Sistema de alarmas para tutelas y derechos de petición de la Universidad del Quindío

Version 1.0

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 18/04/2016 | 1.0 | Creación del documento | Daniel Ramiro Garcia |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 4

1.1 Purpose 4

1.2 Scope 4

1.3 Definitions, Acronyms, and Abbreviations 4

1.4 References 4

1.5 Overview 4

2. Architectural Representation 5

3. Architectural Goals and Constraints 5

4. Use-Case View 6

5. Logical View 7

5.1 Overview 8

5.2 Architecturally Significant Design Packages 8

6. Process View 10

7. Deployment View 10

8. Implementation View 10

8.1 Overview 10

8.2 Layers 11

9. Data View (optional) 11

10. Size and Performance 11

11. Quality 11

# Introduction

La arquitectura es una descripción de las estructuras del sistema, es un artefacto que se debe analizar para determinar el logro de los atributos de calidad, provee una visión del modelo del proyecto, manifestando las decisiones de diseño [1].

En este documento se puede encontrar el desarrollo de componentes importantes en la determinación de una correcta arquitectura de software que soporte de manera adecuada el funcionamiento del sistema de trazabilidad de derechos de petición y tutelas en la Universidad del Quindío.

## Purpose

El propósito de este documento es tener un consenso en el diseño de la arquitectura de software para el sistema de trazabilidad de derechos de petición y tutelas en la Universidad del Quindío.

## Scope

El alcance de este documento, establece actividades que se encuentran involucradas durante el proceso de selección y aplicación de la arquitectura del software a desarrollar, define la distribución de los paquetes del sistema, garantizando que su implementación se realice siguiendo los estándares de calidad establecidos por el docente.

## Definitions, Acronyms, and Abbreviations

**HTML:** HyperText Markup Language (Lenguaje de Marcado de Hipertexto).

**PHP:** Hypertext PreProcesor (Pre Procesador de Hipertexto).

**SGBD:** Sistema de Gestión de Bases de Datos.

**CSS:** Cascade Style Sheets (Hojas de estilo en cascada).

## References

**[1].** Software Architecture in Practice 3rd Edition – Len Bass., Et All, obtenido el 17 de Abril de 2016, obtenido desde: [**http://ptgmedia.pearsoncmg.com/images/9780321815736/samplepages/0321815734.pdf**](http://ptgmedia.pearsoncmg.com/images/9780321815736/samplepages/0321815734.pdf)

**[2].** El modelo 4+1, vistas como guía de los workshops de la ingeniería de software – Sorey García, obtenido el 17 de Abril de 2016, obtenido desde: [**http://blog.soreygarcia.me/2013/04/el‐modelo‐4‐1‐vistas‐como‐guia‐de‐los.html**](http://blog.soreygarcia.me/2013/04/el)

**[3].** Entendiendo el Modelo – Vista – Controlador – CookBook, Obtenido el 19 de Abril de 2016, obtenido desde: [**http://book.cakephp.org/2.0/es/cakephp‐overview/understanding‐model‐view‐controller.html**](http://book.cakephp.org/2.0/es/cakephp)

## Overview

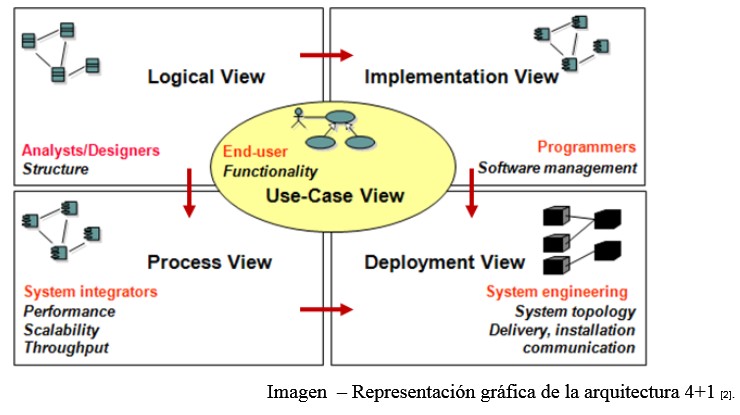
En las 10 secciones de este documento se detalla la arquitectura del software a desarrollar, para ello se presentan las siguientes secciones, así:

* + - Sección 1 resume la descripción del documento, definiciones, acrónimos y referencias.
    - Sección 2 describe la representación arquitectónica del proyecto.

