Software Architecture Document

Estilo Arquitectónico tres capas

Lizzette Betancourt, Mishele Loján

Versión 1.0

Mayo 2015

Historial de revisiones

|  |  |  |  |
| --- | --- | --- | --- |
| Fecha | Versión | Descripción | Autor |
| 29/05/2015 | 1.0 | Documento de arquitectura de software- 3 capas. | Lizzette Betancourt,  Mishele Loján |

Tabla de contenido

[1. Introducción 4](#_Toc420673797)

[1.1 Propósito 4](#_Toc420673798)

[1.2 Alcance 4](#_Toc420673799)

[1.3 Definiciones, siglas y abreviaturas 4](#_Toc420673800)

[1.4 Referencias 4](#_Toc420673801)

[1.5 Overview 5](#_Toc420673802)

[2. Representación arquitectónica 5](#_Toc420673803)

[Vista Lógica 5](#_Toc420673804)

[Vista de Procesos 5](#_Toc420673805)

[Vista de desarrollo 6](#_Toc420673806)

[Vista Física 6](#_Toc420673807)

[Vista de Escenarios (+1) 6](#_Toc420673808)

[3. Limitaciones y objetivos arquitectónicos 6](#_Toc420673809)

[- Capa de presentación: 6](#_Toc420673810)

[- Capa de negocios 7](#_Toc420673811)

[- Capa de datos 7](#_Toc420673812)

[3.1 Seguridad 7](#_Toc420673813)

[3.2 Persistencia 7](#_Toc420673814)

[3.3 Confiabilidad / disponibilidad 7](#_Toc420673815)

[3.4 Rendimiento 8](#_Toc420673816)

[3.5 Mantenimiento 8](#_Toc420673817)

[4. Vista de Escenario – Casos de uso 8](#_Toc420673818)

[4.1 Actores 8](#_Toc420673819)

[5. Vista Lógica 11](#_Toc420673820)

[6. Vista de Proceso 11](#_Toc420673821)

[7. Vista de Desarrollo 12](#_Toc420673822)

[- Capa de presentación 12](#_Toc420673823)

[- Capa de negocio 12](#_Toc420673824)

[- Capa de datos 13](#_Toc420673825)

[8. Vista Física 13](#_Toc420673826)

[9. Tamaño y rendimiento 14](#_Toc420673827)

[10. Cuestiones y preocupaciones 14](#_Toc420673828)

# Introducción

El presente documento nos proporciona el desarrollo de la arquitectura, que permite mostrar cual es la estructura de la aplicación en la que se va a efectuar el estilo arquitectónico de tres capas, para que esto se haga posible se utilizará el modelo de vistas 4+1 que describe la arquitectura del software usando cinco vistas concurrentes que son: vista lógica, procesos, física, desarrollo y escenarios.

El estilo arquitectónico del sistema de reservación de habitaciones de un hotel se encuentra compuesto por 3 capas las cuales son: capa de presentación, capa de negocios y capa de datos que permite distribuir el trabajo de la creación de una aplicación por niveles o capas, en donde cada grupo de trabajo está totalmente abstraído del resto de niveles.

## Propósito

El propósito del presente Documento de arquitectura de software (SAD) es dar a conocer un enfoque de cómo está estructurada la aplicación de tres capas utilizando el modelo de vistas de Kruchten 4+1. Así mismo mostrará detalladamente cuales son los componentes que integran cada una de las vistas de dicho modelo.

## Alcance

El alcance del presente documento consiste en ilustrar cada una de las capas que conforman la aplicación utilizando el modelo de vistas de Kruchten 4+1 para resaltar las decisiones arquitectónicas consideradas para el aplicativo.

