

# Documento de Arquitectura de Software

## IEEE-1471-2000

### Control del documento

---

#### **Proyecto**

*Sistema de Administración de Ítems de Proyecto*

#### **Título**

*Arquitectura del Sistema – [v1.0 al 18 de Agosto de 2019]*

#### **Generado por**

*Augusto Alderete Fruet*

*Verónica Espínola Recalde*

*Sanny Martínez Cáceres*

*Nelson Gómez Alarcón*

#### **Aprobado por**

*Prof. Lilian Dematei*

*Prof. Claudia Rolón*

*Prof. Lilian Riveros*

# 1. Introducción

## 1.1 Propósito

Este documento proporciona una descripción comprensiva arquitectónica del sistema, usando un número finito de vistas diferentes para representar los distintos aspectos que se requieren para capturar y transportar las decisiones significativas que han sido hechas sobre el sistema.

## 1.2 Alcance

El presente documento contiene el diseño elaborado para el proyecto **Sistema de Administración de Ítems de Proyecto**, el cual es producto de un análisis minucioso de los requisitos del sistema, según estos pueden ser satisfechos con las tecnologías y características discutidas con los clientes y usuarios.

El documento está organizado alrededor de tres ideas principales.

1. Las características generales del diseño
2. Los requisitos atendidos por el diseño
3. Los modelos y vistas que lo detallan

Al contrario de muchas otras actividades técnicas, el desarrollo de sistemas intensivos en software dedica la mayoría de sus esfuerzos a la especificación y modelado.

Los modelos son utilizados tanto para el análisis de requisitos, como para el diseño de la solución, así como para la especificación, construcción y despliegue del sistema en su ambiente de explotación.

Los modelos son presentados por vistas o diagramas, generalmente utilizando notaciones gráficas como el UML.

Por otro lado, los programas de computadora son construidos por medio del uso de herramientas de traducción automáticas llamados compiladores, para los cuales es construida la forma lineal y más detallada del software del sistema: el código fuente.

La última sección del documento indica la forma en que se puede obtener el código fuente del proyecto, así como las instrucciones de compilación necesarias para lograr la ejecución de los componentes que este código detalla.

Este documento ha sido generado directamente del análisis del **Sistema de Administración de Ítems de Proyecto** y el modelo de diseño puesto e implementado en StarUML.

StarUML es una herramienta para el modelamiento de software basado en los estándares UML (Unified Modeling Language) y MDA (Model Driven Architecture).

### 1.3 Usuarios Interesados

Este documento de Arquitectura de Software (DAS), puede ser usado por todos aquellos usuarios que deseen comprender el diseño y construcción de la aplicación de Sistema de Configuración de Software, y sirve como base para que los desarrolladores de software puedan construir el bajo nivel de la aplicación usando el lenguaje que más les acomode.

### 1.4 Recomendaciones de conformidad con esta práctica.

N/A.

## 2. Referencias

Las referencias aplicables a este documento son:

- *Especificaciones de los requisitos según la IEEE 830*
- *Architecture Tradeoff Analysis Method*
- *ISO 9126 -2001 Calidad del Software y Métricas de evaluación*
- *The 4+1 View .Kruchten - 1009*

## 3. Definiciones, acrónimos y abreviaciones.

DAS: Documento de Arquitectura de Software

ADMINISTRACION DE ITEMS DE PROYECTO: sistema de gestión que permite el control y seguimiento de los ítems en un proyecto de elaboración de software.

HTTP: Protocolo de Trasferencia de Hipertexto.

TCP: Protocolo de control de transmisión.

ARQUITECTURA DE SOFTWARE: conjunto de elementos estáticos, propios del diseño intelectual del sistema, que definen y dan forma tanto al código fuente, como al comportamiento del software en tiempo de ejecución. Naturalmente este diseño arquitectónico ha de ajustarse a las necesidades y requisitos del proyecto.

DESCRIPCION DE ARQUITECTURA: colección de productos de documentación.

VISTAS: es una representación de un área de interés o perspectiva del sistema en alto nivel.

TIPOS DE VISTAS: especificación de una convención de cómo construir y usar una vista. Deben satisfacer la capacidad de creación y análisis de una vista.

