



Instructivo para la elaboración de la arquitectura de software

Proceso
Gestión de Servicios de Información
y Soporte Tecnológico
Versión 1
16/06/2023

MINISTERIO DE AMBIENTE Y DESARROLLO SOSTENIBLE	INSTRUCTIVO PARA LA ELABORACIÓN DE ARQUITECTURAS DE SOFTWARE	SOMOSIG Sistema Integrado de Gestión
	Proceso: Gestión de Servicios de Información y Soporte Tecnológico	
Versión: 1	Vigencia: 16/06/2023	Código: I-A-GTI-03

TABLA DE CONTENIDO

1.	OBJETIVO DEL INSTRUCTIVO	4
2.	INTRODUCCIÓN AL INSTRUCTIVO.....	4
3.	PLANTILLA PARA LA DOCUMENTACIÓN DE ARQUITECTURAS DE SOFTWARE	6
3.1.	Objetivo.....	7
3.2.	Metas y Restricciones de la Arquitectura de Software	7
3.3.	Qué No Contempla La Arquitectura de Software	8
3.4.	Atributos de Calidad.....	8
3.5.	Diagrama de Contexto (Nivel 1).....	10
3.6.	Diagrama de Contenedores (Nivel 2).....	11
3.7.	Diagrama de Componentes (Nivel 3).....	13
3.8.	Diagramas de Código (Nivel 4).....	14
3.8.1.	Diagrama de Clases	15
3.8.2.	Diagrama Modelo De Datos.....	18
4.	TÉRMINOS Y/O CONCEPTOS DEL INSTRUCTIVO	19
5.	REFERENCIAS DEL INSTRUCTIVO	19



MINISTERIO DE AMBIENTE Y DESARROLLO SOSTENIBLE	INSTRUCTIVO PARA LA ELABORACIÓN DE ARQUITECTURAS DE SOFTWARE	SOMOSIG Sistema Integrado de Gestión
	Proceso: Gestión de Servicios de Información y Soporte Tecnológico	
Versión: 1	Vigencia: 16/06/2023	Código: I-A-GTI-03


LISTA DE TABLAS

Tabla 1.	Instrumento de validación Diagrama de Contexto (Nivel 1)	10
Tabla 2.	Instrumento de validación Diagrama de Contenedores (Nivel 2)	12
Tabla 3.	Información de Contenedor.....	12
Tabla 4.	Instrumento de validación Diagrama de Componentes (Nivel 3)	13
Tabla 5.	Estructura de una clase	15
Tabla 6.	Estructura de Interface.....	17

LISTA DE IMÁGENES

Imagen 1.	Elementos Modelo C4.....	4
Imagen 2.	Elementos Arquitectura C4	5
Imagen 3.	Resumen del Modelo C4.....	6
Imagen 4.	Diagrama de Contexto - Sistema VITAL Propósito Ejemplo	11
Imagen 5.	Diagrama de Contenedores - Integración Ecosistema VITAL - Propósito Ejemplo.....	12
Imagen 6.	Diagrama de Componentes - Integración Vital - Propósito Ejemplo	14
Imagen 7.	Imagen 8 Diagrama de Clases Ecosistema VITAL - Propósito Ejemplo	17
Imagen 8.	Modelo de Datos - Ecosistema VITAL - Propósito Ejemplo	18



MINISTERIO DE AMBIENTE Y DESARROLLO SOSTENIBLE	INSTRUCTIVO PARA LA ELABORACIÓN DE ARQUITECTURAS DE SOFTWARE	
	Proceso: Gestión de Servicios de Información y Soporte Tecnológico	
Versión: 1	Vigencia: 16/06/2023	Código: I-A-GTI-03

1. OBJETIVO DEL INSTRUCTIVO

Dar lineamientos a los procesos técnicos y las especificaciones de diseño que requiere el Arquitecto de Software en la implementación de soluciones tecnológicas desarrolladas por la Oficina de Tecnologías de la Información y la Comunicación y otros terceros para el Ministerio de Ambiente y Desarrollo Sostenible.

2. INTRODUCCIÓN AL INSTRUCTIVO

Este documento es el referente para la elaboración y construcción de la Arquitectura de Software de los diferentes Sistemas de Información, Nuevos Módulos o Artefactos tecnológicos que defina la Oficina de Tecnologías de la Información y la Comunicación del Ministerio de Ambiente y Desarrollo Sostenible.