* + - Sección 3 lista las metas y las restricciones del sistema.
    - Sección 4 a 8 describen las diferentes vistas de la que consta el sistema.
    - Sección 9 Describe la tabla de requerimientos no funcionales a tener en cuenta para la puesta en funcionamiento de la aplicación.
    - Sección 10 Incluye los componentes de calidad a tener en cuenta para la puesta en funcionamiento de la aplicación.

# Architectural Representation

Para este proyecto se utilizará el modelo arquitectónico “4+1” (Kruchten, 1995). Este modelo se basa en varios elementos arquitectónicos con el fin de satisfacer los requerimientos de funcionalidad y rendimiento del sistema, igualmente buscan satisfacer componentes no funcionales como lo son la confiabilidad, usabilidad, escalabilidad, entre otros [2].



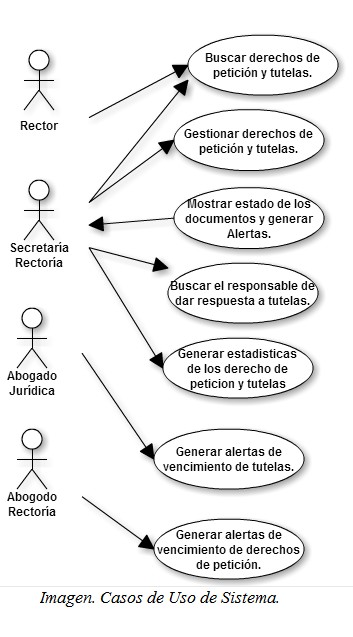
# Architectural Goals and Constraints

A continuación se nombran algunas metas y restricciones de la arquitectura.

* Basado en software libre tanto como sea posible*.*
* Utilización del framework con las características necesarias para el desarrollo de la aplicación.
* Los patrones elegidos para el desarrollo de la aplicación deben ser compatibles con las herramientas de desarrollo seleccionadas.
* Garantizar la disponibilidad de la información para los usuarios.
* Garantizar la seguridad de la información de los documentos y de los usuarios.

# Use-Case View

El diagrama de casos de uso representa la forma de cómo el usuario interactúa con el sistema a desarrollar, así podremos obtener el conjunto de escenarios que contendrá cada iteración.

Cada caso de uso tiene una descripción que especifica la funcionalidad que se incorporará al sistema propuesto.

**CU1­Gestionar derechos de petición y tutelas**

Este caso de uso de sistema permite registrar, modificar y eliminar derechos de petición y tutelas en la universidad del Quindío.

**CU2­Buscar derechos de petición y tutelas.**

Este caso de uso permite consultar los derechos de petición y las tutelas, y la respectiva información de cada uno.

**CU3­Mostrar estados de los documentos y generar alertas de vencimiento.**

Este caso de uso devuelve al usuario encargado el estado actual en que se encuentra el documento (derecho de petición o tutela) para continuó a esto generar alerta de vencimiento.

**CU4­Buscar el responsable de dar respuesta a los derechos de petición.**

Este caso de uso permite identificar la persona de la oficina de jurídica encargada de dar respuesta a una tutela.

**CU5­Generar estadísticas de los derechos de petición.**

Este caso de uso permite generar un reporte estadístico de los derechos de petición y tutelas recibidos vs resueltos.