STAKEHOLDER: Individuo, equipo u organización con intereses relativos al sistema.

ESCENARIO: especifica el comportamiento y limita el interés de un área específica del sistema para uno o varios stakeholders.

MODULO O COMPONENTE: cualquier elemento estructural abstracto, visible, externo, de alto nivel, analizable, que pueda constituir una funcionalidad de la solución del sistema.

ATRIBUTOS DE CALIDAD: un atributo de calidad, es una cualidad deseable de la solución, que pueda manifestarse en forma de requerimiento no funcional, que pueda ser medible, testeable y finalmente evaluable.

## **4. Framework Conceptual**

### **4.1 Descripción de la arquitectura en contexto**

Este documento presenta la arquitectura como una serie de vistas basadas en la arquitectura de software del modelo 4+1 de Kruchten. Estas vistas son: la vista de escenarios, la vista lógica, la vista de desarrollos, la vista física, la vista de procesos. No hay ninguna vista separada de una misma implementación, descrita en este documento. Estas vistas están hechas sobre Lenguaje de modelo unificado (UML) en su versión 3.0 desarrolladas usando StarUML en su versión 3.1.0.

Los estilos arquitectónicos serán referenciados en este documento de arquitectura, según las recomendaciones de la Arquitectura de software del modelo 4+1 de Kruchten.

### **4.2 Stakeholders y sus roles**

Este documento representa la identificación de Stakeholders y sus roles a partir de la interpretación de los casos de uso del Negocio.

### **4.3 Actividades de arquitectura en el ciclo de vida**

N/A.

### **4.4 Usos de las descripciones de arquitectura**

Las descripciones de arquitectura de este documento se usarán para referenciar el diseño del sistema de software de ADMINISTRACION DE ITEMS DE PROYECTO.

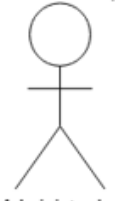
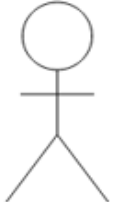
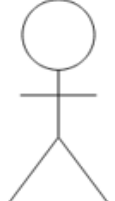
## **5. Descripciones prácticas de arquitectura**

N/A.

### **5.1 Documentación de la arquitectura**

N/A.

## 5.2 Identificación de los Stakeholders y sus responsabilidades

Stakeholder	descripción	escenario	Vistas
 Administrador	Es el usuario encargado de la creación del proyecto y la asignación correspondiente de roles a los usuarios.	<ul style="list-style-type: none"> <li>• Escenario de negocios</li> <li>• Escenario de diseño</li> </ul>	<ul style="list-style-type: none"> <li>• CU Crear Proyecto</li> <li>• CU Asignar Líder</li> <li>• CU Asignar Roles</li> <li>• CU Identificar Usuario</li> </ul>
 Lider de Proyecto	Es la persona encargada de crear y realizar la actualización de los estados de ítems, fases y líneas bases del proyecto.	<ul style="list-style-type: none"> <li>• Escenario de negocios</li> <li>• Escenario de diseño</li> </ul>	<ul style="list-style-type: none"> <li>• CU Crear ítem</li> <li>• CU Crear Línea Base</li> <li>• CU Añadir ítem a LB</li> <li>• CU Actualizar estado de ítem</li> <li>• CU Actualizar estado de LB</li> <li>• CU Identificar Usuario</li> <li>• CU Imprimir informes</li> </ul>
 Desarrollador	Es la persona encargada de realizar el desarrollo del software del proyecto.	<ul style="list-style-type: none"> <li>• Escenario de negocios</li> <li>• Escenario de diseño</li> </ul>	<ul style="list-style-type: none"> <li>• CU Identificar Usuario</li> <li>• CU Modificar descripción de ítem</li> </ul>