## Definiciones, siglas y abreviaturas

**4+1:** Architectural view model

**MySQL:** Sistema de gestión de base de datos relacional (RDBMS)

**SAD:** Software Architecture Document

**UML:** Unified Modeling Language

## Referencias

* López, Eliazar. (s.f). Arquitectura de n capas. Recuperado el 16 de Mayo de 2015. Disponible en: <http://www.academia.edu/10102692/Arquitectura_de_n_capas>
* Kruchten, Philippe (s.f) Planos Arquitectónicos: El Modelo de “4+1” Vistas de la Arquitectura del Software∗. Recuperado el 15 de Mayo de 2015. Disponible en: <http://cic.puj.edu.co/wiki/lib/exe/fetch.php?media=materias:modelo4_1.pdf>

## Resumen

Este documento de arquitectura de software está estructurado de la siguiente forma:

**Sección 2**: describe el uso de cada vista del modelo 4+1.

**Sección 3**: describe las limitaciones arquitectónicas de la aplicación.

**Sección 4:** Diagramas UML para representar la vista de casos de uso

**Sección 5:** Diagramas UML para representar la vista Lógica.

**Sección 6:** Diagramas UML para representar la vista de Procesos

**Sección 7:** Diagramas UML para representar la vista de desarrollo

**Sección 8:** Diagramas UML para representar la vista física.

**Sección 9:** Contendrá información acerca del rendimiento de la aplicación.

# Representación arquitectónica

El modelo de arquitectura de software está representado por las siguientes vistas:

## **Vista Lógica**

**Audiencia:** Diseñadores.

**Área**: En esta vista representa la funcionalidad que el sistema proporciona a los usuarios finales. Es decir representa lo que el sistema va hacer y cada una de las funciones que ofrece.

**Artefactos relacionados:** Diagrama de clases, diagrama paquetes y diagrama de secuencia

## **Vista de Procesos**

**Audiencia**: Integrador de sistemas.

**Área:** Muestra los procesos del sistema así como la comunicación de estos y se centra en el comportamiento en tiempo de ejecución del mismo.

**Artefactos relacionados:** Diagrama de actividades.

## **Vista de desarrollo**

**Audiencia:** Programadores

**Área:** mostrar cómo está dividido el sistema software en componentes y las dependencias que hay entre esos componentes.

**Artefactos relacionados:** Diagrama de componentes y diagrama de paquetes.

## **Vista Física**

**Audiencia:** Ingeniero en Sistemas.

**Área:** Muestra todos los componentes físicos del sistema y las conexiones entre estos componentes que conforman las soluciones que incluyen en los sistemas.

**Artefactos relacionados:** Diagrama de despliegue

## **Vista de Escenarios (+1)**

**Audiencia:** Interesados y usuarios finales

**Área:** representada por los casos de uso software y va a tener la función de unir y relacionar las otras 4 vistas, esto quiere decir que desde un caso de uso podemos ver cómo se van ligando las otras 4 vistas, con lo que tendremos una trazabilidad de componentes, clases, equipos, paquetes, etc., para realizar cada caso de uso**.**

**Artefactos relacionados:** Diagramas de casos de uso.

# Limitaciones y objetivos arquitectónicos

La arquitectura a ser utilizada en este aplicativo es la arquitectura 3 capas, las cuales se definen a continuación:

## Capa de presentación:

Esta capa es la que ve el usuario (también denominada capa de usuario), comunica información y captura información del usuario en un mínimo proceso. También es conocida como interfaz gráfica y debe tener características como ser entendible y accesible para el usuario. Esta capa solo se comunica con la capa de negocios ya que esta contiene la lógica del sistema, y cuando la capa de presentación envía solicitudes de información la capa de negocio le devuelve una respuesta. Para que exista la conexión entre esta capa y la capa de negocios se utiliza el patrón arquitectónico MVC (modelo–vista–controlador).

## Capa de negocios

Se denomina capa de negocio (también lógica de negocio) porque aquí es donde está la lógica y las reglas que deben cumplirse para la funcionalidad del sistema. Además en esta capa residen los programas que se ejecutan también recibe las peticiones del usuario y se envían respuestas tras el proceso. Esta capa se comunica con la capa de presentación al momento de recibir las solicitudes y presentar los resultados y con la capa de datos, para solicitar al gestor de base de datos almacenar o recuperar datos de él, para que exista conexión entre la capa de negocios y la capa se datos que utiliza el patrón arquitectónico DAO (Data Access Object).

## Capa de datos

En esta capa es donde residen los datos y es la encargada de acceder a los mismos. Está formada por uno o varios gestores de base de datos que realizan todo tipo de almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

## Seguridad

En este estilo de arquitectura, la seguridad se puede definir independientemente para cada servicio y en cada nivel. Aunque la capa que más seguridad debe tener es la capa de datos, ya que ésta es donde se alberga toda la información que contiene el sistema de gestión de base de datos.

