**Documento de Arquitectura**

**Project Planning Redmine (PPR)**

1. **Introducción**

En este documento se pretende detallar la arquitectura de dicho sistema; Tales como, los requerimientos funcionales, los atributos de calidad y las tácticas utilizadas para lograr el cumplimiento de cada atributo de calidad; De igual modo, se describen las principales vistas de la arquitectura; De esta forma se asegura de que el enfoque de diseño de un sistema provea un rendimiento aceptable.

1. **Fondo**

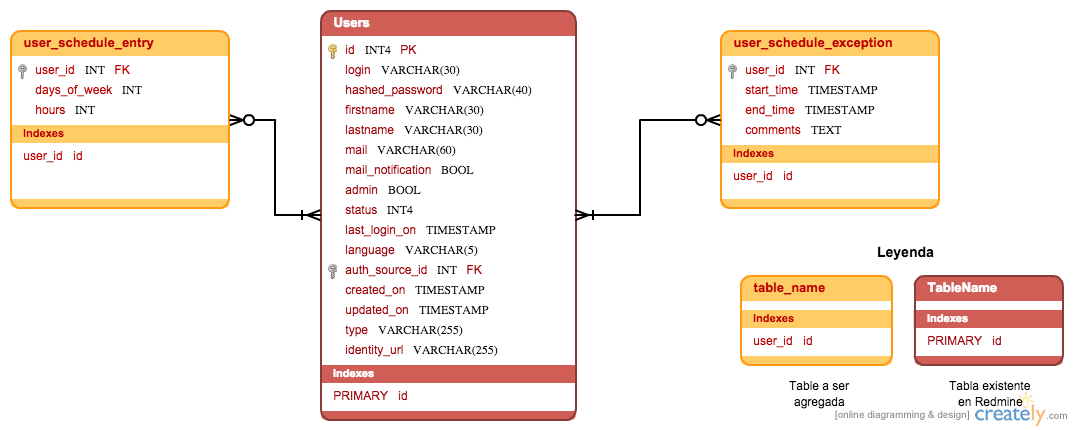
Este proyecto nace con la necesidad de automatizar y eficientizar el uso de las funcionalidades de una herramienta para la gestión de proyectos que incluye un sistema de seguimiento de incidentes con seguimiento de errores, llamado Redmine.

Aunque Redmine posee la funcionalidad de calendarizar, el objetivo principal de este es automatizar las acciones de planificación mostrando cual es la disponibilidad de los recursos que tiene el cliente y poder calendarizar, basado en las dependencias que exista entre las tareas dadas por el administrador.

Este proyecto consta de dos roles principales Encardo de proyecto y Gerente de proyectos. El primero se encarga de crear un evento en horario de un recurso y ver calendarización de uno o más recursos en un rango de fechas, el segundo es una generalización del anterior, su función es calendarizar proyecto y establecer cantidad de horas disponible por día de un recurso.

Finalmente lo que se quiere lograr con este proyecto es crear un visual allocation para ver la disponibilidad de los recursos en la semana. Es un Plugin de lo que realmente ofrece Redmine.

1. **Diagrama de contexto**
2. **Tácticas y patrones**
   1. **Rendimiento**
      1. Incrementar la eficiencia computacional.
   2. **Seguridad**
      1. Autenticar usuarios
      2. Autorizar usuarios
3. **Descomposición**
   1. **Diagrama**
   2. **Catálogo**
   3. **Definición**
4. **Modelo físico de data**
   1. **Diagrama**