**CU6­Generar alertas de vencimiento de tutelas**

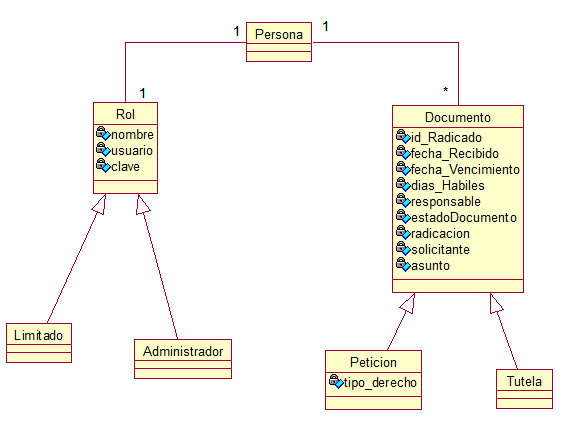
Este caso de uso permite generar alertas a medida que el tiempo de vencimiento de una tutela se va acortando.

**CU7­Generar alertas de vencimientos de derechos de petición.**

Este caso de uso permite generar alertas a medida que el tiempo de vencimiento de un derecho de petición se va acortando.

# Logical View

En la vista lógica se describe el diagrama de clases del proyecto sistema de alarmas para derechos de petición y tutelas en la Universidad del Quindío.



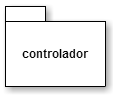
## Overview

Descripción: La vista codifica y mantiene la presentación final de nuestra aplicación de cara al usuario. Es decir, en la vista colocaremos todo el código HTML, CSS, JavaScript, etc. que se tiene que generar para producir la página tal cual queremos que la vea el usuario.



Descripción: Todo el código que tiene que ver con el acceso a base de datos. En el modelo mantendremos encapsulada la complejidad de nuestra base de datos y simplemente crearemos funciones para recibir, insertar, actualizar o borrar información de nuestras tablas. Al mantenerse todas las llamadas a la base de datos en un mismo código.



Descripción: El controlador podríamos decir que es la parte más importante, porque hace de enlace entre el modelo, la vista y cualquier otro recurso que se tenga que procesar en el servidor para generar la página web

## Architecturally Significant Design Packages

|  |  |
| --- | --- |
| Propiedad | Descripción |
| Nombre | Persona |
| Descripción | Es la clase principal donde se tienen personas que pueden ser usuarios del sistema o responsables de un documento (derecho petición, tutela) |
| atributos | id, nombre, apellido, estado |

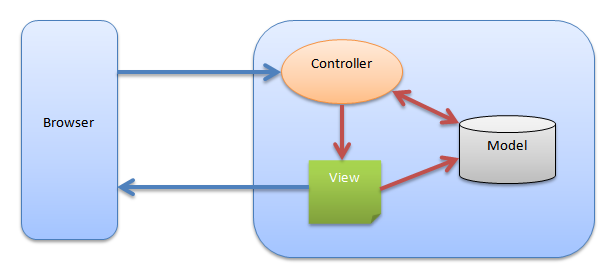
|  |  |
| --- | --- |
| Propiedad | Descripción |
| Nombre | Documento |
| Descripción | Es la clase encargada de crear los documentos (derecho de petición, tutela) que serán asignados a una persona, o la persona tendrá la información de uno o muchos documentos |
| atributos | id\_radicado, fecha\_recibido, fecha\_vencimiento, dias\_habiles, responsable |

|  |  |
| --- | --- |
| Propiedad | Descripción |
| Nombre | Rol |
| Descripción | Es la clase que define un rol específico para la persona un rol puede ser el de la secretaria |
| atributos | id\_Rol, nombre, privilegio |