## Persistencia

La persistencia se debe aplicar en la capa de datos ya que aquí reposan los datos en el gestor de base de datos

## Confiabilidad / disponibilidad

En la arquitectura de tres capas, las aplicaciones son descentralizadas es decir cada capa se especializa en una determinada tarea. Es por esto que permite un mayor grado de flexibilidad, mejor rendimiento, escalabilidad, disponibilidad y utilización de recursos.

## Rendimiento

La arquitectura de tres capas permite un mayor rendimiento ya que las capas están distribuidas, esto además permite la escalabilidad.

## Mantenimiento

Como esta arquitectura tiene varias capas al momento de dar mantenimiento no habría problema, ya que se podría hacer capa por capa, no dando mantenimiento al sistema como tal.

# Vista de Escenario

La vista de escenario permite representar la funcionalidad que tiene el sistema mediante el diagrama de casos de uso.

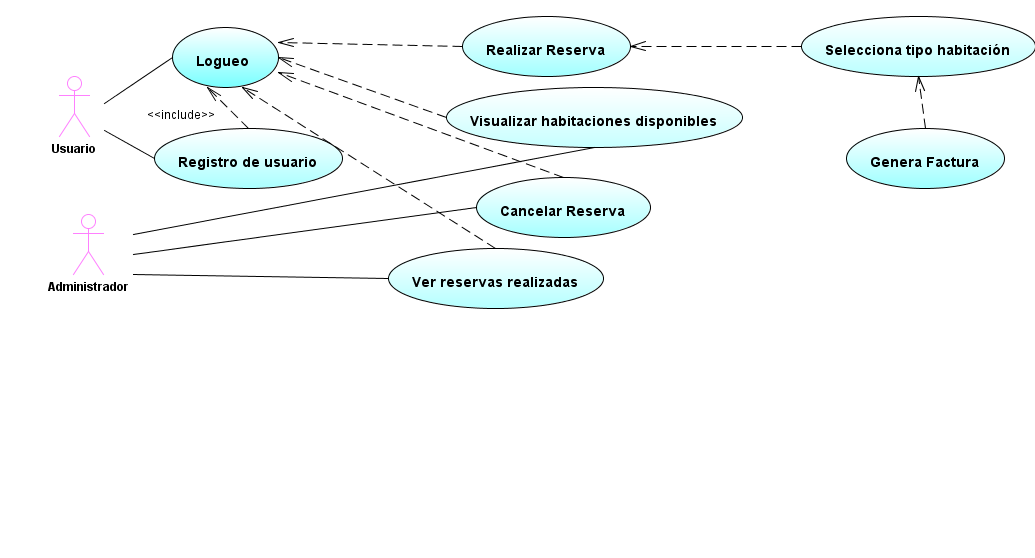


Fig. 1 Diagrama de caso de uso

## Actores

Los actores que formar parte de este sistema son:

**Administrador:** Es aquel que puede realizar acciones como ver reservas realizadas, visualizar habitaciones disponibles y cancelar reservas.

**Usuario:** Es aquel que realiza acciones como logueo, registrar usuario, visualizar habitaciones disponibles, realizar reservas, consultar reserva, ver reservas realizadas.

# Vista Lógica

A la vista lógica se la representa mediante el diagrama de clases en el cual se puede observar cuales son las relaciones entre las clases, los métodos y operaciones que se involucran en el sistema.

