

# Arquitectura Alissta

## Sobre el documento

El presente documento contiene el diseño elaborado para el proyecto POSITIVA SG-SST, 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.

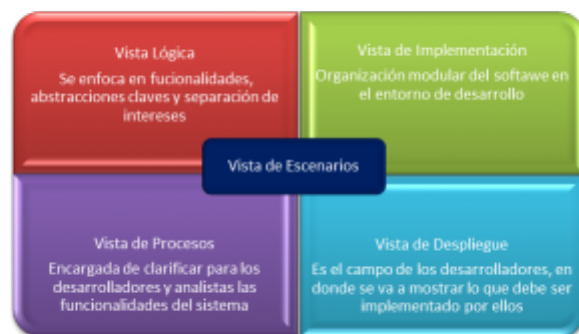
Al contrario de muchas otras actividades técnicas, el desarrollo de sistemas intensivos en software dedica la mayoría de sus esfuerzos a la especificación y modelado. Los modelos son utilizados tanto para el análisis de requisitos, como para el diseño de la solución, así como para la especificación, construcción y despliegue del sistema en su ambiente de explotación. Los modelos son presentados por medio de 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 final 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.

## Descripción General

Se entiende por arquitectura del software, como el 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. Esta sección describe en términos generales, las ideas principales detrás de la arquitectura escogida para el mismo.

El diseño será representado por medio del modelo de "4+1 Vistas" (Kruchten, vol. 12), cuyo objetivo es mostrar, en cada una de las vistas, una perspectiva o visión de un conjunto de elementos del proyecto y sus relaciones, esto desde el punto de vista de la arquitectura. En unión, las 4+1 vistas representan las decisiones de diseño y la forma como se desarrollará el proyecto. Este modelo posee un alto grado de importancia debido que está estrechamente relacionado con todos los Stakeholders según su rol dentro del desarrollo del proyecto. El modelo se divide en 4+1 vistas que se describen en la Figura 1.



El modelo de vistas múltiples, organiza una descripción de la arquitectura de software utilizando cinco vistas concurrentes, las cuales permiten aproximar de manera aislada los intereses de los diferentes stakeholders de la arquitectura: los usuarios finales, los desarrolladores, entre otros; y manejar de manera separada los requerimientos funcionales y no funcionales. Se capturan las decisiones de diseño en cuatro de las vistas y utilizan la quinta vista para ilustrar y validarlas.

El modelo propone las siguientes perspectivas o vistas:

1. **Vista lógica:** Ofrece soporte a los requerimientos funcionales, lo que el sistema debe proveer en términos de servicios a sus usuarios.
2. **Vista de procesos:** La vista de procesos permite describir los procesos del sistema y como estos se comunican.
3. **Vista física o de despliegue:** La vista física describe como es instalada la aplicación y como se ejecuta en una red de computadores. Para describir esta vista, en el presente documento se utilizó un diagrama de despliegue.
4. **Vista de desarrollo o de implementación:** Esta vista se concentra en la organización en módulos del software. En el presente documento, fue representada por el diagrama de paquetes.
5. **Vista de casos de uso:** La vista de casos de uso consolida las vistas anteriores, donde los escenarios se convierten en una abstracción de los requerimientos más importantes. Para describir esta vista, en el presente documento como anexo se encuentran los casos de uso.

Una vez explicadas las vistas que propone Kruchten y la forma de documentarlas, se puede apreciar que es un modelo bastante bueno para documentar la arquitectura de un sistema software “complejo”, ya que todos los stakeholders pueden entender el sistema que se está desarrollando desde diferentes perspectivas.

## Posicionamiento y Alcance

Positiva SG-SST Es un sistema web que permite a las empresas asociadas a POSITIVA ARL gestionar, administrar y aplicar la norma 1072 de 2015, la resolución 0312 del 2019 y la 1401 del 2007 la cual exige a las empresas implementar un Sistema de Gestión de la Seguridad y Salud en el Trabajo (SG-SST) El sistema web se Construirá y será soportado por las tecnologías Microsoft visual estudio 2013 últimate con una versión Microsoft .Net Framework 4.6.01586, con un motor de Base de Datos SQL SERVER 2014 management Studio respectivamente.

## Objetivos del Diseño

A continuación se describen los objetivos de acuerdo al tipo de tecnología en la cual se soportan los módulos:

- Definir la arquitectura soportada en el sistema. - Establecer lineamientos para el uso de los recursos utilizados en el desarrollo de los componentes del sistema. - Definir lineamientos para la creación del modelo de datos que soportara el sistema.

## Características Principales

SG-SST es un sistema que tiene las siguientes características principales:

• Diseño basado en módulos. • Construcción y desarrollo bajo el Paradigma POO. • Estructura de datos relacional. • Ambiente cliente/servidor. • Patrón de Diseño MVC. • Integración con sistemas operativos para dispositivos móviles Android, IOS. • Fácil de utilizar e interfaz amigable con el usuario y con la tecnología Responsive Design.