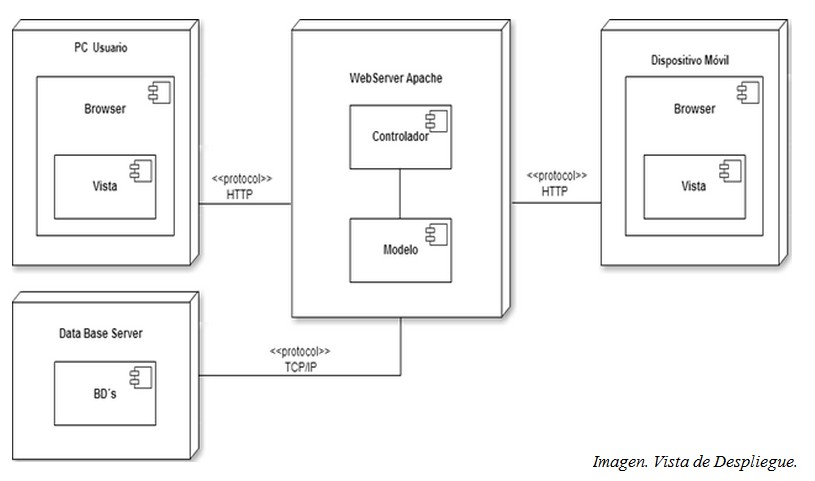
|  |  |
| --- | --- |
| Propiedad | Descripción |
| Nombre | Alarma |
| Descripción | En esta clase se pretende manejar un sistema de semaforización teniendo en cuenta tres estados fundamentales de vencimiento de la fecha de un documento, estos estados son:  estado Crítico: se trata de la fecha límite dada por el sistema al usuario para atender un derecho de petición o tutela  Estado Aceptable: es un estado intermedio entre un estado Crítico y un estado de Advertencia el cual quiere decir que la fecha de vencimiento de un documento tiene un tiempo prudente a ser atendida.  estado de Advertencia: en este estado es más que todo informativo, se permite ver información de los documentos próximos a vencerse pero con la certeza de tener un tiempo aún mayor que en el estado Aceptable y Crítico |
| atributos | estado, fecha , nombre |

# Process View

A través de un browser el usuario envía una petición, la cual es recibida por el controlador. Este último se encarga de procesar la petición y acceder al modelo (donde están los datos) para resolver dicha petición. Para luego enviar estos datos al controlador, ya que el controlador está vinculado con la vista en donde está la información que se le presenta al usuario. El siguiente grafico muestra la relación entre estos componentes.

