sistemas separados que presenten características comunes, que contribuyan a una correcta integración con servicios y aplicaciones externas.

El objetivo básico de sistemas para la planificación de recursos empresariales es el de mejorar el mercado utilizando la tecnología de la información más amplia, acelerando la actualización de productos empresariales, la competitividad. Que apoyados por ordenadores y tecnología de fabricación se logra la automatización del proceso de producción, en casos de manufacturación, por ejemplo.

Existen algunos puntos claves sobre el manejo de la información empresarial: el primero es el automatización de procesos de producción y diseño de productos; La segunda es la transmisión de información, los departamentos. trabajar juntos y automatización suministro, sistemas; producción, incluyendo personal, circulación o servicios y red de información empresarial; y por último tercero es la gestión de la información, financiero, material, y otra información de gestión de todo el proceso. En el fondo de la planificación de recursos empresariales, las empresas mejoran la competitividad. Por lo tanto, cada vez hay más empresas que pagan por atención a la información, y una actualización estratégica de altura, con gran cantidad de recursos financieros, humanos y recursos materiales. [7] Por eso es importante y significativo para el desarrollo de las empresas manejar de forma eficaz la información empresarial.

A pesar de que en este proyecto no se planea realizar un ERP completo como tal, el mismo si cumple con ciertas características de estos sistemas o ataca algunos de los problemas que un ERP soluciona.

# Capítulo 3

## Solución planteada

### 3.1. Propuesta

Se propone desarrollar una plataforma web modular que posea facilidades que permitan un acceso más directo y rápido hacia la información. La plataforma será desarrollada mediante módulos los cuales tendrán un conjunto de funcionalidades asociadas a una o más entidades de datos.

La solución consta de los siguientes módulos:

1. Empleados:
2. Proyectos
3. Equipos de desarrollo
4. Habilidades (“Skills”)
5. Clientes
6. Actividades

El software estará compuesto por 2 componentes macro Backend y Frontend siguiendo el patrón arquitectónico cliente servidor en donde el cliente será el frontend o capa de presentación, y el backend será el servidor el cual aloja la capa de datos y los puntos de anclaje mediante los cuales el cliente podrá absorber los datos que requiera mostrar en la interfaz:

**Backend:** Se encarga del almacenamiento y gestión de los datos. Como gestor de bases de datos se utilizará Postgres. El Back-end será desarrollado utilizando Ruby on Rails.

**Frontend:** Tiene la función de realizar la comunicación entre los usuarios finales y el Backend., será desarrollado utilizando React js.

## 3.2. Patrocinadores

**Tabla 3.1** Patrocinadores del proyecto

Nombre	Rol en el proyecto	Nombre del puesto	Descripción de puesto	Objetivo
Cristina Hernandez	Cliente	Recursos humanos	Se encarga de la gestión del recurso humano	Definir las necesidades a solventar.
Johnny Xu	Líder Técnico	Ingeniero de software	Se encarga de desarrollar y liderar proyectos	Liderar el proyecto y definir el alcance.
Emmanuel Murillo	Mentor	Ingeniero de software	Se encarga de desarrollar software	Guiar al practicante en cuanto a consultas y su estadía en la empresa

## 3.3. Metodología

**Tabla 3.2** Metodología propuesta para alcanzar los objetivos.

**General:** Contribuir con el manejo de recursos y actividades de Pernix SA mediante la creación de una plataforma centralizada de software para la gestión de la empresa.

Objetivo	Tarea	Meta	Indicador
Definir los requerimientos funcionales	<ol style="list-style-type: none"> <li>1. Entrevistar a los involucrados para lograr obtener la información requerida.</li> <li>2. Analizar los datos obtenidos a partir de las entrevistas y modelarlos en historias de usuario.</li> </ol>	<ol style="list-style-type: none"> <li>1. Obtener un reporte con las historias de usuario y requerimientos claros y concisos.</li> <li>2. Tener un contexto amplio sobre la lógica de negocio del proyecto.</li> <li>3. Priorizar y agrupar los requerimientos en iteraciones</li> </ol>	<ol style="list-style-type: none"> <li>1. Entrevistar a al menos 3 de los patrocinadores.</li> <li>2. Definir al menos 30 requerimientos funcionales</li> </ol>
Diseñar la plataforma de software centralizada.	<ol style="list-style-type: none"> <li>1. Diseñar mockups para la interfaz gráfica de usuario.</li> <li>2. Realizar un diagrama de todas las entidades que compondrán la capa de datos.</li> <li>3. Realizar diagramas de flujo de la aplicación</li> <li>4. Definir los recursos o</li> </ol>	<ol style="list-style-type: none"> <li>1. Obtener un diseño que sirva de guía para la fase de desarrollo del proyecto y de una perspectiva clara de lo que se quiere obtener.</li> <li>2. Obtener un esquema de módulos en los cuales se dividirá la aplicación.</li> </ol>	<ol style="list-style-type: none"> <li>1. El cliente debe validar el diseño aprobando cada sección con una aceptación de 10 - 9</li> </ol>

	endpoints necesarios así como sus parámetros y datos .		
Desarrollar la plataforma de software centralizada.	<ol style="list-style-type: none"> <li>1. Desarrollar el backend en Ruby on Rails bajo una arquitectura de servicios REST e implementando un ORM para realizar migraciones y mapeos de la base de datos.</li> <li>2. Desarrollar un cliente web en React.js utilizando programación reactiva e interfaces dinámicas y flexibles.</li> </ol>	Obtener una plataforma funcional y lista para testing que cumpla con los requerimientos funcionales definidos en la etapa 1 y plasmados en el diseño de la etapa 2.	<ol style="list-style-type: none"> <li>1. Se deben aprobar todas las funcionalidades según los criterios de aceptación definidos al inicio.</li> </ol>
Evaluar la plataforma de software centralizada.	<ol style="list-style-type: none"> <li>1. Diseñar pruebas unitarias para front-end con la herramienta jest para react y rspec para el backend en rubí.</li> <li>2. Testear los componentes del frontend mediante la herramienta storybooks</li> </ol>	<ol style="list-style-type: none"> <li>1. Depurar todos los errores y bugs que se encuentren mediante la ejecución de los casos de pruebas.</li> </ol>	<ol style="list-style-type: none"> <li>1. Realizar al menos 60 casos de prueba de forma variada entre pruebas unitarias y</li> </ol>

### 3.5. Análisis de los Riesgos:

**Tabla 3.3** Cuadro de riesgos



Nombre o descripción	Categoría	Causa
Perdida de código	Tecnológico	Causas externas, fallos en los repositorios, código no respaldado
Errores en las integraciones	Tecnológico	Mala comunicación entre los compañeros de trabajo
Deserción de patrocinadores por renuncia, viaje de trabajo, vacaciones...	Personas	Puede causar retrasos equivalentes al nivel de importancia del patrocinador
Errores en el manejo de versiones	Tecnológico	Al actualizar repositorios o subir cambios pueden ocasionar conflictos o incluso pérdida de cambios lo cual atrasaría el proyecto
Robo de equipos	Externo	Podría presentarse un escenario en el que al practicante le hurten su máquina de trabajo y esta contenga avances importantes sin respaldar.