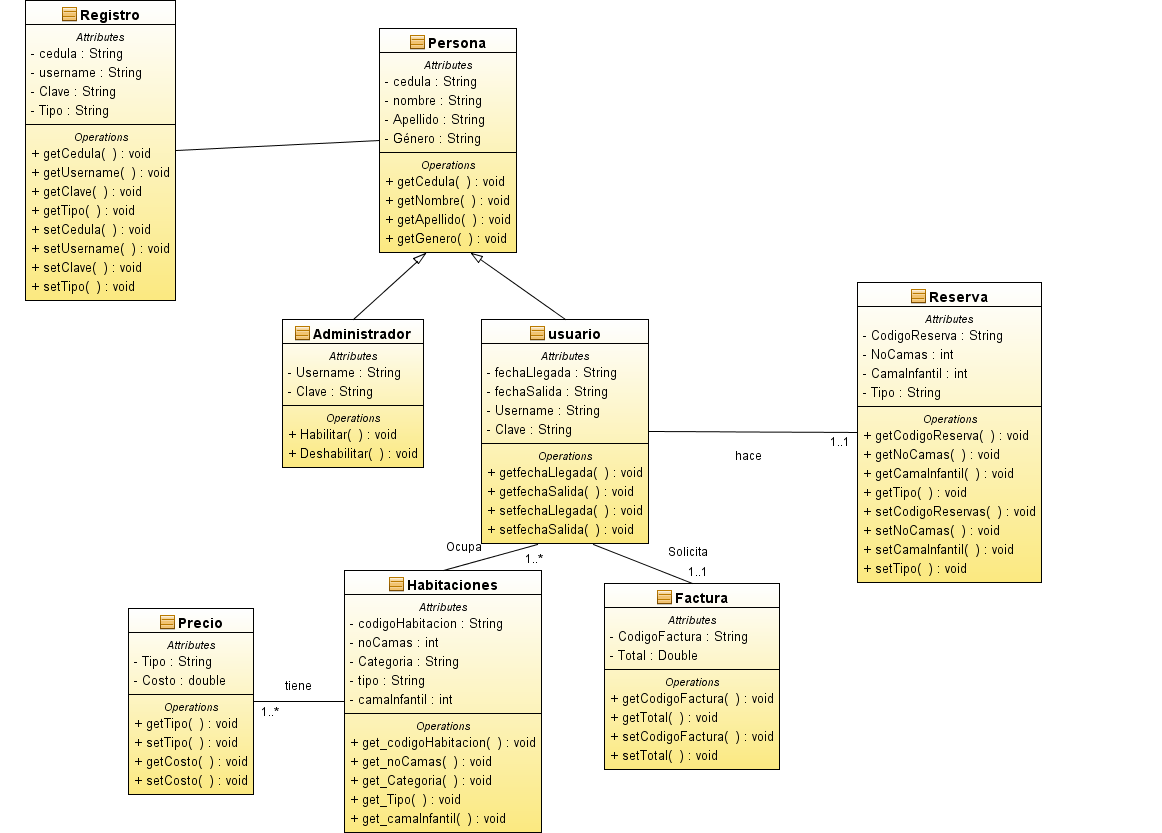


Fig. 2 Diagrama de clases

# Vista de Proceso

Para poder representar la vista de procesos se la hace mediante el diagrama de actividades, que permite mostrar el flujo de procesos y el comportamiento del sistema.