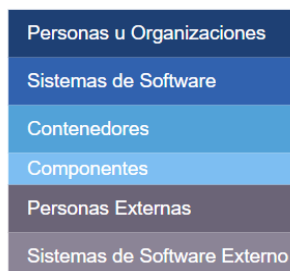
La arquitectura de software está a cargo del Arquitecto de Software y debe cumplir con las actividades de diseñar o actualizar la arquitectura de software de sistemas de Información, definidos por la Oficina de Tecnologías y Sistemas de la Información del Ministerio de Ambiente y Desarrollo Sostenible descritas en el procedimiento "P-A-GTI-03 Desarrollo y mantenimiento de sistemas de información y componentes de software".

La arquitectura de software está basada en el Modelo C4 de visualización de Arquitectura de Software para los diferentes proyectos que se desarrollen para el Ministerio y así optimizar recursos, flujos de proceso y desarrollo.

El Modelo C4 es una forma sencilla y eficaz de representar la arquitectura de software en diferentes niveles de abstracción. Consta de cuatro niveles: contexto del sistema, contenedores, componentes y código (Brown, 2013). El modelo C4 aborda las limitaciones de los enfoques de modelado existentes, proporcionando una notación estandarizada que es simple, intuitiva y lo suficientemente flexible como para representar sistemas de software de diferentes tamaños y complejidad.

Imagen 1. Elementos Modelo C4

Elementos Modelo C4




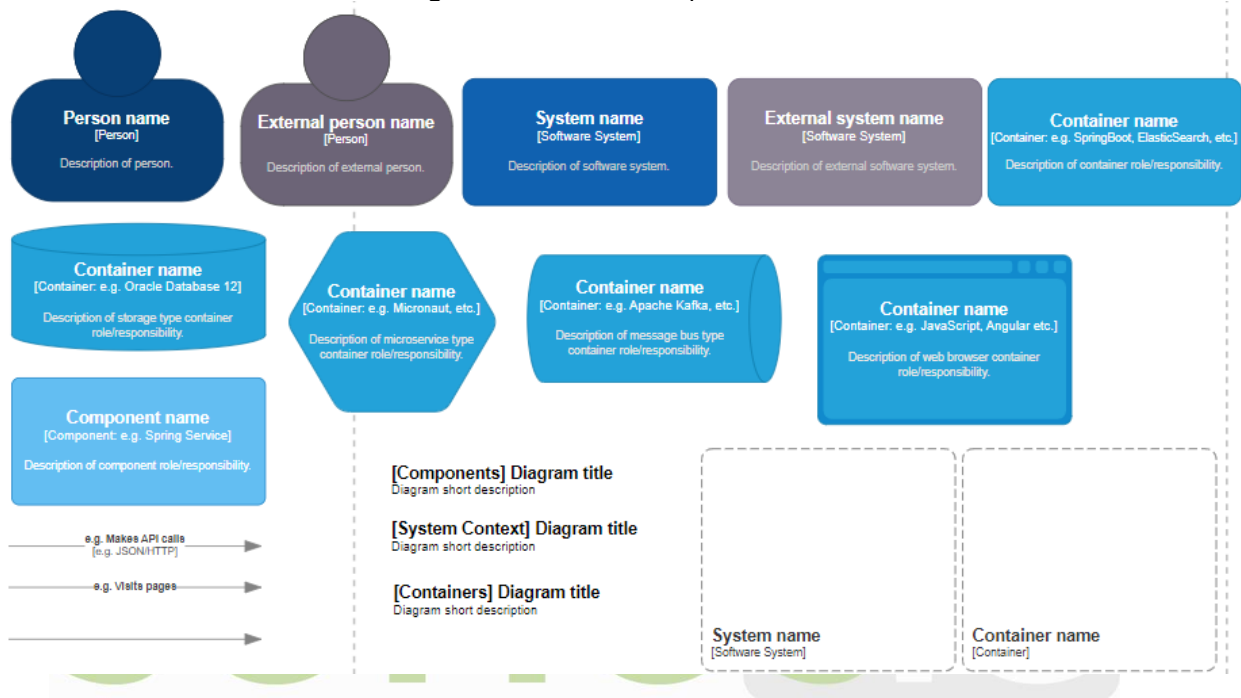
MINISTERIO DE AMBIENTE Y DESARROLLO SOSTENIBLE	INSTRUCTIVO PARA LA ELABORACIÓN DE ARQUITECTURAS DE SOFTWARE	 Sistema Integrado de Gestión
	Proceso: Gestión de Servicios de Información y Soporte Tecnológico	
Versión: 1	Vigencia: 16/06/2023	Código: I-A-GTI-03


Imagen 2. Elementos Arquitectura C4



El énfasis del modelo C4 en la simplicidad y la flexibilidad facilita la comprensión y la comunicación de la arquitectura de software a las partes interesadas. Proporciona una forma coherente de representar los sistemas de software que pueden utilizar todas las partes interesadas, desde los desarrolladores hasta los jefes de proyecto (Brown, 2013).