# Capítulo 4

## Definición de requerimientos

En esta sección se describirán las funcionalidades requeridas en el sistema para el cumplimiento del alcance establecido. Para la obtención de los requerimientos y el contexto detallado del problema, se plantearon una serie de actividades en las cuales se conversó y se definió en qué consistirá el proyecto y cuál será su alcance.

### 4.1. Tareas realizadas para definir los requerimientos

Para la definición de los requerimientos se realizaron 4 reuniones de las cuales en 3 participaron los principales stakeholders, en estas reuniones se habló meramente sobre la lógica de negocio en la cual se centraba el sistema. Cristina Hernández, de recursos humanos es quien conoce mejor el contexto ya que ella es quien realiza de forma manual muchas de las funcionalidades que se quieren mejorar con el sistema.

**Sesión 1(stakeholders):** Se habló sobre la gestión de los usuarios, se definió que estos iban a ser registrados por el usuario administrador. Una vez que se creaba su perfil el empleado nuevo, la persona registrada debe proceder a completar el resto de información mediante una opción de edición de perfil con el resto de información de empleado, así como su horario de trabajo y habilidades (técnicas y blandas). También se definió que en el sistema existirían diferentes roles de acceso al sistema, de acuerdo a una matriz de permisos que se definiría más adelante.

**Sesión 2 (stakeholders):** En esta ocasión se abordaron los temas correspondientes a la gestión de proyectos, se determinó que la aplicación debe permitir, identificar posibles miembros para nuevos proyectos. Esto basado en sus habilidades técnicas y blandas, cada proyecto debe tener una lista de skills requeridos y en base a eso sugerir a los empleados que cumplan con esos

requisitos. Esto con el fin de facilitar la reubicación de personal al liberarse un proyecto.

**Sesión 3 (stakeholders):** En esta sesión se discutieron funcionalidades relacionadas a la eliminación o desactivación de entidades como proyectos, usuarios, equipos de proyectos y clientes. Se determinó que los perfiles de usuario se desactivarán, pero que asu vez se mantendrían en la base de datos por una posible reactivación. Al igual que en los usuarios los proyectos también tendrán un status de activo/inactivo ya que así es como se están trabajando actualmente mediante hojas de Excel.

**Sesión 4 (Equipo de trabajo):** En esta reunión participó el equipo de trabajo conformado por los integrantes Saúl Zamora (líder técnico) Johnny Xu (PM), David Lobo (Desarrollador) y Erwin Salas (Desarrollador). Como parte de las labores realizadas ese día, se analizaron los requerimientos del proyecto, Se crearon los repositorios y equipos respectivos en **bitbucket**. Además de la creación del backlog en **trelllo** en el que se desglosaron las tareas correspondientes para cada historia de usuario.

## 4.2 Resultados obtenidos en la definición de los requerimientos




### 4.2.1 Historias de usuario

Uno de los puntos claves en el desarrollo de un proyecto de software, las historias de usuario son la base para definir el rumbo de un proyecto. Representan el producto que se quiere elaborar y sus requerimientos y métricas a cumplir. Cada una de estas historias puede descomponerse de una o más tareas a realizar en la parte programada.

Como resultado de las sesiones con los stakeholders se obtuvieron un total de 30 requerimientos funcionales los cuales se explican detalladamente en el anexo [historias de usuario](#), divididos en entregables adaptando el proyecto a la metodología ágil scrum.

### 4.2.2 Requerimientos de interfaz:

**Tabla 4.1:** Requerimientos de interfaz

Nombre: Colores de Pernix	
<b>Id:</b> RI-01	<b>Prioridad:</b> Baja
<b>Usuario:</b> Ninguno	<b>Dependencia:</b> Ninguna
<b>Descripción:</b> La aplicación debe usar colores que representen a la empresa en este caso son:  1. Naranja:  2. Azul:  3. Verde: 	
<b>Observaciones:</b> Ninguna	

# Capítulo 5

## Diseño de la plataforma de software

En este capítulo se abordará a detalle el diseño de la plataforma Pernix Central, dicho diseño se divide en varias partes esenciales que componen el software. En otras palabras, este diseño se convertirá en los planos o guía para el desarrollo del sistema.

## 5.1. Tareas realizadas para definir el diseño

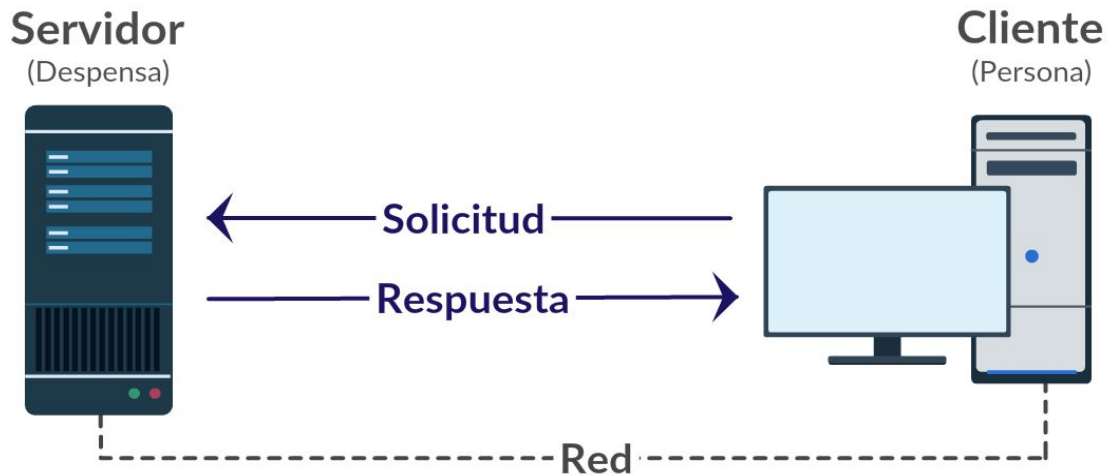
**Análisis de los requerimientos:** Para diseñar la aplicación solicitada por el cliente, se debe analizar cada uno de los requerimientos o historias de usuario definidas en el capítulo 4. Esto permite poder formar un contexto de lo que se quiere desarrollar, los datos que se deben modelar, así como las vistas que representarán cada requerimiento. Este paso permitirá poder generar un criterio más firme con respecto a las herramientas y tecnologías a usar.

**Investigación:** Para poder diseñar la plataforma se requiere conocer a fondo las herramientas y tecnologías más aptas y que se adecuen a los requerimientos que se deben solventar, conocer bien las fortalezas y debilidades de cada una.

**Consultar a los expertos:** Gracias al programa de aprendices de Pernix los miembros del programa tienen acceso a consultas técnicas y apoyo como parte de sus mentores. De manera que facilita obtener opiniones sólidas y recomendaciones sobre las tecnologías y herramientas a considerar.

## 5.2 Resultados obtenidos en el diseño de la plataforma de software

**Diagrama de arquitectura:** En este diagrama se representan los componentes macro que conforman el sistema en cuestión. Para este proyecto en específico se utilizó la arquitectura cliente servidor, patrón muy utilizado por gran cantidad de sistemas existentes r



**Figura 5.1 Diagrama de arquitectura**

**Cliente:** Comprende a todos los dispositivos que consumen los servicios de datos provenientes de la capa de servidor, este dispositivo puede ser desde una computadora de escritorio, Tablet o dispositivo móvil. Al ser desarrollado en JavaScript y un ambiente de desarrollo web el cliente podrá ser consumido mediante un navegador web que cuente con alguno de los motores de renderización populares, Gecko, Webkit, y Safari .