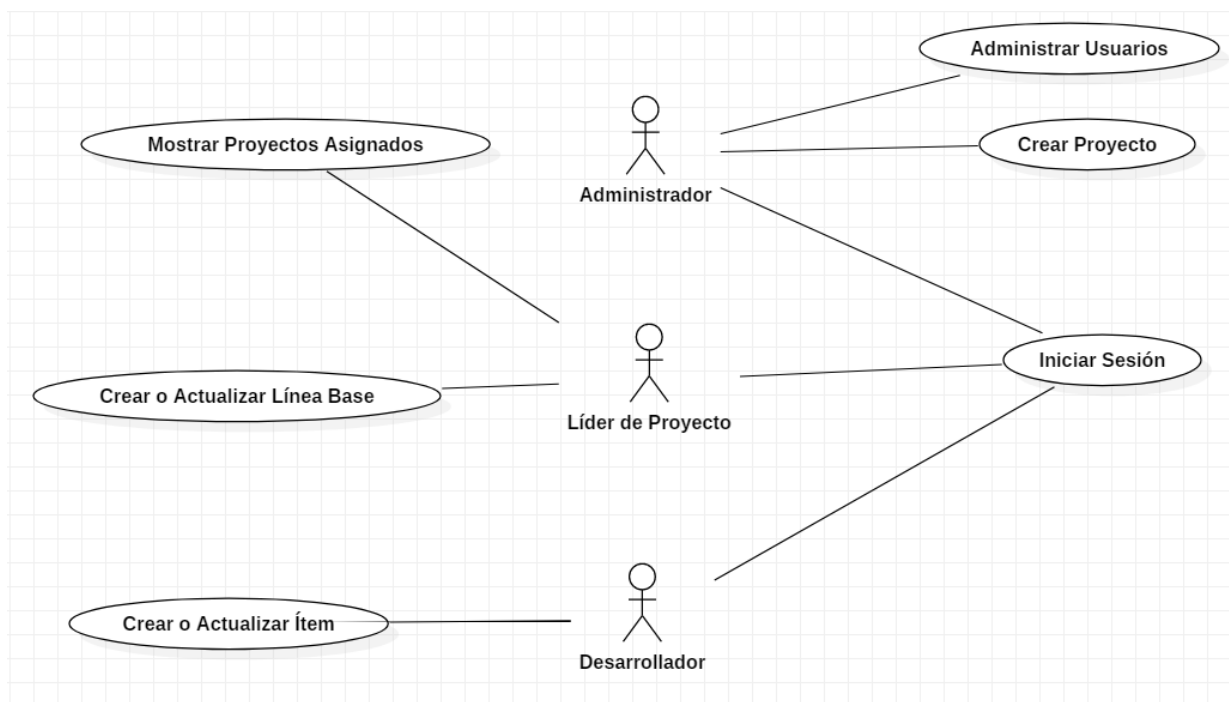
### 5.3 Selección de puntos de vista de la arquitectura.