El modelo C4 ofrece una sintaxis que podría hacer de la conceptualización y abstracción de la funcionalidad de un sistema una tarea más fácil y menos complicada. Se basa en una jerarquía de diagramas, la cual define los sistemas de software a través de cuatro puntos de vista (niveles). Cada uno de estos niveles del modelo C4 se centra en una determinada perspectiva del sistema. Cada diagrama del modelo de arquitectura C4 utiliza un conjunto estándar de símbolos y notación para representar los distintos componentes y relaciones dentro del sistema. Por ejemplo, se usan rectángulos para representar contenedores, círculos para representar componentes y flechas para mostrar las relaciones entre ellos.

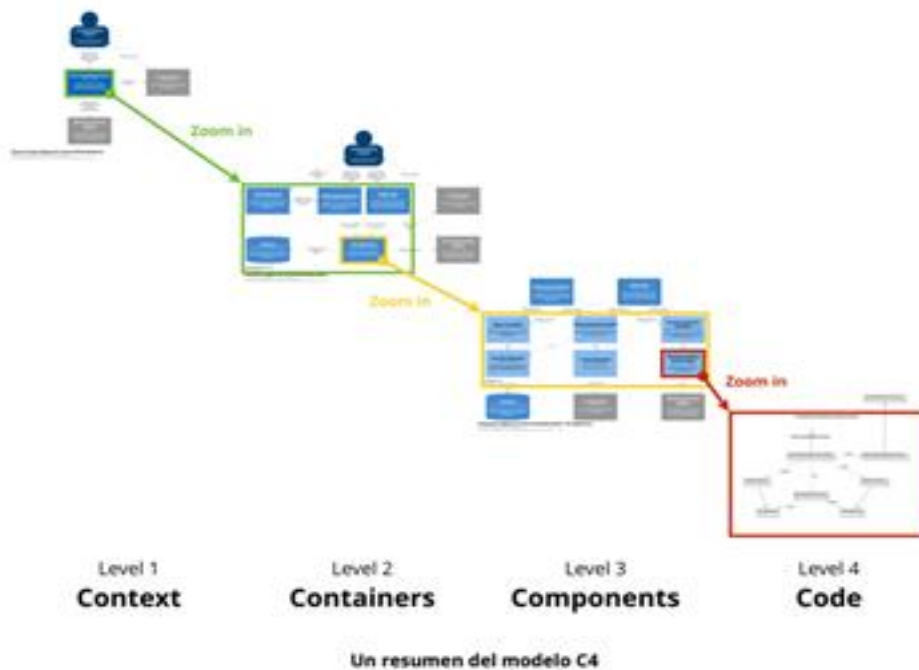
El primer nivel del diagrama representa el contexto en el que el sistema se va a emplear, por ejemplo, los usuarios, sistemas externos y factores ambientales que influyen en su diseño. El segundo nivel del diagrama muestra los contenedores que forman parte del sistema, por ejemplo, las aplicaciones web, las bases de datos y los servidores de aplicaciones. Cada contenedor se representa mediante un rectángulo y se etiqueta con su nombre y tecnología subyacente.

MINISTERIO DE AMBIENTE Y DESARROLLO SOSTENIBLE	INSTRUCTIVO PARA LA ELABORACIÓN DE ARQUITECTURAS DE SOFTWARE	
	Proceso: Gestión de Servicios de Información y Soporte Tecnológico	
Versión: 1	Vigencia: 16/06/2023	Código: I-A-GTI-03

El tercer nivel del diagrama muestra los componentes que conforman cada contenedor. Los componentes se representan mediante círculos y se agrupan dentro del contenedor correspondiente. Cada componente se etiqueta con su nombre y responsabilidad principal. Además, se muestra la relación entre los distintos componentes mediante líneas que indican las dependencias y comunicaciones entre ellos.

Por último, el cuarto nivel del diagrama representa el código que implementa cada componente. Se puede mostrar el código fuente o un subconjunto de clases o módulos relevantes. Cada nivel del diagrama descrito se utiliza para enfocar y detallar aspectos específicos del sistema, proporcionando una vista diferente y útil para distintos roles o audiencias. Además, estos diagramas pueden ayudar a identificar oportunidades de mejora en el diseño y a comunicar de manera efectiva la arquitectura del sistema a los interesados.