## Restricciones y Limitaciones

De acuerdo a las tecnologías utilizadas en el desarrollo del sistema se presentan las siguientes restricciones y limitaciones:

- La solución de Software debe ser una sola, no se permite integración con otro desarrollo u otras aplicaciones.
- Debe de estar construida con lenguajes de programación orientados a objetos (.Net).
- La solución debe de estar construida con un motor de Base de datos SQL server 2014 o superior.
- El desarrollo para móviles debe soportar sistemas operativos Android y iOS.
- La solución contratada debe estar construida en ambiente web.
- La solución debe estar construida en capas: Presentación, Negocio y Persistencia (datos).
- La solución debe funcionar sobre los siguientes Web Browser, como mínimo: Chrome 41.x o superior.
- La solución (hardware, software, conectividad) debe tener representación y soporte tecnológico y funcional dentro del territorio colombiano, esencialmente en la ciudad de Bogotá.
- El idioma del sistema de información debe ser español (adaptado a la terminología colombiana).
- Se debe garantizar un Data-Center de Tier-3 mínimo. La empresa ADA realizará esta gestión por el periodo de un año (Renovable al término de este).
- El sistema debe utilizar el protocolo de seguridad HTTPS (Hipertexto Transfer Protocol Secure)

- El sitio web deberá contar con un certificado SSL con validación extendida por 3 años a nombre del dominio de Positiva Compañía de Seguros S.A.

## Requisitos Atendidos

La motivación y el fundamento de todo lo hecho en el proyecto, no son otros sino los requisitos y necesidades, tanto del cliente como de los futuros usuarios del sistema.

Es por esto, que en esta sección se indican los requisitos atendidos por el diseño o arquitectura que se describirá en las próximas secciones.

## Requisitos Suplementarios

- La aplicación app debe estar sincronizada con la versión web en todo momento.
- Arquitectura en 3 capas: presentación, negocio y persistencia (datos).
- Arquitectura escalable vertical y horizontalmente.
- El sistema debe tener mecanismos de integración a otros sistemas de información tanto internos como de terceros utilizando el modelo y plataforma de integración SOAP y REST de la compañía.
- El sistema debe garantizar la integridad en los datos y las operaciones sobre los mismos.
- El sistema debe tener mecanismos de captura y manejo de errores en componentes, manejo de excepcionales.
- El sistema debe integrar ayudas en línea para cada uno de los módulos del sistema en español.
- El sistema debe notificar al usuario sobre todos los procesos que se ejecuten de manera exitosa o defectuosa.
- El sistema debe asegurar tiempos de respuesta de máximo 2 segundos por transacción.
- El sistema debe permitir cancelar procesos ejecutados en cualquier momento.
- El sistema debe asegurar la consistencia, integridad y disponibilidad de la información procesada.
- El sistema debe ser complementario, compatible e interoperable con las diferentes Plataformas tecnológicas con que cuenta la compañía para el soporte de los procesos de promoción y prevención.
- El sistema debe contener ayudas en línea, en donde se implementen los manuales de usuario y de procedimientos que competen al manejo de esta información.
- Las ayudas se refieren a cómo utilizar cada módulo del sistema cuando es utilizado por el usuario, por ejemplo ingresar información, hacer consultas, como sacar reportes, editar un ítem. Además se tendrá un manual de usuario que explicará de manera general como utilizar el sistema.
- El sistema debe soportar volúmenes de 1000 empresas con promedio de 2 usuarios por empresa para el primer año.
- El sistema debe contar con una disponibilidad de mínimo 99.5% de las veces en que un usuario intente accederlo.
- El sistema debe garantizar el ingreso a través de diferentes perfiles de acceso.
- El sistema debe limitar las opciones de menú y submenú de cada uno de los usuarios de acuerdo al perfil.
- El sistema debe permitir realizar la recuperación de contraseñas (correo electrónico, mesa de ayuda y PIN).

## Vistas y Planos

Los sistemas intensivos en software, se encuentran formados por un conjunto de componentes, que no son más que los elementos listos para ser ejecutados producidos por el proyecto.

Dichos componentes se distribuyen sobre los distintos equipos según lo que se detalla en la vista de despliegue.

Por otro lado, la vista lógica presenta al sistema como un todo, indicando en términos propios de la tecnología utilizada, las partes que lo forman y las relaciones principales entre ellas.

Finalmente se presenta el llamado modelo de datos, que contiene la estructura de almacenamiento de información requerida por el sistema aquí descrito.

## Resumen de Vistas

| Vista                      | Contiene                | Se relaciona con                                                                                   | Comentarios                                                        |
|----------------------------|-------------------------|----------------------------------------------------------------------------------------------------|--------------------------------------------------------------------|
| <b>Vista Física</b>        | Diagrama de Despliegue  | Tiene relación con las interfaces de comunicación del sistema, y artefactos que lo componen.       | Contiene los nodos que forman la arquitectura (Tipología).         |
| <b>Vista de Procesos</b>   | Diagrama de Secuencia   | Tiene Relación con las Clases, Objetos Modelo que componen la aplicación.                          | Muestra la interacción de los diferentes elementos del sistema.    |
| <b>Vista Lógica</b>        | Diagrama de Componentes | Tiene relación con la aplicación en y con el funcionamiento general de los componentes integrados. | Muestra el alcance de las funcionalidades que componen el sistema. |
| <b>Vista De Desarrollo</b> | Diagrama de Paquetes    | Tiene relación entre los componentes del producto y como se comunican entre si                     | Muestra la relación de los componentes del sistema.                |

From:

<http://wiki.adacsc.co/> - Wiki

Permanent link:

[http://wiki.adacsc.co/doku.php?id=ada:arquitectura\\_alissta&rev=1642725412](http://wiki.adacsc.co/doku.php?id=ada:arquitectura_alissta&rev=1642725412)

Last update: 2022/01/21 00:36