****

* 1. **Catálogo**

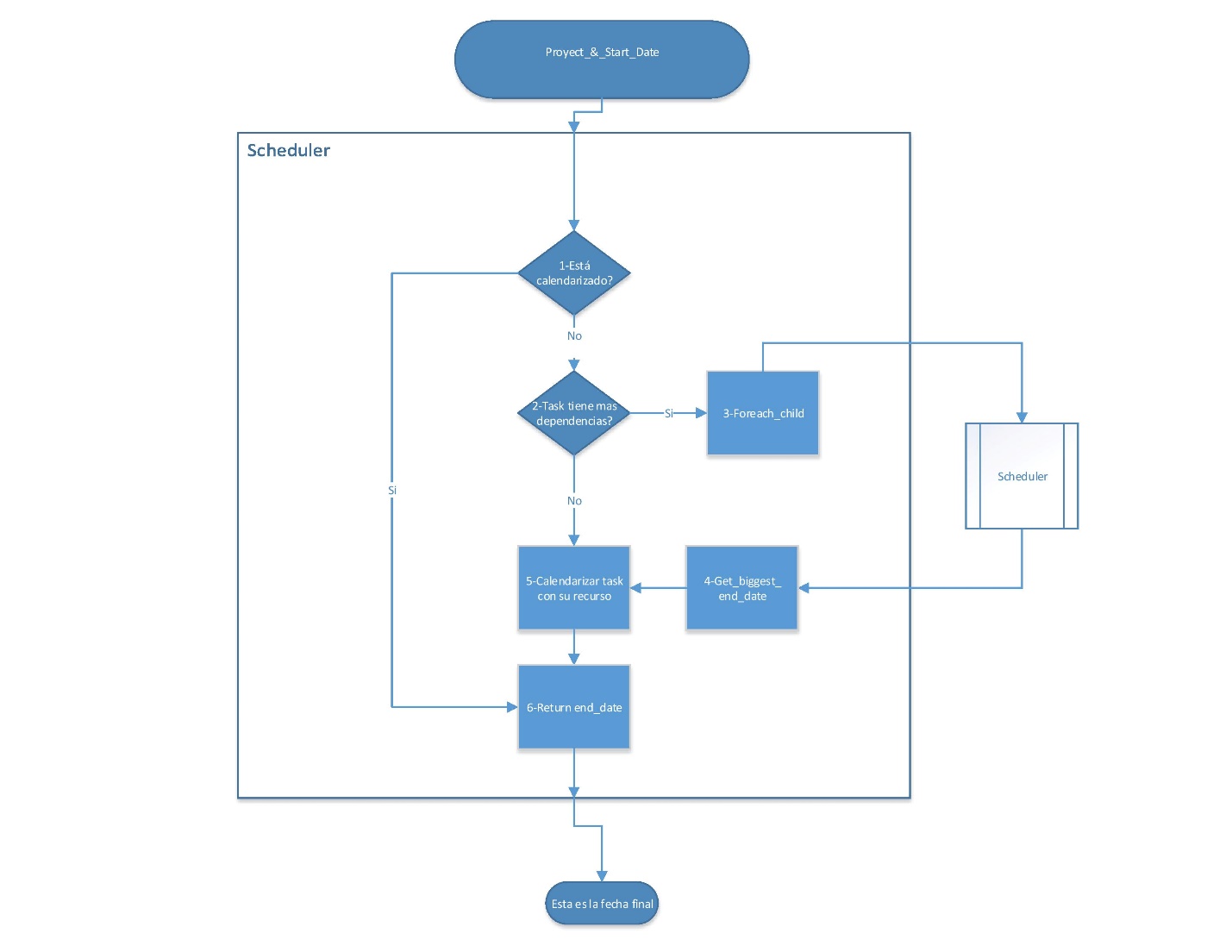
|  |  |
| --- | --- |
| **Elementos** | * Tablas existentes * Tablas a crear |
| **Relaciones** | * Uno a uno * Uno a muchos * Muchos a uno * Mucho a muchos |
| **Propiedades** | * Nombre * Responsabilidad |

* 1. **Definición**

|  |  |
| --- | --- |
| **Nombre** | user\_schedule\_entry |
| **Responsabilidad** | La responsabilidad de esta tabla es almacenar la disponibilidad por día en horas que posee un recurso. |

|  |  |
| --- | --- |
| **Nombre** | user\_schedule\_exception |
| **Responsabilidad** | La responsabilidad de esta tabla es almacenar las excepciones por las cuales un recurso no pueda cumplir con su horario de trabajo. |