Imagen 3. Resumen del Modelo C4



3. PLANTILLA PARA LA DOCUMENTACIÓN DE ARQUITECTURAS DE SOFTWARE

La Arquitectura de Software debe ser documentada en la wiki del Ministerio de Ambiente y Desarrollo Sostenible haciendo uso de la plantilla definida por la Oficina de Tecnologías de la Información y la Comunicación para tal fin, la cual tiene el siguiente contenido:

MINISTERIO DE AMBIENTE Y DESARROLLO SOSTENIBLE	INSTRUCTIVO PARA LA ELABORACIÓN DE ARQUITECTURAS DE SOFTWARE	SOMOSIG Sistema Integrado de Gestión
	Proceso: Gestión de Servicios de Información y Soporte Tecnológico	
Versión: 1	Vigencia: 16/06/2023	Código: I-A-GTI-03

- Objetivo
- Metas y restricciones de la Arquitectura de Software
- Qué no contempla la arquitectura de software
- Atributos de calidad
- Diagrama de contexto (Nivel 1)
- Diagrama de contenedores (Nivel 2)
- Diagrama de componentes (Nivel 3)
- Diagramas de código (Nivel 4)
 - Diagrama de clases
 - Diagrama modelo de datos

A continuación, se describe el contenido y criterios de aceptación de cada uno de los componentes de la Arquitectura de Software.

3.1. Objetivo

En esta sección se debe describir el propósito de la Arquitectura de Software y proporcionar una breve descripción del sistema de información o componente de software que se está documentando.

Ej.:

El propósito de este documento es proporcionar la información técnica y componentes del sistema de información **NOMBRE DEL SISTEMA DE INFORMACIÓN O COMPONENTE DE SOFTWARE**. En este documento, se proporcionan las vistas del sistema basadas en un modelo C4.


El sistema de información **NOMBRE DEL SISTEMA DE INFORMACIÓN O COMPONENTE DE SOFTWARE** es una solución tecnológica para describir las principales funcionalidades del sistema de información o componente de software.

3.2. Metas y Restricciones de la Arquitectura de Software

En esta sección se deben describir las metas y restricciones de la Arquitectura de Software

Ej.:



MINISTERIO DE AMBIENTE Y DESARROLLO SOSTENIBLE	INSTRUCTIVO PARA LA ELABORACIÓN DE ARQUITECTURAS DE SOFTWARE	 Sistema Integrado de Gestión
	Proceso: Gestión de Servicios de Información y Soporte Tecnológico	
Versión: 1	Vigencia: 16/06/2023	Código: I-A-GTI-03

Es competencia de la arquitectura que se define para el proyecto de desarrollo cumplir con las siguientes metas:

1. Se deberán seguir los estándares de diseño definidos por la Oficina de Tecnologías de la Información y la Comunicación.
2. Las funcionalidades por desarrollar deberán instalarse en la Infraestructura definida por la Oficina de Tecnologías de la Información y la Comunicación.
3. El acceso a las bases de datos se realizará a través de una capa intermedia que provea las interfaces necesarias para gestionar los datos.
4. El proceso de interoperabilidad se debe dar vía APIs “Web Services” al igual que toda la lógica de negocio que este atado al proceso.
5. El desarrollo debe contemplar componentes en distintas arquitecturas, separando los componentes de Frontend y de Backend con APIs de acceso a negocio y datos.
6. Se debe respetar y cumplir la Arquitectura de Referencia definida por la Oficina de Tecnologías de la Información y la Comunicación para la ejecución del proyecto.

3.3. Qué No Contempla La Arquitectura de Software

En esta sección se deben describir las excepciones de la Arquitectura de Software

Ej.:

- La disponibilidad de la información dependerá también de la disponibilidad de las conexiones tanto de bases de datos, servidor web y/o conexión a internet/intranet u otras conexiones.
- Los proyectos deben ejecutarse en infraestructura definida por el arquitecto de software y será él quien establezca donde se despliegue el proyecto y este será documentado en el manual de despliegue.
- La disponibilidad de las Bases de Datos en el entorno Legacy estarán sujetas a los canales de internet que tenga el cliente, proveedor o Ministerio de Ambiente y Desarrollo Sostenible. Esto se debe definir en los acuerdos de niveles de servicio de acuerdo con el tipo de conexión.


3.4. Atributos de Calidad

En esta sección se deben describir los atributos de calidad que se deben tener en cuenta en la Arquitectura del Software.

Ej.:


- **Seguridad:** La seguridad también está en los distintos componentes **y se deben contemplar y garantizar que cumplan con las siguientes características:**



MINISTERIO DE AMBIENTE Y DESARROLLO SOSTENIBLE	INSTRUCTIVO PARA LA ELABORACIÓN DE ARQUITECTURAS DE SOFTWARE	
	Proceso: Gestión de Servicios de Información y Soporte Tecnológico	
Versión: 1	Vigencia: 16/06/2023	Código: I-A-GTI-03