Vistas	UML
Escenarios	Casos de uso
Lógica	Clases
Procesos	Secuencia

### 5.4 Vistas de la arquitectura

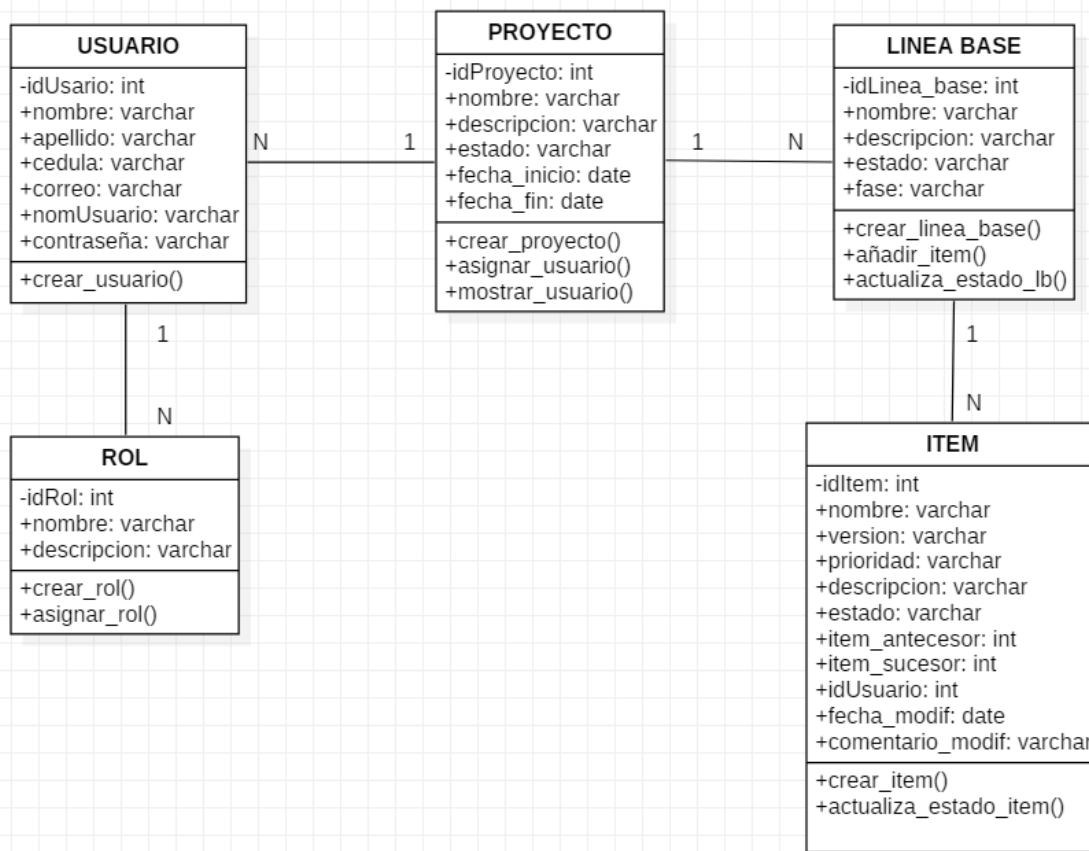
Vistas.- escenarios

Diagrama.- Caso de uso del negocio – Caso de uso de diseño



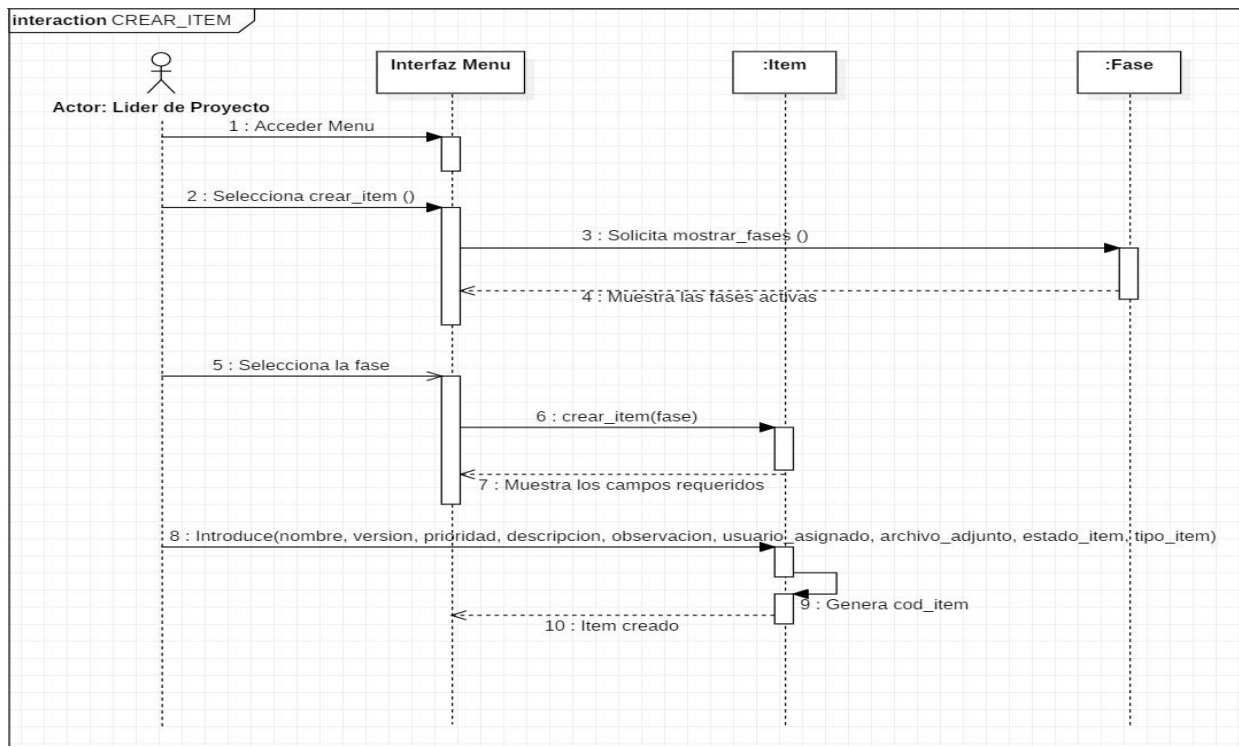
## Vista.- Lógica

### Diagramas.- Clases



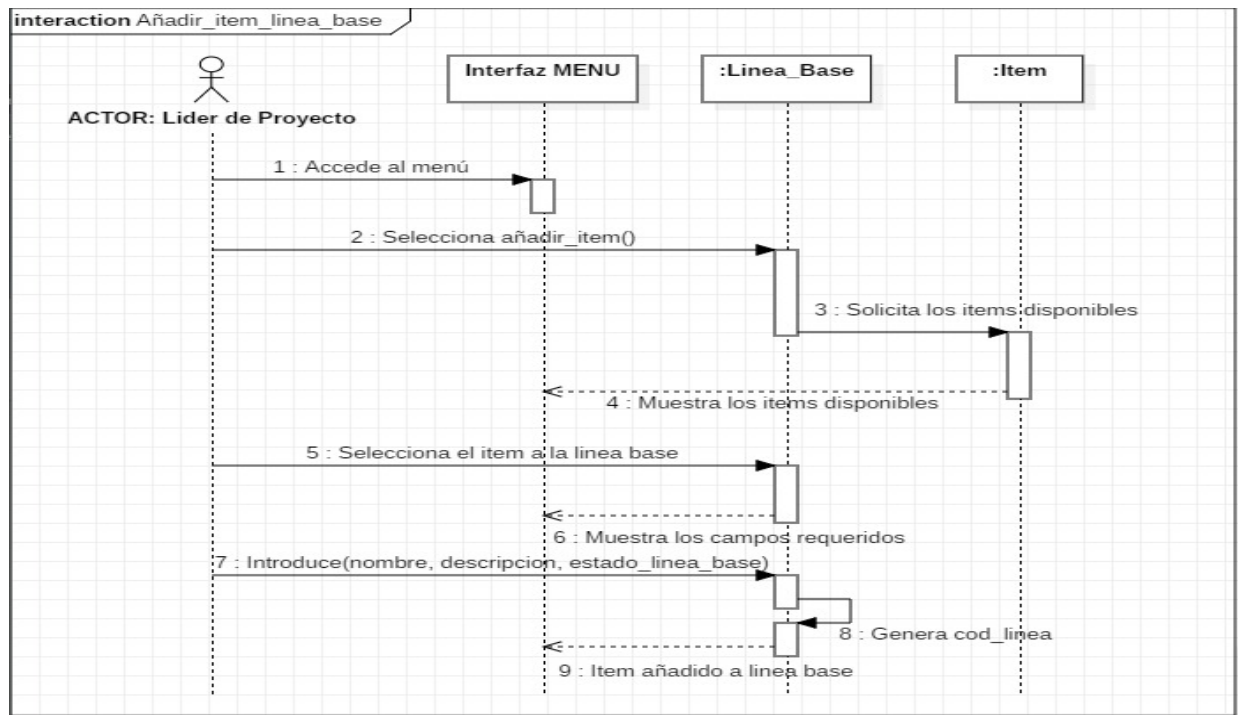
## Vista.- Procesos

### Diagrama.- Secuencia (Crear\_item)

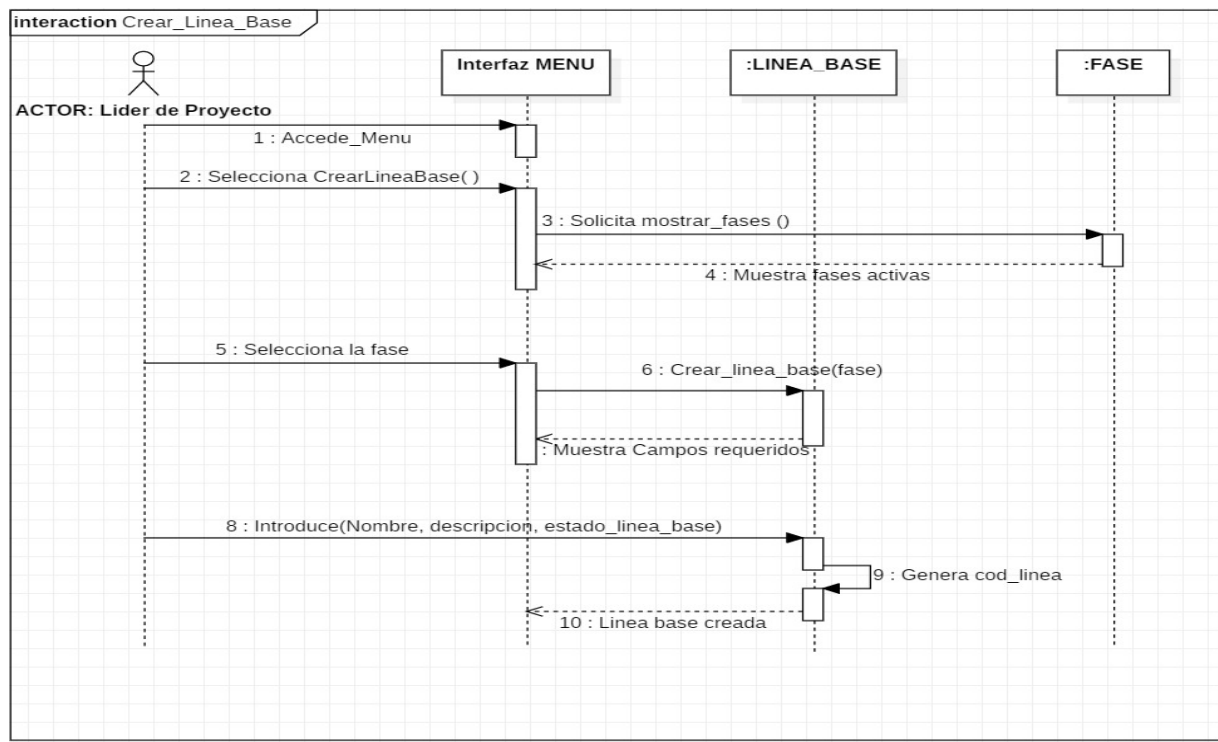




## Diagrama de secuencia (Añadir\_item\_linea\_base)



## Diagrama de secuencia (Crear\_linea\_base)



## 5.5 Consistencia en la cantidad de vistas de la arquitectura.

### DESCRIPCION DE MODULOS

Nombre del módulo	descripción	Componentes incluidos
Administración	Módulo que permite la administración de usuarios y roles, además de configuraciones del sistema	<ul style="list-style-type: none"><li>• Crear_usuario</li><li>• Crear_rol</li><li>• Asignar_rol</li><li>• </li></ul>
Gestión de Configuración	Módulo que permite la generación de líneas bases. Deberá permitir el control de las líneas bases generadas por producto y la administración de los cambios de los elementos que se encuentren en las líneas bases.	<ul style="list-style-type: none"><li>• Crear_linea_base</li><li>• Actualiza_estado_lb</li><li>• Añadir_item</li></ul>
Desarrollo	Módulo que administra todos los elementos de los productos.	<ul style="list-style-type: none"><li>• Crear_item</li><li>• Actualiza_estado_item</li><li>• Crear_proyecto</li></ul>

### DESCRIPCION DE COMPONENTES

Nombre del componente	descripción	Componentes relacionados
Ítems	Contiene la lógica para: Crear nuevos ítems y actualizar los estados de los ítems.	<ul style="list-style-type: none"><li>• Líneas</li></ul>
Proyectos	Contiene la lógica para: la creación del proyecto a llevarse a cabo.	<ul style="list-style-type: none"><li>• Líneas</li></ul>