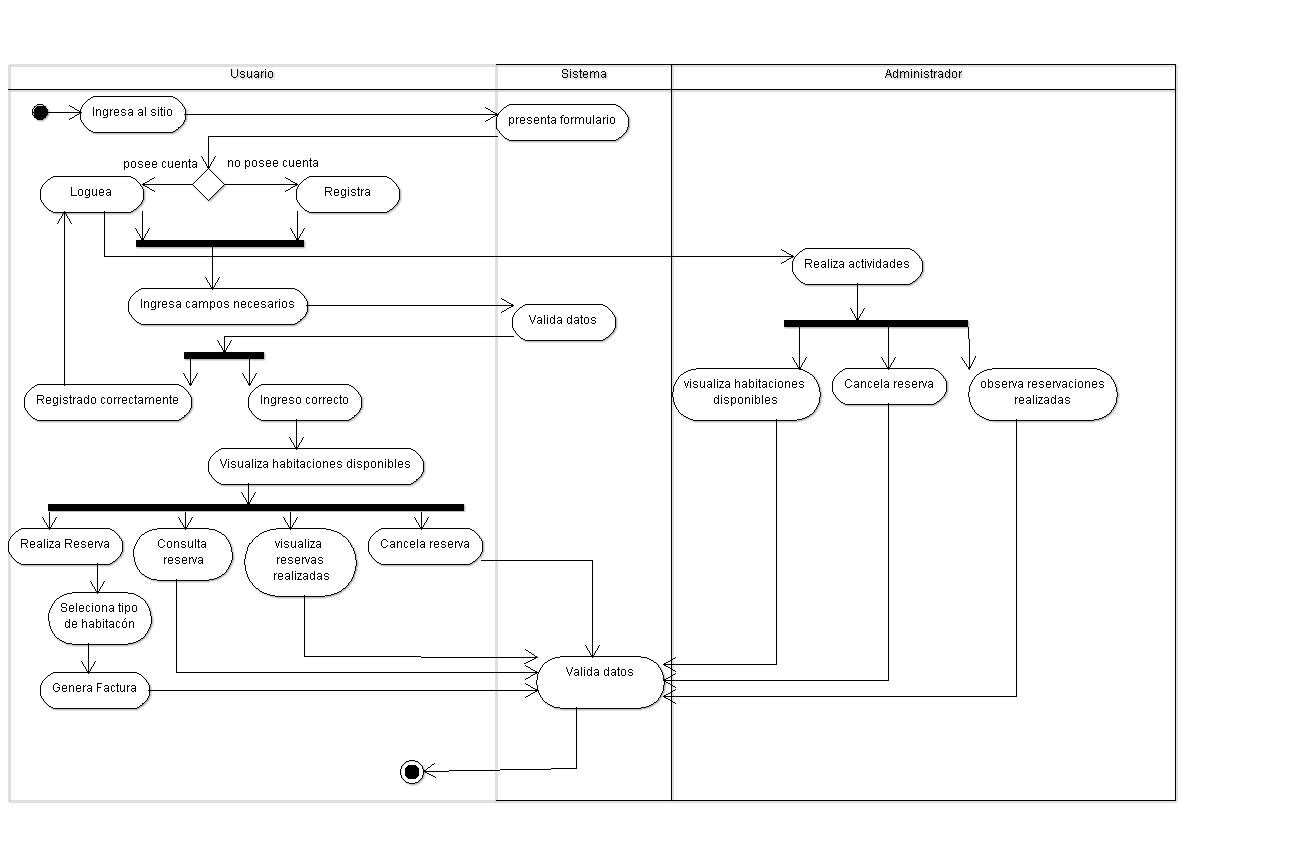


Fig. 3 Diagrama de actividades

# Vista de Desarrollo

Esta vista presenta la organización estática del sistema. El sistema de reservación de habitaciones de un hotel consta de tres capas presenta dependencias y componente para su correcto funcionamiento que son:

## Capa de presentación

Esta capa está compuesta por interfaces que verá el usuario, dependiendo de las acciones que realice. Esta capa se comunica a la capa de negocio el patrón MVC, que consiste en unir estas dos capas excluyendo a la capa de datos.

## Capa de negocio

Aquí se presenta la lógica el contiene el sistema que el usuario no puede ver. La conexión que tiene con la capa de datos la realiza mediante el patrón arquitectónico DAO (Data Access Object).

## Capa de datos

Esta capa contiene el almacenamiento de los datos del sistema, que es el motor de base de datos MYSQL.