**Servidor :** En esta capa se encarga de gestionar el flujo de los datos, el almacenamiento consulta edición y eliminación. El mismo se encuentra conectado directamente con la base de datos y juega un papel de intermediario entre la interfaz de usuario y la base de datos. El servidor se encuentra comunicado mediante una conexión de red que permite al cliente acceder a él mediante peticiones HTTP.

**Componentes y servicios:** En esta sección se explicarán los componentes micro de la aplicación, estos van desde servicios web hasta gestores de estados y demás asociados a las tecnologías utilizadas en el proyecto.

El cliente web será desarrollado bajo el esquema SPA (Single Page Application). Este patrón consiste en abstraer la aplicación dividida por páginas, es decir que cada vista de la aplicación será representada por una página. De esta forma se puede garantizar una fácil navegación en la aplicación mediante el “ApplicationRouter” ver Figura 5.2.

**Componentes:** Como parte de las tendencias en desarrollo front-end, react integra un esquema de desarrollo basado en componentes. Esto nos permite poder dividir nuestra interfaz tanto como creamos conveniente. Este este esquema va muy acorde con el principio de responsabilidad única, en el que podemos crear componentes con una única función, fácil de probar y de cambiar.

**Redux:** Esta librería tiene un papel crucial como parte de los componentes del cliente en la arquitectura. Este es utilizado para gestionar los estados de los componentes mediante módulos, los cuales representan cada una de las entidades del sistema (tablas, modelos). Permitiendo estructurar una arquitectura consistente y fácil de acoplar con el esquema “restful “de los servicios web.

**Sagas:** Diseñadas para el manejo el manejo de peticiones acciones asíncronas, las sagas son útiles porque permiten la ejecución de funciones en bloques y la interacción directa con el estado de redux, por lo que es posible disparar acciones directamente desde la función saga como tal.

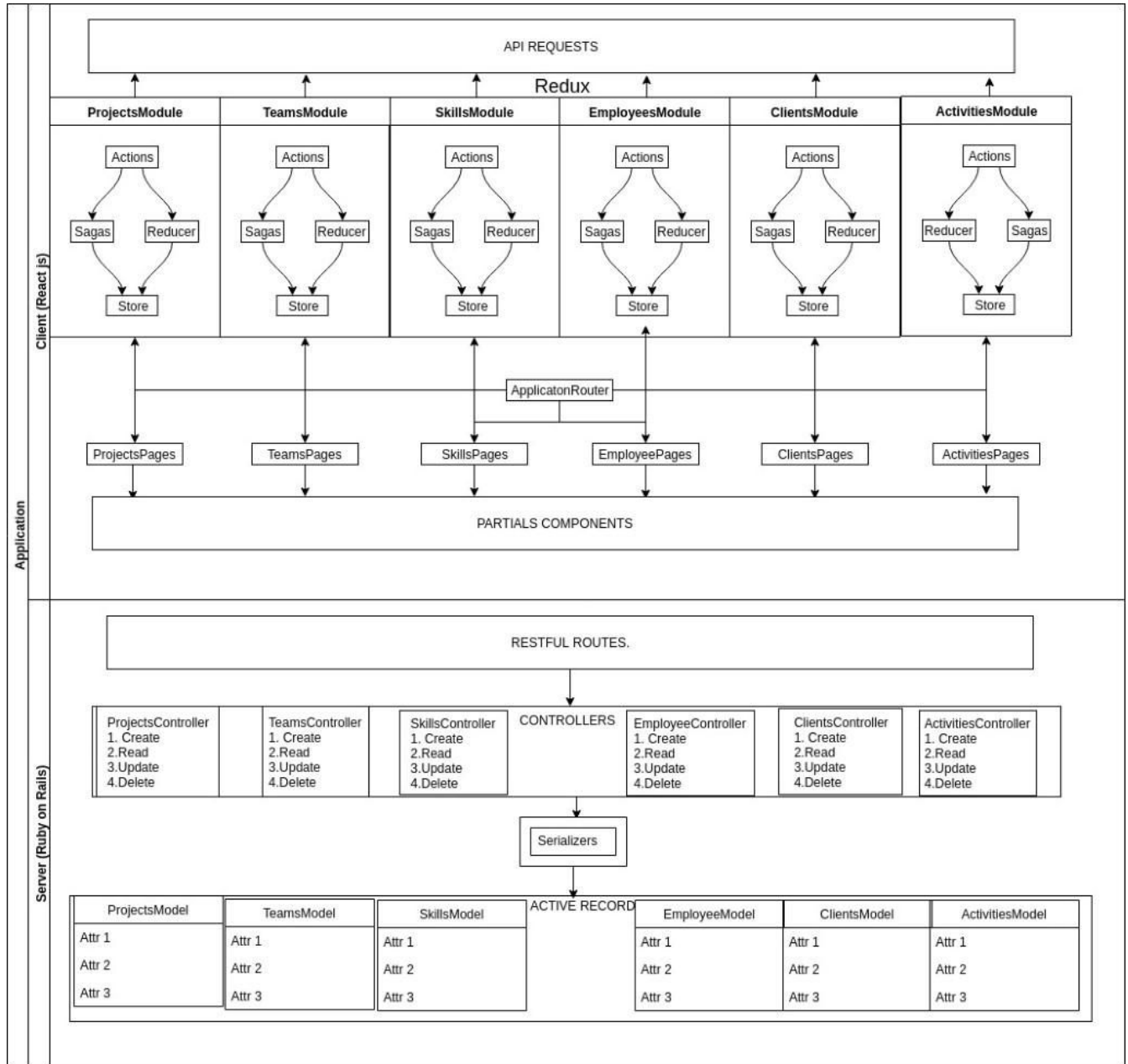


Figura 5.2 Componentes de la aplicación

**Servicios web:** Estos se encuentran diseñados bajo es estándar restful, lo cual permite un patrón de transferencia de la información, utilizando los estados Http para indicar la acción a efectuar sobre los datos. Estas acciones van desde crear nuevos registros de entidades hasta eliminar y actualizar los mismos.



A continuación, se explica la estructuración de los servicios web utilizados mediante un ejemplo con la entidad “skills”.