- Diseño de Arquitectura segura y evaluación amenazas
 - Autenticación Segura
 - Gestión de sesiones
 - Control de acceso
 - Validación de Arquitectura y codificación
 - Criptografía almacenada
 - Gestión y registro de errores
 - Protección de datos
 - Comunicación segura y cifrado fuerte
 - Código malicioso
 - Lógica de negocios secuencia y segura
 - Archivos y recursos con accesos limitados
 - Servicio web y API
 - Configuración segura para el uso de la aplicación
- **Escalabilidad:** El desarrollo de Sistemas de Información, nuevos Módulos o Artefactos tecnológicos deben permitir la escalabilidad tanto horizontal como vertical, para un rendimiento óptimo y se deben cumplir y seguir las recomendaciones dadas en el manual de despliegue.
 - **Flexibilidad:** El patrón de arquitectura de capas, permite una alta cohesión y bajo acoplamiento entre los componentes, esto acompañado de buenas prácticas de diseño y desarrollo de software.
 - **Disponibilidad:** La arquitectura de software debe ser contemplada con una alta disponibilidad en ambientes productivos y los Sistemas de Información, nuevos Módulos o Artefactos tecnológicos tendrán una disponibilidad 24/7, esto debe ir acompañado de los acuerdos de nivel de servicios (ANS) que se deben definir con el área de infraestructura y de acuerdo con el crecimiento de usuarios concurrentes del sistema.
 - **Rendimiento:** Se debe garantizar el rendimiento de los Sistemas de Información, Nuevos Módulos o Artefactos tecnológicos, de manera adecuada y de acuerdo con la óptima configuración, esto quiere decir que las herramientas, librerías, componentes, servidores de aplicación utilizados en la Arquitectura de Software, deberán ser ajustadas reescribiendo las configuraciones por defecto de los fabricantes para lograr el óptimo desempeño de acuerdo a las características de la Arquitectura de Software; sistemas de cache para optimizar el acceso a los recursos e información que tenga los componentes, optimización de los objetos de base de datos de acuerdo a sus planes de ejecución creando índices u otras optimizaciones en el acceso a ellos.
 - **Observabilidad:** El desarrollo de Sistemas de Información, nuevos Módulos o Artefactos tecnológicos debe tener una vista conectada de todos los datos de rendimiento del sistema, en tiempo real, de esta forma detectar los problemas de forma rápida, entender la causa del problema y así generar buena experiencia al usuario final, los cuales deberán proveer logs de aplicación que evidencias malos funcionamiento, errores o cualquiera información que la aplicación provea y sea de interés para su análisis.



MINISTERIO DE AMBIENTE Y DESARROLLO SOSTENIBLE	INSTRUCTIVO PARA LA ELABORACIÓN DE ARQUITECTURAS DE SOFTWARE	 Sistema Integrado de Gestión
	Proceso: Gestión de Servicios de Información y Soporte Tecnológico	
Versión: 1	Vigencia: 16/06/2023	Código: I-A-GTI-03

3.5. Diagrama de Contexto (Nivel 1)

En esta sección se debe documentar y describir el diagrama de contexto (Nivel 1) de la Arquitectura del Software.

En los desarrollos realizados para la Oficina de Tecnologías de la Información un diagrama de contexto del sistema es el punto de partida para diagramar y documentar un sistema de información, artefacto o módulo y poder describir un panorama general y real de este.

El primer nivel es el diagrama de contexto, que muestra el sistema y sus dependencias externas. Se centra en enmarcar el sistema software a modelar mediante la representación de las diferentes personas o actores involucrados y los sistemas externos que proporcionan algún tipo de servicio al sistema. Las interacciones entre componentes se representan mediante líneas de flechas discontinuas anotadas por una descripción con el papel de la relación. Los servicios externos o ya implementados se colorean diferentes del sistema de software de destino. En este nivel, se omiten otros detalles relativos a los sistemas de software implicados, proporcionando una visión de alto nivel del contexto en el que se enmarcará el sistema objetivo.

El diagrama de Contexto debe incluir Personas y Sistemas de Información con los cuales interactúa el Sistema en desarrollos internos y externos del Ministerio de Ambiente y Desarrollo Sostenible.

De acuerdo con la metodología C4 el diagrama de contexto del sistema debe responder a las siguientes preguntas para validar su correcta construcción:

Tabla 1. Instrumento de validación Diagrama de Contexto (Nivel 1)

1. ¿Cuál es el sistema de software que se está construyendo (o se ha construido)?	Respuesta:
2. ¿Quién lo está usando?	Respuesta:
3. ¿Cómo encaja con el entorno existente?	Respuesta:

En el siguiente ejemplo se presenta un Diagrama de Contexto del Sistema VITAL con sus usuarios y otros sistemas con los que interactúa. Se debe generar una vista amplia, sin muchos detalles técnicos específicos, pero sí debe tener un enfoque importante en las personas (usuarios, roles, actores) y otros sistemas con los que interactúa.