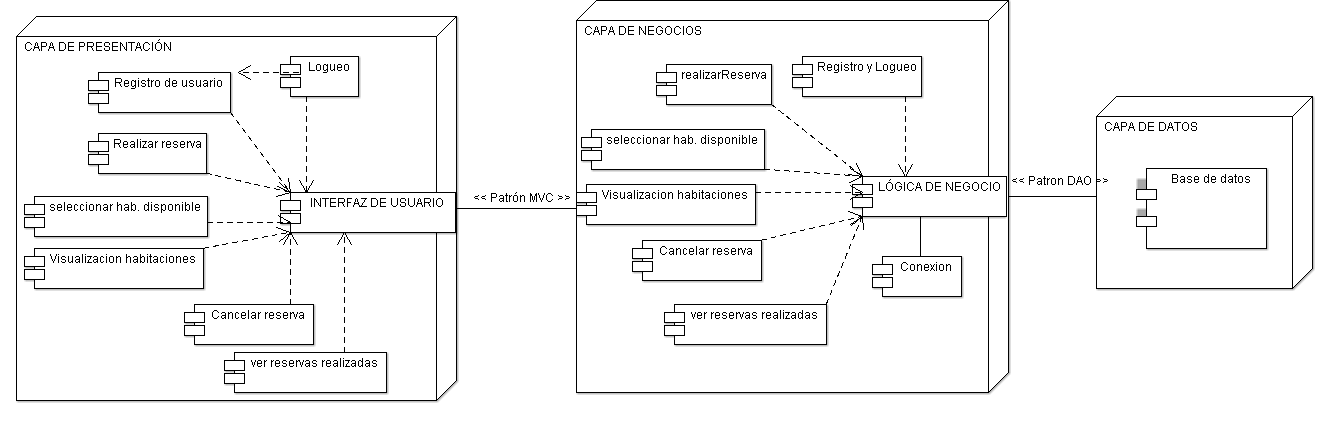


Fig. 4 Diagrama de componentes

# Vista Física

El sistema de reservación de habitaciones de un hotel en tres capas se encuentra compuesta por varias funciones que nos permiten realizar el intercambio de datos entre varias capas de un servicio web; todas las capas que componen este sistema pueden estar en un mismo ordenador, pero también la capa de presentación puede estar en un ordenador y la siguientes capas en otro.

## Capa de presentación

La conexión entre la presente capa con el servidor de aplicación se realizará a través de del protocolo Http con los métodos Get y Post.

* Ordenador Windows / Linux
* Navegador web: Permite observar los servicios que ofrece el sistema.
* Html5: Lenguaje de programación para maquetar una página web.
* Css3: Contiene los estilos que se añadirán a la página web, para mejorar su presentación hacia el usuario

## Capa de negocio

La conexión entre esta capa y la capa de datos se realizara mediante el protocolo TCP/IP.

Para esta capa utilizaremos un Servidor de aplicación:

* Apache: Servidor Web que acepta o recibe las peticiones procesadas a través del navegador.
* PHP: lenguaje que permite la creación de funciones para el proceso de funcionamiento del sistema.

## Capa de negocio

* Motor de base de datos MYSQL: Permite gestionar los datos del sistema, creando tablas para de acuerdo a la información requerida.

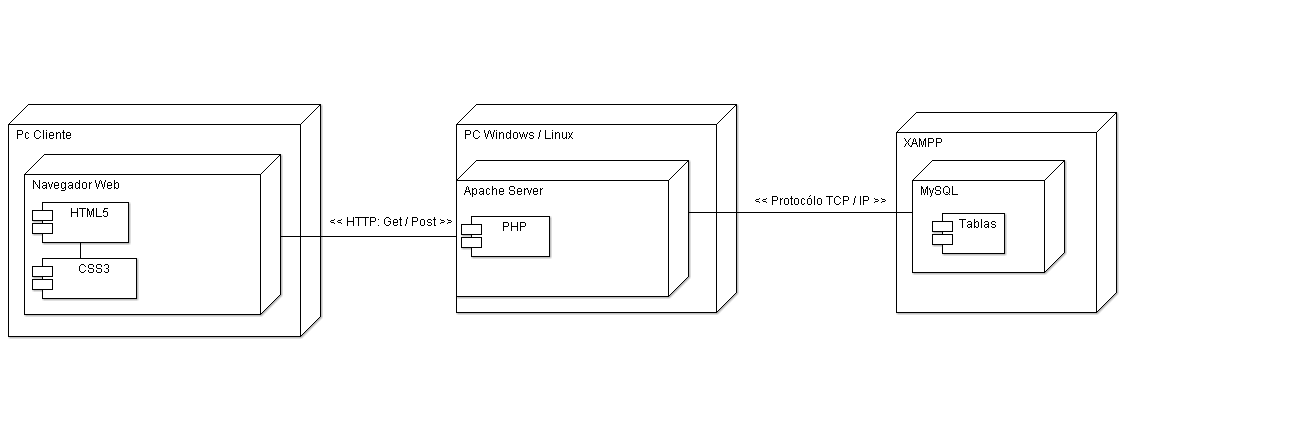


Fig. 5 Diagrama de despliegue

# Tamaño y rendimiento

* El sistema de reservación de habitaciones de un hotel permitirá el acceso concurrente de varios usuarios, de modo que puedan realizar transacciones simultáneas.
* La plataforma del sistema operativo puede ser Windows o Linux.
* La interfaz de usuario del sistema de reservación de habitaciones deberá ser gráfica y orientada a ventanas, y su manejo se realizará por medio de teclado y ratón. Deberá además ser diseñada teniendo siempre en cuenta la facilidad de uso y el hecho de que está orientada a una comunidad de usuarios con conocimientos básicos en computación.