### GET Skills index

`/v1/skills`

Response:

```
[  
  {  
    "id": number,  
    "name": string,  
    "skill_type": string,  
    "level": string  
  },  
  ...  
]
```

### GET Skill show

`/v1/skills/:id`

Response:

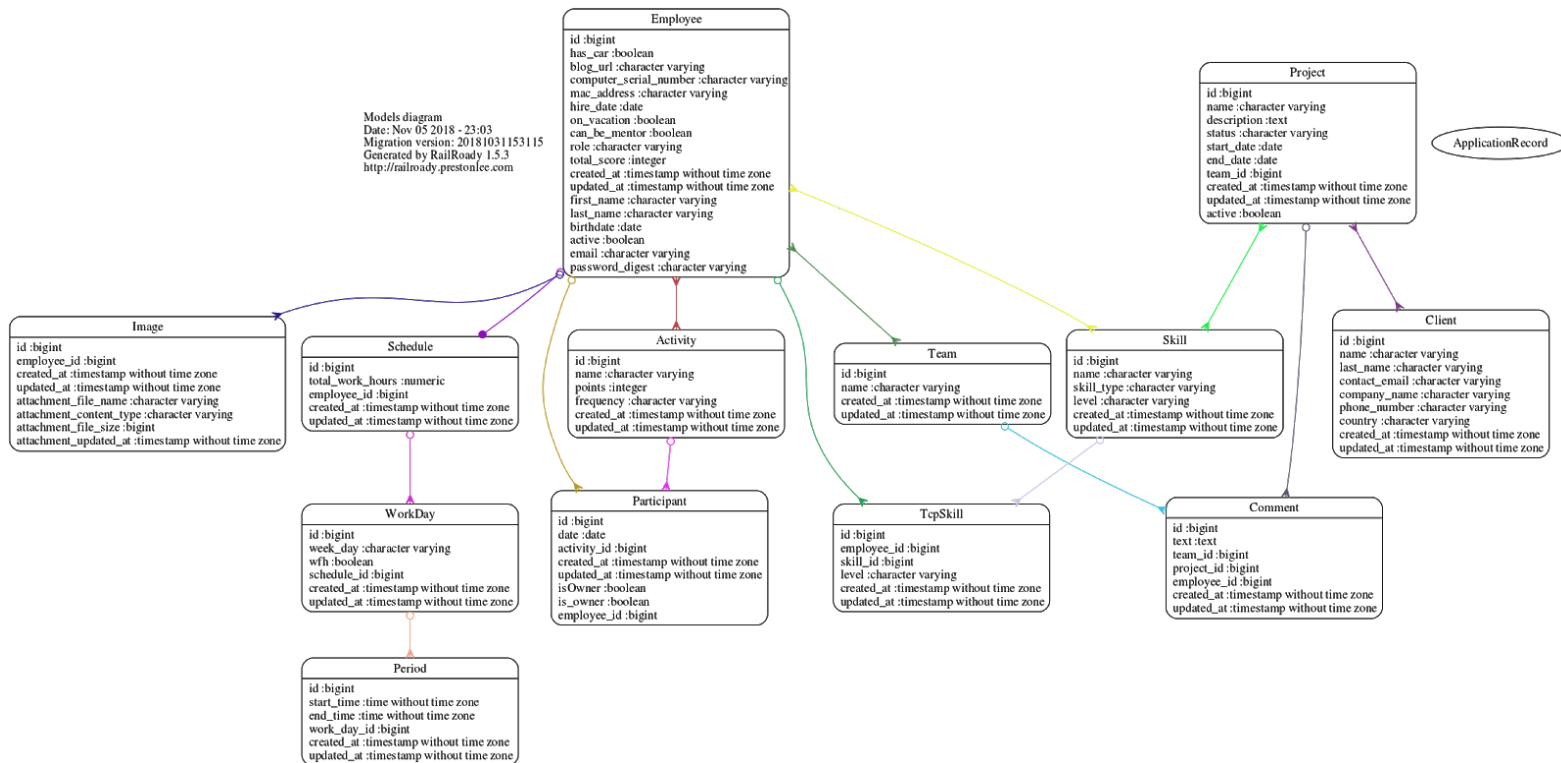
```
{  
  "id": number,  
  "name": string,  
  "skill_type": string,  
  "level": string  
}
```

*Figura 5.3 Ejemplo de servicio*

Los servicios utilizan los métodos http de GET, POST, PUT, DELETE para identificar a qué acción corresponde el servicio solicitado. En la imagen 3 se muestra un ejemplo de servicio, **OBTENER** representa la traducción al español del método http **GET**. En la figura 3 se muestra la ruta `/v1/skills` haciendo referencia al extremo que retorna el listado de skills existentes en la aplicación. La forma completa del url tiene como prefijo la dirección del host donde se aloja el servidor por ejemplo “localhost:3000” seguido del nombre de la entidad que queremos consultar. Si realizamos una petición a la misma ruta pero agregando un parámetro id en la ruta

`/v1/skills/{id}` obtenemos únicamente el elemento asociado a ese skill.

**Diagrama de clases:** Dicho diagrama representa los modelos de datos que serán representados mediante abstracciones u objetos. A nivel de programación se convierten en las estructuras de datos utilizadas a lo largo de todo el código fuente de la aplicación.



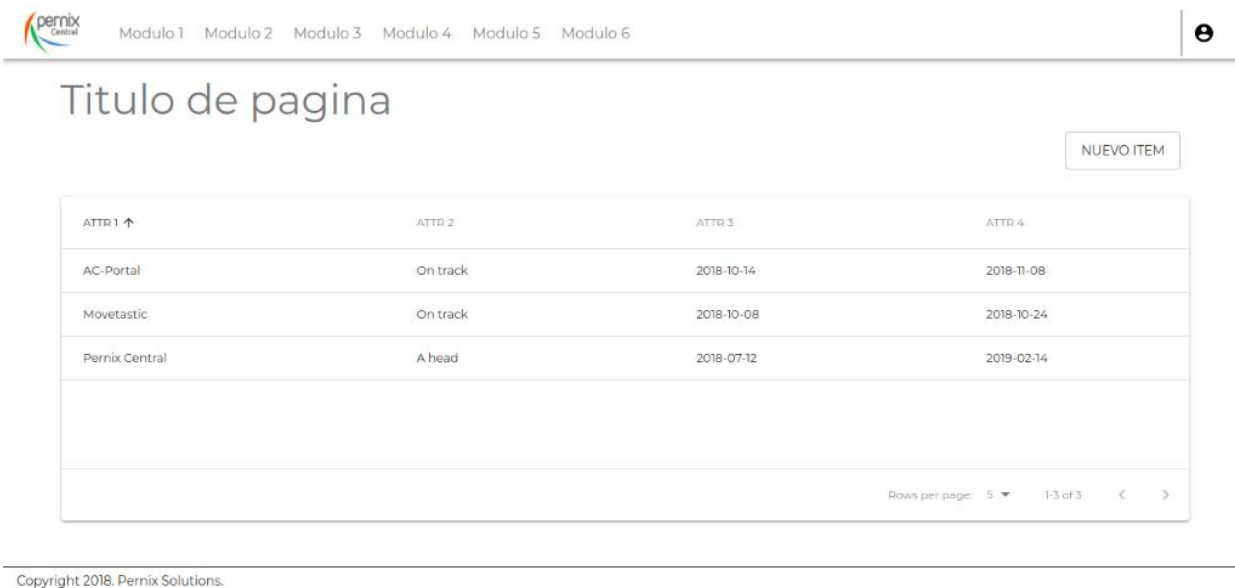
**Figura 5.4 Diagrama de clases**

**Interfaz de usuario:** El diseño de interfaz de usuario, se basa en “material design”, lo cual es una guía de estilo definida por google. Se puede decir que es como un tipo de patrón de interfaz. Para el diseño de la interfaz de usuario será diseñada bajo un esquema basado en componente, tomando en cuenta que la tendencia en tecnologías web apunta hacia ese paradigma, por lo cual es importante considerar estos aspectos desde la fase de diseño. Como el backend de la aplicación utiliza un estándar “restful” para estructurar sus servicios, es importante tomar en cuenta dicha convención a la hora de diseñar la interfaz de usuario.