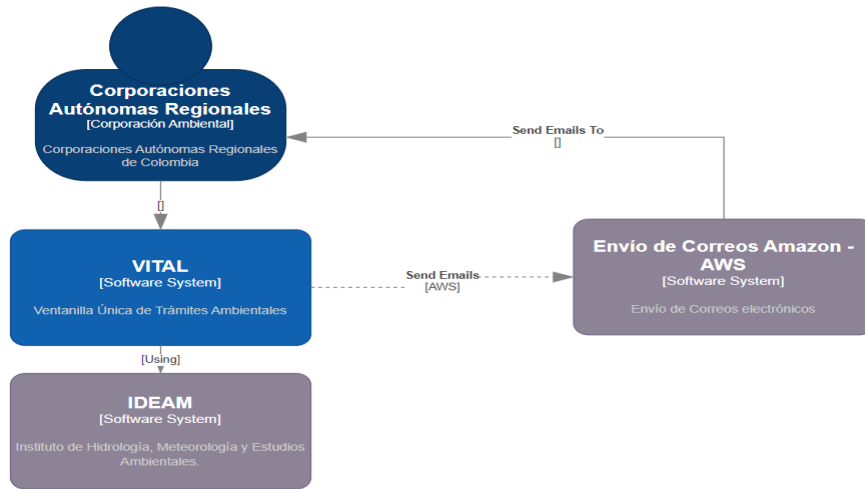
MINISTERIO DE AMBIENTE Y DESARROLLO SOSTENIBLE	INSTRUCTIVO PARA LA ELABORACIÓN DE ARQUITECTURAS DE SOFTWARE	 Sistema Integrado de Gestión
	Proceso: Gestión de Servicios de Información y Soporte Tecnológico	
Versión: 1	Vigencia: 16/06/2023	Código: I-A-GTI-03

Imagen 4. Diagrama de Contexto - Sistema VITAL Propósito Ejemplo



3.6. Diagrama de Contenedores (Nivel 2)

En esta sección se debe documentar y describir el diagrama de contenedores (Nivel 2) de la Arquitectura del Software.

El segundo nivel es el diagrama de contenedores. Un contenedor representa algo en lo que se ejecutan los componentes o donde residen los datos y que necesita ejecutarse para que el sistema de software funcione. Muestra los componentes de alto nivel del sistema, pueden ser físicos, como los servidores físicos o virtuales o contenedores Docker; esta vista esboza los componentes necesarios para proporcionar los servicios que ofrecería el sistema, como el servidor web, el servidor de bases de datos, el servidor de aplicaciones, y cómo interactúan entre sí.

Los contenedores suelen ser ejecutables que se inician como parte del sistema general, pero no tienen por qué ser procesos independientes por derecho propio. Por ejemplo, cada aplicación web Java EE o cada sitio web .NET es un contenedor independiente, indistintamente de si se ejecutan en el mismo proceso físico del servidor web. La clave para entender un sistema de software desde la perspectiva de los contenedores es que cualquier comunicación entre contenedores requerirá probablemente una interfaz remota, como un servicio web SOAP, una interfaz RESTfull, Java RMI, Microsoft WCF, mensajería, entre otros.

En el diseño de un diagrama de Contenedor se pueden incluir los detalles técnicos que más relevancia tengan del Sistema a desarrollar ejemplo: tecnologías, puertos, protocolos, etc., además se debe incluir Personas y Sistemas de información con los cuales interactúa el Sistema y contenedores.

De acuerdo con la Metodología C4 el Diagrama de Contenedor del sistema a desarrollar debe responder a las siguientes preguntas para validar su correcta construcción:

MINISTERIO DE AMBIENTE Y DESARROLLO SOSTENIBLE	INSTRUCTIVO PARA LA ELABORACIÓN DE ARQUITECTURAS DE SOFTWARE	SOMOSIG Sistema Integrado de Gestión
	Proceso: Gestión de Servicios de Información y Soporte Tecnológico	
Versión: 1	Vigencia: 16/06/2023	Código: I-A-GTI-03

Tabla 2. Instrumento de validación Diagrama de Contenedores (Nivel 2)