1. **Diagrama de flujo del algoritmo de calendarización**
   1. **Diagrama**

****

* 1. **Catálogo**

|  |  |
| --- | --- |
| **Elementos** | * Punto de inicio o final * Decisión * Proceso * Proceso predefinido |
| **Relaciones** |  |
| **Propiedades** | * Nombre * Tipo de elemento * Acción |

* 1. **Definición**

|  |  |
| --- | --- |
| **Nombre** | Project & Start Date |
| **Tipo de elemento** | Punto de inicio |
| **Acción** |  |

|  |  |
| --- | --- |
| **Nombre** | Está calendarizada? |
| **Tipo de elemento** | Decisión |
| **Acción** | Verifica si una tarea está calendarizada. Si está calendarizada, pasa a devolver la fecha de calendarización de dicha tarea. De lo contrario, pasa a verificar si tiene dependencias. |

|  |  |
| --- | --- |
| **Nombre** | Tiene dependencias? |
| **Tipo de elemento** | Decisión |
| **Acción** | Verifica si una tarea tiene dependencias. Si tiene dependencias, pasa a recorrerlas una a una. De lo contrario pasa a calendarizar la tarea con respecto al recurso asignado. |

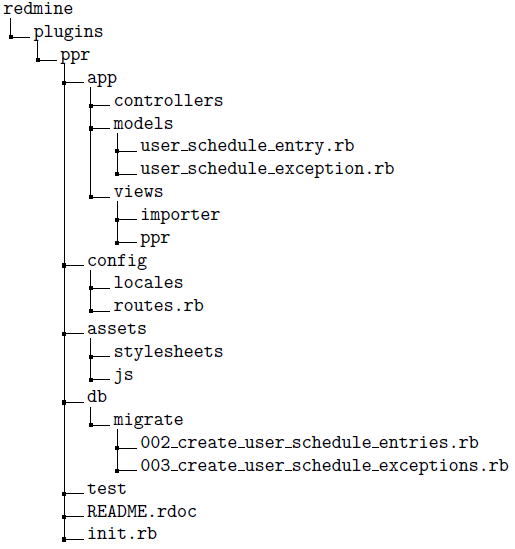
|  |  |
| --- | --- |
| **Nombre** | For each child |
| **Tipo de elemento** | Proceso |
| **Acción** | Recorre todas las dependencias de una tarea, y por cada una llama al proceso predefinido Scheduler. |

|  |  |
| --- | --- |
| **Nombre** | Get biggest\_end\_date |
| **Tipo de elemento** | Proceso |
| **Acción** | Obtiene la última fecha en la cual una tarea fue asignada y pasa a calendarizar la tarea con respecto al recurso asignado. |

|  |  |
| --- | --- |
| **Nombre** | Calendarizar task con su recurso |
| **Tipo de elemento** | Proceso |
| **Acción** | Calendariza la tarea tomando en cuenta la disponibilidad del recurso al cual fue asignada dicha tarea. |

|  |  |
| --- | --- |
| **Nombre** | Return end\_date |
| **Tipo de elemento** | Proceso |
| **Acción** | Devuelve la fecha a la cual la tarea fue asignada. |

1. **Implementación**
   1. **Diagrama**

****

* 1. **Catálogo**

|  |  |
| --- | --- |
| **Elementos** | * Directorios * Archivos |
| **Relaciones** | * Asignado a |

1. **Conclusión**

En este documento hemos demostrado como implementaremos la solución de Project Planning Redmine.

PPR será desarrollada como un plugin el cual expandirá las funciones del plugin Importer. El cual, es actualmente utilizado por el cliente para calendarizar sus tareas. PPR pretende automatizar este proceso de calendarización tomando en cuenta las tareas proveídas por el cliente. Así como también información sobre los recursos humanos del proyecto. Tales como: los horarios disponibles, y excepciones a estos horarios.

Como se puede ver en los mockups mostrados, donde sea posible, PPR se integrará a las vistas de Redmine para proveer un ambiente familiar a usuarios existentes de Redmine. Para el horario y excepciones de usuarios, se expandirá la vista de Usuarios de Redmine mediante el uso de Hooks. Para la calendarización, se expandirá las vistas de plugin Importer.