**Figura 5.5 Barra de navegación**

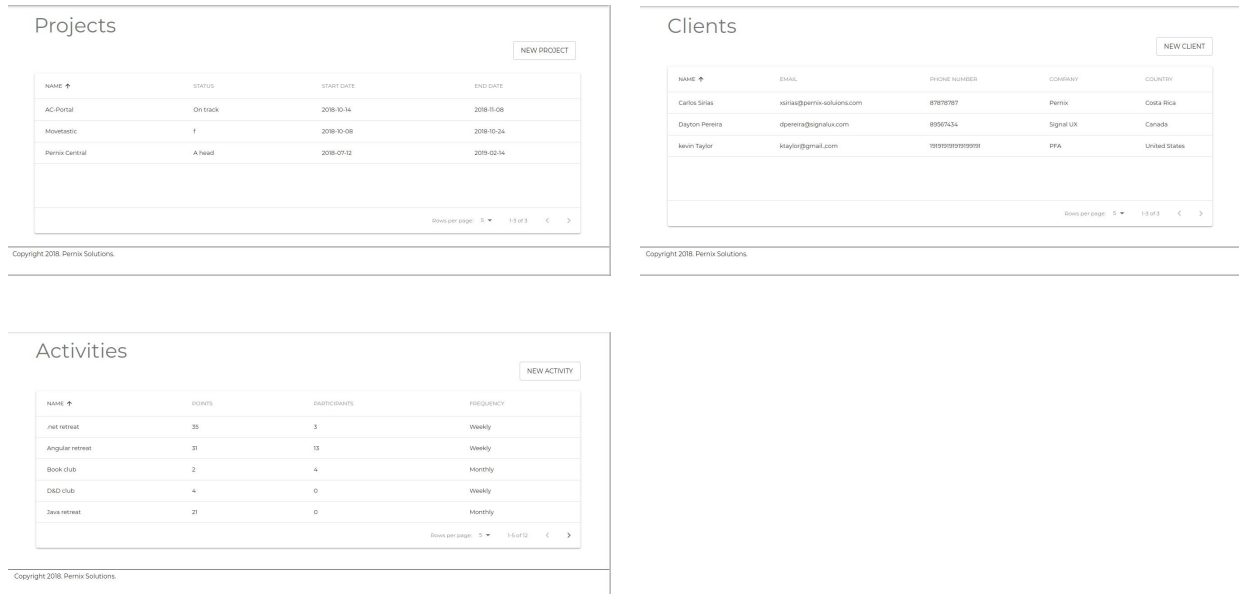
Uno de los aspectos más importantes en el diseño de una experiencia de usuario confortable es la navegación. A menudo se convierte en el componente de interfaz más utilizado por los usuarios. Se encarga de interconectar vistas de la aplicación e incluso manejar en envío y paso de parámetros entre ellas. Tal y como se muestra en la **Figura 5.4** el componente utilizado para este proyecto, consta de hipervínculos que dirigen al usuario a las vistas principales de cada módulo. para lograr esto a nivel de react cada página debe estar importada y asociada a una ruta específica en el “Application Router” ver **Figura 5.2**.



Copyright 2018. Pernix Solutions.

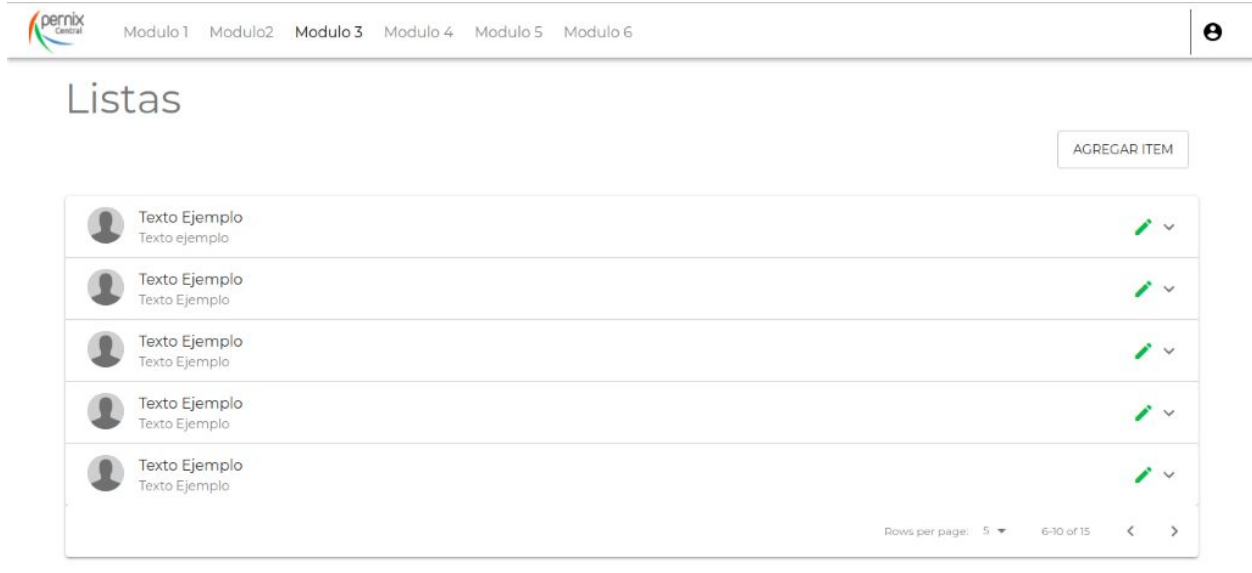
**Figura 5.6 Tablas de registros**

Componente genérico que mostrará listados de registros cargados directamente a través del servidor. En el “Título de página” se describe el nombre de la entidad que se está cargando en pantalla. El componente debe tener opciones de paginación y para cambiar cantidad de registros por página.



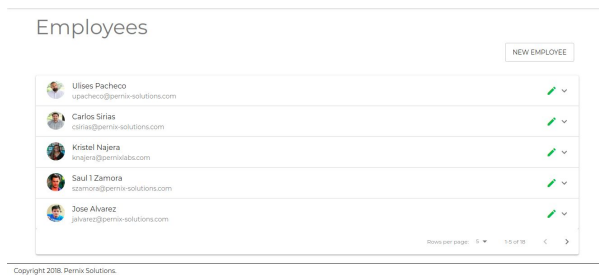
**Figura 5.7** *Vistas de componente tabla*

En la **Figura 5.7** se muestran maquetas de lo que serán las vistas que utilizarán el componente de tabla para mostrar sus registros. Este componente se comporta de igual forma para cualquiera de estas 3 entidades. En la esquina inferior derecha de la **Figura 5.6**, se pueden observar los botones de paginado y de selección de cantidad de registros a mostrar.



**Figura 5.8 Listas de registros**

Al igual que el componente anterior este tiene la funcionalidad de mostrar listados de registros, con la diferencia de que en este componente podemos mostrar avatar o iconos que represente a la entidad en cuestión, por ejemplo al cargar la entidad empleado se muestra un avatar con la imagen de cada registro de empleado.