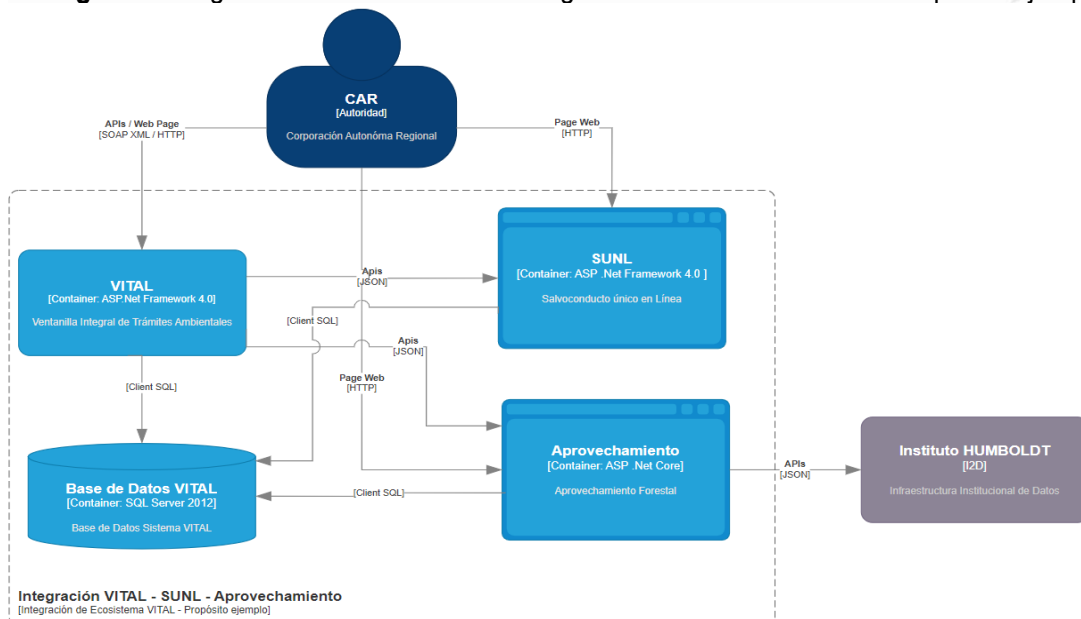
1. ¿Cuál es la forma general del sistema de software?	Respuesta:
2. ¿Cuáles son las decisiones tecnológicas de alto nivel?	Respuesta:
3. ¿Cómo se distribuyen las responsabilidades en todo el sistema?	Respuesta:
4. ¿Cómo se comunican los contenedores entre sí?	Respuesta:
5. Como desarrollador, ¿dónde necesito escribir código para implementar funciones?	Respuesta:


La información que se debe obtener de cada uno de los contenedores:

Tabla 3. Información de Contenedor

NOMBRE:	NOMBRE DEL CONTENEDOR
Tecnología:	Tecnología de Implementación (p. ej., Aplicación Python, Aplicación Web Django, Python Aplicación Fast Api, Aplicación React o Angular, esquema de base de datos relacional o Esquema de datos no relacional, etc.).
Descripción:	Una Descripción del Contenedor, lista de responsabilidades del contenedor o entidades/tablas/archivos/etc que se almacenan.

Imagen 5. Diagrama de Contenedores - Integración Ecosistema VITAL - Propósito Ejemplo



MINISTERIO DE AMBIENTE Y DESARROLLO SOSTENIBLE	INSTRUCTIVO PARA LA ELABORACIÓN DE ARQUITECTURAS DE SOFTWARE	 Sistema Integrado de Gestión
	Proceso: Gestión de Servicios de Información y Soporte Tecnológico	
Versión: 1	Vigencia: 16/06/2023	Código: I-A-GTI-03

3.7. Diagrama de Componentes (Nivel 3)

En esta sección se debe documentar y describir el diagrama de componentes (Nivel 3) de la Arquitectura del Software.

El tercer nivel es el Diagrama de Componentes. Los componentes son los bloques de construcción del sistema, puede considerarse una agrupación lógica de una o varias clases. Cada componente realiza una función específica y puede comunicarse con otros componentes a través de interfaces definidas. Este principio hace hincapié en la necesidad de modularizar y desacoplar los componentes para que el sistema sea más flexible y fácil de mantener.

Por ejemplo, un componente de auditoría o un servicio de autenticación que es utilizado por otros componentes para determinar si se permite el acceso a un recurso específico. Los componentes suelen estar formados por una serie de clases que colaboran entre sí, todas ellas detrás de un contrato de nivel superior. De nuevo, se omiten otros detalles como la estructura interna de los componentes, proporcionando únicamente una descripción funcional de cada contenedor y las relaciones entre ellos. Partiendo del diagrama de Contenedor que muestra los elementos tecnológicos de alto nivel, se debe realizar una descomposición de cada contenedor para mostrar los componentes que este contiene.