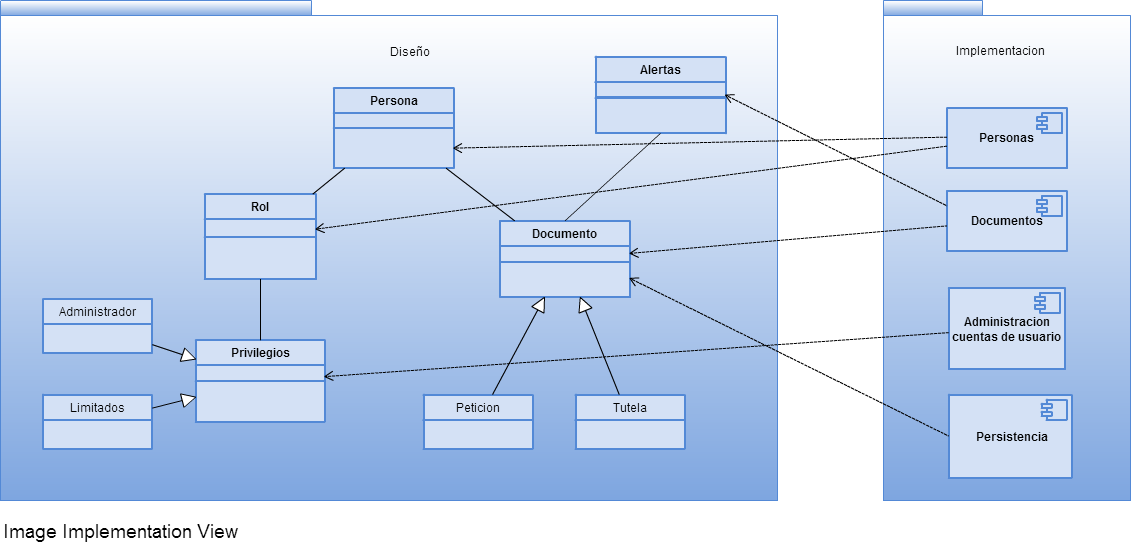


# Deployment View

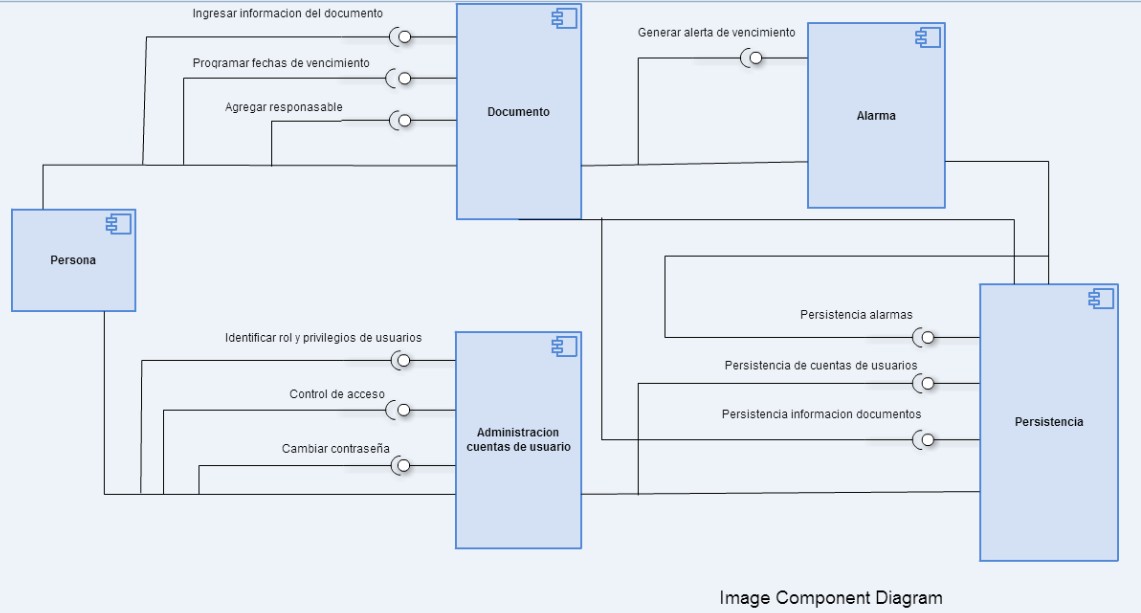
En la Deployment view se muestra el comportamiento del hardware para cumplir con las funcionalidades de la aplicación.

# Implementation View

La arquitectura de la aplicación se muestra con dos paquetes, de los cuales el primero muestra los diagramas de clase y el segundo la implementación. La siguiente imagen refleja la arquitectura de la aplicación.



## Overview

Los componentes que el sistema y sus relaciones contienen responsabilidades de las funciones de cada componente la siguiente imagen representa el comportamiento de los componentes de la aplicación.

## Layers

[For each layer, include a subsection with its name, an enumeration of the subsystems located in the layer, and a component diagram.]

# Data View (optional)

[A description of the persistent data storage perspective of the system. This section is optional if there is little or no persistent data, or the translation between the Design Model and the Data Model is trivial.]

# Size and Performance

[A description of the major dimensioning characteristics of the software that impact the architecture, as well as the target performance constraints.]

# Quality

[A description of how the software architecture contributes to all capabilities (other than functionality) of the system: extensibility, reliability, portability, and so on. If these characteristics have special significance, such as safety, security or privacy implications, they must be clearly delineated.]