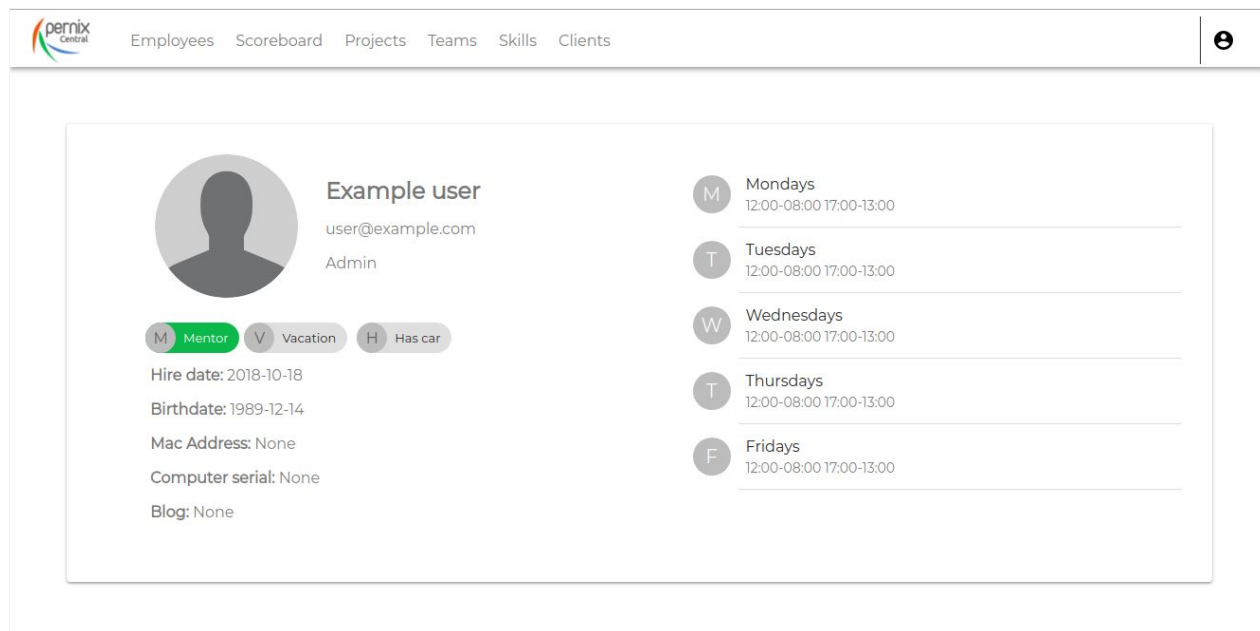


**Figura 5.9** *Vistas con componentes lista*

Los componentes y vistas mostradas anteriormente tienen la funcionalidad de mostrar los registros de datos. Logrando una experiencia de usuario agradable y fácil de manejar.

**Figura 5.9** *Formularios*

En la **Figura 5.8** se muestra un ejemplo de formulario, estos se utilizan tanto para editar como para crear nuevos registros. Este componente será utilizado por todas las vistas de crear y editar de la aplicación, con la variante de campos asociados a cada una.



**Figura 5.10 Perfil de usuario**

Esta página está destinada para mostrar la información completa correspondiente a cada empleado, además se puede utilizar para cargar la información del usuario en sesión.

1	Ulises Pacheco Crafter	141 pts	▼
2	Carlos Sirias Admin	133 pts	▼
3	Kristel Najera Apprentice	128 pts	▼
4	Saul I Zamora Admin	127 pts	▼
5	Jose Alvarez Admin	106 pts	▼

Rows per page: 5 | 1-5 of 18 | < >

Copyright 2018. Pernix Solutions.

**Figura 5.11** Tablero de posiciones

La página de tablero de posiciones es similar a un componente de listado con ciertas diferencias esenciales. Por ejemplo, las primeras 3 posiciones que poseen una franja de color indicando la primera posición con color dorado, la segunda con plateado y la tercera con color bronce. Además cada ítem de la lista posee una columna que indica la cantidad de puntos de cada participante.

# Capítulo 6

## Desarrollo de la plataforma de software

Al haber concluido la etapa de diseño de la plataforma, se obtuvieron los diseños, diagramas y demás que serán utilizados como guía para el desarrollo. Como metodología de desarrollo, se utiliza scrum, debido a su flexibilidad y adaptación al cambio. Considerando que los involucrados o “stakeholders” en el proyecto son un grupo de aproximadamente 6 personas, es muy probable que estos puedan generar diferentes opiniones y, por ende, cambios en los requerimientos.



## 6.1. Tareas realizadas para el desarrollo de la plataforma de software.

**Entrenamiento:** Cuando se va a iniciar el desarrollo de un proyecto es importante conocer las tecnologías a utilizar, ya que de esta forma se puede aprovechar al máximo su potencial. Existen miles de librerías y herramientas que se pueden combinar y crear un conjunto de herramientas amigable con el desarrollador.

Durante el entrenamiento se busca conocer las características mejor en ambiente de desarrollo y explorar las herramientas a utilizar, en este caso los lenguajes Ruby y JavaScript. Estos lenguajes vienen acompañados con sus respectivas librerías y frameworks, en este caso react.js y Ruby on Rails.

### **Instalación de herramientas:**

- a. *Visual studio code:* Editor de código utilizado para el desarrollo del proyecto, este posee extensiones muy útiles. Por ejemplo, los “linters”, estos son como correctores ortográficos de código, los cuales denotan como errores el código que no coincida con el estándar del lenguaje en cuestión.
- a. *Ruby:* Lenguaje de programación de alto nivel, interpretado y de fácil lectura. Este se puede instalar con los manejador de dependencias Rbenv o RVM
- a. *Node :* Node js es un framework que permite ejecutar javascript del lado del servidor, además posee un gestor de dependencias llamado npm, el cual es indispensable en cualquier proyecto de javascript.
- a. *Postgres :* Para la instalación de postgres en el sistema operativo ubuntu, solo se requiere ejecutar el comando `sudo apt-get install postgresql`, luego de realizar este paso de debe crear un usuario y luego queda listo para crear bases de datos.
- a. *Insomnia :* Cliente rest, es una herramienta muy útil para hacer peticiones http a un servidor web. De esta manera se pueden probar los servicios web sin necesidad de que estén implementados en vistas de la interfaz.

**Configuración del ambiente:** La configuración implica muchos aspectos muy importantes a la hora de iniciar el desarrollo de un software, no hay nada más satisfactorio para un programador que tener todo su ambiente de desarrollo configurado y listo para programar. Pero para lograr esto primero se deben descargar e instalar una serie de requisitos de software y herramientas de programación. Dentro de las funciones requeridas para la configuración del ambiente podemos mencionar:

1. Crear repositorios: Consiste en crear repositorios en alguna plataforma de control de versiones y definir una metodología para el desarrollo.
2. Inicializar los directorios de los proyectos: Este punto se refiere a la generación de los “boilerplate” o estructura inicial de directorios para cada proyecto, server y cliente.
3. Creación de usuarios de base de datos: Para acceder al motor de base de datos es necesario crear un usuario para manejar las bases asociadas al proyecto.
4. Crear archivos de configuración de ambiente (.env) para cada proyecto e incluir las variables necesarias.

## 6.2. Resultados obtenidos en el desarrollo de la plataforma de software.

Para el desarrollo del proyecto se utilizó la metodología scrum, dividiendo el proyecto en varios entregables. Para cada entregable se determinan los requerimientos o historias de usuario que compondrán cada etapa en el desarrollo, generando un producto incremental que permite una mayor adopción al cambio.