Líneas	Contiene la lógica para almacenar las líneas bases abiertas con el detalle de los ítems asignados.	<ul style="list-style-type: none"> <li>• Ítems</li> <li>• Proyectos</li> </ul>
Users	Contiene la lógica para almacenar los datos de los usuarios que acceden al sistema.	<ul style="list-style-type: none"> <li>• Permission_user</li> <li>• Rol_user</li> </ul>
Roles	Contiene el detalle de los roles	<ul style="list-style-type: none"> <li>• Rol_user</li> <li>• Permission_role</li> </ul>
Role_user	Contiene la lógica de los roles asignados a los usuarios	<ul style="list-style-type: none"> <li>• Users</li> <li>• Roles</li> </ul>
Permission_role	Contiene la lógica de los permisos asignados a los roles	<ul style="list-style-type: none"> <li>• Roles</li> <li>• Permissions</li> </ul>
Permissions	Contiene el detalle de los permisos	<ul style="list-style-type: none"> <li>• Permission_user</li> <li>• Permission_role</li> </ul>
Permission_user	Contiene el detalle de los permisos de los usuarios	<ul style="list-style-type: none"> <li>• Users</li> <li>• Permissions</li> </ul>

## **DESCRIPCION DE CONECTORES**

### **5.6 Arquitectura lógica.**

#### **Performances**

La arquitectura de software escogida apoya a los requerimientos no funcionales y requerimientos de arquitectura de sistemas descritos en los anexos de este documento.

1. El inicio del sistema cuenta con un tiempo límite de 10 segundos.
2. El sistema permite la conexión simultánea de hasta 100 usuarios, garantizando el buen desempeño del mismo.
3. Requerirá el espacio de disco de menos de 20 MB y la RAM de 32 MB.

#### **Calidad**

La arquitectura de software apoya las exigencias de calidad, como estipulado en la especificación anexa a este documento.

1. El interfaz de usuario será WEB.
2. El interfaz de usuario del Sistema ADMINISTRACION DE ITEMS DE PROYECTO será diseñado para la facilidad de uso y será apropiado para asegurar las normas de usabilidad universal.
3. El sistema cuenta con una interfaz de ayuda para disipar dudas y facilitar la utilización al usuario.
4. Las contraseñas e informaciones de los usuarios se encuentran cifrados de manera a garantizar la seguridad de los mismos.

### **5.7 Ejemplo de uso.**

N/A.

### **5.8 Detalles de la implementación**

La especificación de un sistema intensivo en software tiene como última representación al código fuente de los componentes. Este código indica los más finos detalles del software, por medio de un lenguaje preciso, capaz de ser traducido automáticamente a instrucciones de la máquina. Acompaña al código, las llamadas *previsiones de compilación*, constituidos por todos los elementos de soporte necesarios para realizar la construcción de los componentes a partir del conjunto de códigos. Esta

sección detalla la obtención y uso del paquete de código fuente para el proyecto. De manera de facilitar el uso de este, para las futuras ampliaciones o correcciones del sistema.

### 5.8.1 Lenguajes y plataformas

La lógica de diseño arquitectónico aplicada en este documento, abre la posibilidad de que la implementación de bajo nivel sea efectuada con el lenguaje de programación PHP y como motor de base de datos utilizaremos **MySQL**.

La curva de aprendizaje de ambos lenguajes es muy baja, por lo tanto, es bastante sencillo aprender y aplicar tanto **PHP** como **MySQL**.

**PHP** tiene una interacción muy buena con HTML para crear sitios web de una forma sencilla.

Tiene una comunidad muy amplia donde resolver las dudas, tanto oficial como extraoficialmente.

Los entornos de desarrollo de **PHP** y de **MySQL** son fáciles de utilizar y de configurar y ambos son lenguajes fiables, eficientes y fáciles de usar.

**Laravel** es un framework con una comunidad muy activa que orbita a su alrededor, esto permite que el framework crezca y siempre existan paquetes y herramientas nuevas disponibles para su uso.