A Continuación, se explican los detalles correspondientes a los resultados obtenidos en cada una de las etapas y demos del producto desarrollado.

**Demo 1:** Esta demo se presentó el día lunes 27 de agosto, durante el staffing meeting de Pernix en la que participaron los altos cargos de la empresa. En esta primera demo se presentaron cuatro funcionalidades en las que se trabajó durante aproximadamente 3 semanas, a esto sumado 2 semanas de planning y training de las tecnologías.

Características de la aplicación desarrolladas para la demo:

- A. Sistema de autenticación.
- B. Sesiones y perfil de usuario.
- C. Componentes de navegación.
- D. Protección de rutas basadas en roles.

**Demo 2:** Esta demo se presentó el día miércoles 10 de septiembre, esta vez en una reunión con Cristina Hernández y Fernando Cardoce. En esta segunda demo se presentaron 6 funcionalidades contando con las correcciones de la primera demo.

Características de la aplicación desarrolladas para la demo:

- A. Registrar empleados.
- B. Editar empleados.
- C. Editar perfil de empleado.
- D. Ver listado de empleados.
- E. Registrar proyectos.
- F. Editar proyectos.

**Demo 3:** Esta demo se presentó la tercera iteración del proyecto el día lunes 24 de septiembre, durante el staffing meeting de Pernix concluyendo exitosamente un sprint más. Para esta demo se presentaron cinco funcionalidades en las que se trabajó durante aproximadamente 2 semanas.

Características de la aplicación desarrolladas para la demo:

- A. Mostrar listado de proyectos.
- B. Mostrar perfil de proyecto.
- C. Mostrar listado de equipos.
- D. Editar equipos.
- E. Crear equipos.

**Demo 4:** Esta demo se presentó la cuarta iteración del proyecto el día lunes 8 de octubre, durante el staffing meeting de Pernix realizando una demostración de los avances realizados en el proyecto hasta la fecha. Para esta demo se presentaron cinco funcionalidades en las que se trabajó

durante aproximadamente 2 semanas.

Características de la aplicación desarrolladas para la demo:

- A. Agregar miembros a un equipo.
- B. Eliminar miembros de un equipo.
- C. Ver detalles de un equipo.
- D. Agregar “Skills”
- E. Editar “Skills”
- F. Ver listado de “Skills”

**Demo 5:** Esta demo se presentó la quinta iteración del proyecto el día lunes 22 de octubre, durante el staffing meeting de Pernix realizando una demostración de los avances realizados en el proyecto hasta la fecha. Para esta demo se presentaron cinco funcionalidades en las que se trabajó durante aproximadamente 2 semanas.

Características de la aplicación desarrolladas para la demo:

- A. Mostrar listado de clientes.
- B. Mostrar detalles de clientes.
- C. Editar información de clientes.
- D. Agregar nuevos clientes.
- E. Eliminar clientes de la aplicación.

**Demo 6:** Esta demo se presentó la sexta iteración del proyecto el día lunes 8 de noviembre, durante el staffing meeting de Pernix realizando una demostración de los resultados finales y últimas características agregadas al proyecto correspondientes al módulo “scoreboard”

Características de la aplicación desarrolladas para la demo:

- F. Mostrar listado de actividades.
- G. Mostrar detalles de actividades.
- H. Editar información de actividades.
- I. Agregar nuevas actividades.
- J. Agregar participantes a las actividades (sumar puntos)

- K. Mostrar tablero de posiciones
- L. Mostrar modo presentación del tablero

# Capítulo 7

## 7.1 Tareas realizadas para la evaluación de la plataforma de software.

**Pruebas de snapshots:** Consiste en generar un snapshot o captura del estado, correspondientes a los módulos de redux que maneja las acciones sobre el estado compartido de cada entidad. Los snapshots sirven como punto de referencia para probar que las acciones se están manejando de forma adecuada según el comportamiento esperado por el snapshot con respecto al estado inicial

**Pruebas unitarias:** En su gran mayoría realizadas a nivel de servidor, se encargan de validar el correcto funcionamiento de cada una de las partes o unidades que conforman el sistema. Se utilizaron gemas y dependencias que permiten la generación de datos aleatorios. Los cuales casi siempre son requeridos para la ejecución de los casos de prueba.

**Pruebas de rendimiento:** Corresponden a casos de prueba en los que se propone realizar x acción en el sistema. Tomando en cuenta el tiempo que le tomó al usuario realizar mediante la metodología actual utilizado google drive. Comparando los resultados al realizarse la acción equivalente mediante la plataforma desarrollada.

## 7.2 Resultados obtenidos de la evaluación de la plataforma de software.

**Código de la aplicación cliente:** Una vez generados las suite de pruebas y teniendo instaladas las dependencias y módulos de node. Solo queda entrar a la terminal en la ubicación raíz del proyecto y digitar el comando `yarn test` y la aplicación que gestiona los casos de prueba procederá a

ejecutarlos, obteniendo el siguiente resultado.

```
Test Suites: 14 passed, 14 total
Tests:      61 passed, 61 total
Snapshots:  51 passed, 51 total
Time:       2.687s
Ran all test suites.

Watch Usage: Press w to show more.
```

**Figura 7.1 Resultados de pruebas**

Como se muestra en la **Figura 11** como resultado de la ejecución se observa que todos los casos de prueba y suites fueron completados exitosamente en un tiempo de 2.68 segundos.

**Código del servidor:** Como se mencionaba anteriormente para realizar las pruebas unitarias en el lado del servidor se utilizó la librería rspec, con la cual podemos probar el código correspondiente a los modelos y controladores que componen el servidor. asegurando el correcto funcionamiento del código según lo esperado. También se requiere de alguna librería que permita la generación de datos de prueba.

```
→ pernix-central-server git:(develop) X rspec spec
Run options: include {:focus=>true}

All examples were filtered out; ignoring {:focus=>true}

AuthorizeApiRequest
#call
  when valid request
    returns employee object
  when invalid request
    when missing token
      raises a MissingToken error
    when invalid token
      raises an InvalidToken error
    when token is expired
      raises ExceptionHandler::ExpiredSignature error
    fake token
      handles JWT::DecodeError

EmployeeAuthentication
#authenticate
  when valid credentials
    assigns the employee
    assigns an auth token
  when invalid credentials
    raises an authentication error

ApplicationController
#authorize_request
  when auth token is passed
    sets the current employee
  when auth token is not passed
    raises MissingToken error
```

**Figura 7.2 Ejemplo de test con rspec.**

Para una vez desarrollados los casos de prueba del lado del servidor, ejecutarlos tan sencillo como abrir una terminal, posicionarse en la carpeta raíz del proyecto en rails y escribir el comando rspec spec e inmediatamente la aplicación empezará a correr uno a uno todos los casos de prueba y

mostrando el resultado en consola.

```
Activity
  validations
    should validate that :name cannot be empty/falsy
    should validate that :points cannot be empty/falsy
    should validate that :frequency cannot be empty/falsy
    should validate that :points looks like a number greater than 0
  associations
    should have many participants

Employee
  validations
    should validate that :first_name cannot be empty/falsy
    should validate that :last_name cannot be empty/falsy
    should validate that :email cannot be empty/falsy
    should validate that :birthdate cannot be empty/falsy
    should validate that :password_digest cannot be empty/falsy
    should validate that :email is case-sensitively unique
  associations
    should have one schedule
    should have many participants
    should have many activities through participants
    should have many tcp_skills
    should have many skills through tcp_skills
    should have and belong to many teams
  model actions
    create a default scheduke after create an employee

Participant
  validations
    should validate that :date cannot be empty/falsy
  associations
    should belong to activity
    should belong to employee
```

*Figura 7.3 Ejemplo de test con rspec.*

En la figura anterior se puede observar la ejecución de pruebas correspondientes a cada modelo de active record, estos tests se basan en probar las validaciones y que la creación de los modelos cumpla con la estructura definida en cada uno.



```
WorkDay
  associations
    should belong to schedule
    should have many periods

Activities API
  GET index
    returns all the activities
  GET show
    gets the correct activity
  POST activity
    succeeds
    creates the new skill
  PUT activity
    succeeds
    updates activity
    update (add) participants
    updates the employee scores

Authentication
  POST /auth/login
    When request is valid
      returns an authentication token
      returns the employee
    When request is invalid
      returns a failure message
```

*Figura 7.4 Ejemplo de test con rspec 2*

El caso de prueba observado figura anterior corresponde a los servicios o end-points, obteniendo resultados satisfactorios aprobando cada uno de los casos. El color verde y el contenido del mensaje en la consola indican que se está probando en ese momento y si fue exitoso o erróneo según el color.

### ***Comparaciones en tiempos de respuesta***

Prueba 1: Para esta prueba el escenario es el siguiente: Se plantea una prueba en la que se edite la información de un proyecto específicamente el cambio de estatus del mismo, comparando la metodología antigua utilizando documentos xls con el nuevo sistema digital:



**Métricas:**

1. Mac mini (Mid 2011)
2. 2.3 GHz Intel Core i5
3. 16 GB 1333 MHz DDR3
4. Ancho de banda en descarga 36.45 mps
5. Ancho de banda en carga 37.27 mps
6. Navegador google chrome

**Resultados:** Como resultados se obtuvo una diferencia de 12 segundos entre los tiempos obtenidos por cada caso de prueba.

Tiempo de respuesta con Pernix Central: 6.23 seg.

Tiempo de respuesta con documentos xls. 18.3 seg.

# Capítulo 8

## Conclusiones

Con respecto a la finalización del proyecto, y cumplimiento de objetivos, los mismos fueron abordados satisfactoriamente, al haber realizado y completado las tareas definidas para cada una de las etapas de planificación, diseño, desarrollo y evaluación definidas en los objetivos. Como resultado, se obtuvo un producto de calidad basado en los requerimientos establecidos inicialmente.

**Procesos:** Como conclusiones referentes a los procesos relacionados al proyecto, podemos rescatar que al obtener una herramienta para la gestión de la información de la empresa se garantiza una respuesta rápida en cuanto a la toma de decisiones y manejo de las actividades que se realizan, se podrán agilizar muchos procesos y de esta forma centralizar la información. Logrando lo anterior se puede ahorrar tiempo ya que el acceso a la información se facilita. A su vez a largo plazo los tiempos ahorrados gracias a la solución podría convertirse en ganancias.

Ahorrar tiempo al final se traduce en ahorrar dinero. Además, que considerando el crecimiento de la empresa al generarse mayor volumen de información se llegara a volver cada vez más necesaria la implementación de esta plataforma. Que con los años van a generarse datos suficientes que en un futuro puedan convertirse en datos estadísticos de los cuales se pueda minar datos.

**Tecnologías:** Durante el desarrollo del proyecto se utilizó un stack de desarrollo muy dinámico y flexible. Las herramientas utilizadas ofrecen un ecosistema efectivo y fácil de acoplar. Gracias a la librería redux utilizada en el frontend, la división de la aplicación web en módulos fue bastante clara. Además con complementos como redux-saga es fácil poder orquestar la gestión de peticiones asíncronas así como los efectos y cambios que las mismas pueden generar sobre el estado de la aplicación. Pude reconocer ciertas debilidades que presenta react.js como librería por si sola es útil para construir interfaces dinámicas. no obstante sus funcionalidades se limitan a eso por lo cual es altamente recomendable incorporar librerías externas para complementarse. Además, es muy útil para conectar múltiples componentes en un mismo estado o módulo.

Por otra parte, con respecto a ruby y el backend, puedo recalcar ciertas fortalezas muy marcadas de el framework Rails, una de ellas son las gemas, existe una gran cantidad de estas librerías e implementarlas en el proyecto muchas veces se ahorra tiempo y esfuerzo porque cada gema es fácil de acoplar, y en general tienen buena documentación, por ejemplo, mediante la gema paperclip se puede delegar los métodos para el tratamiento de imágenes, redimensionar, generar varios tamaños entre otros.

**Experiencia profesional:** Al realizar el proyecto con éxito y concluir el proceso de práctica profesional, se genera conocimiento muy valioso y experiencia, que en muchos casos por lo general los estudiantes carecen de ella cuando están en la universidad. En la academia nos desenvolvemos de forma distinta, quizás un poco alejados de la realidad del mercado y entorno laboral. El cual no solo es un ambiente retador si no muy competitivo y riguroso. Gracias a la experiencia adquirida durante las 16 semanas laborando como practicante puedo tener una idea más esclarecida sobre cómo es un ambiente laboral serio y como se dice popularmente, cómo se hacen las cosas en “la calle” . Durante este periodo se tuvo la oportunidad de poner en práctica toda la teoría y enseñanzas adquiridas durante mi carrera universitaria como estudiante de

ingeniería.

**Trabajos futuros:** Como trabajos a futuro se pretenden incorporar nuevos módulos, como la generación de reportes en pdf correspondientes a los proyectos y su información. También se quiere emplear un tipo de trofeos o logros para los empleados que se tengan más puntos en el tablero de posiciones, para ello se deben agregar temporadas trimestrales para poder reiniciar el tablero de posiciones y asignar los trofeos a los ganadores de esa temporada. Esto con el fin de poder recompensar de alguna forma a los contribuyentes con mayor cantidad de trofeos y por ende un mayor desempeño

## Bibliografía

1. [1] Pernix SA. The Playbook,
2. [2] Sánchez, G. S. (2013, Junio 20). Sistemas ERP – Enterprise Resource Planning - Historia y Evolución (I). Consultado 15, Julio, 2018, en <http://stratic.es/erp-i-historia-y-evolucion/>
3. [3] Scott Klein. "Professional WCF programming". 2007. Wiley publishing.
4. [4] Fred A. Cummins, "Enterprise Integration: An Architecture for Enterprise Application and Systems Integration". 2002. Wiley.
5. [5] Gian Trotta (2003). Integration Best Practices - Dancing around EAI: Consultado el 7 de Noviembre de 2018 en [http://www.ebizq.net/topics/int\\_sbp/features/3463.html](http://www.ebizq.net/topics/int_sbp/features/3463.html)

6. [6] Wexing Song and Deng Dao Li, “Status and development in Chinese enterprises information resources management,” Overview of Management, No 10 ,2008 , pagina 50-52.
7. [7] Huiqun Huang (2009) “Evaluate Enterprise Resource Planning Based on Rough-set Unascertained Model” consultado el 8 de Noviembre de 2018.