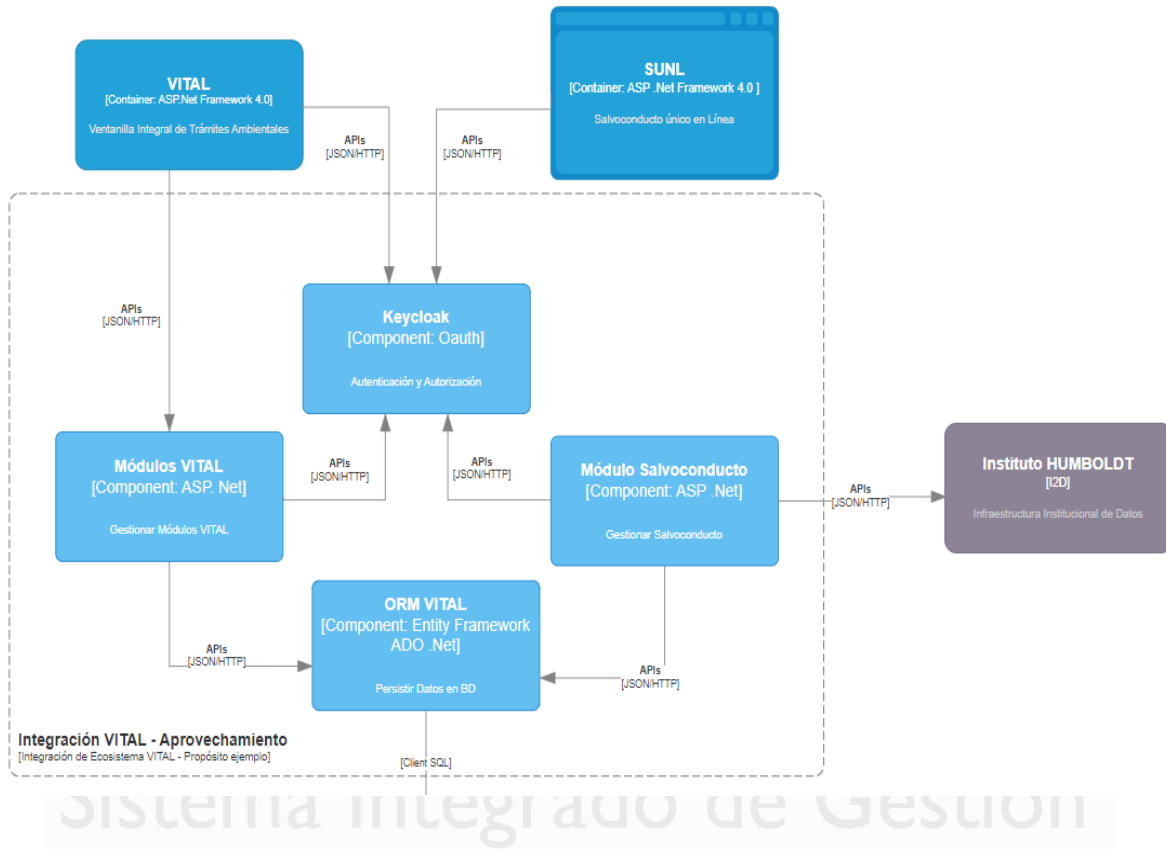
De acuerdo con la metodología C4 el Diagrama de Componentes del sistema debe responder a las siguientes preguntas para validar su correcta construcción:

Tabla 4. Instrumento de validación Diagrama de Componentes (Nivel 3)

1. ¿De qué componentes se compone cada contenedor?	Respuesta:
2. ¿Todos los componentes tienen un hogar (es decir, residen en un contenedor)?	Respuesta:
3. ¿Está claro cómo funciona el software a un alto nivel?	Respuesta:

MINISTERIO DE AMBIENTE Y DESARROLLO SOSTENIBLE	INSTRUCTIVO PARA LA ELABORACIÓN DE ARQUITECTURAS DE SOFTWARE	SOMOSIG Sistema Integrado de Gestión
	Proceso: Gestión de Servicios de Información y Soporte Tecnológico	
Versión: 1	Vigencia: 16/06/2023	Código: I-A-GTI-03

Imagen 6. Diagrama de Componentes - Integración Vital - Propósito Ejemplo




3.8. Diagramas de Código (Nivel 4)

En esta sección se deben documentar y describir los diagramas de código (Nivel 4) de la Arquitectura del Software, que contemplan:

- Diagrama de clases
- Diagrama del modelo de datos

El cuarto nivel es el diagrama del código. Representa el código que implementa cada componente. Este nivel proporciona detalles adicionales sobre cómo se ejecuta cada componente y cómo se conectan entre sí. Puede mostrar el código fuente o un subconjunto de clases o módulos relevantes. El diagrama de código ofrece una visión detallada de la implementación del sistema, incluidas las estructuras de código y su relación con la arquitectura de componentes.

En los Diagramas de nivel de Código se presentan los detalles de implementación de un componente individual, esto se debe realizar a través del diagrama de clases y Modelo de Datos.

MINISTERIO DE AMBIENTE Y DESARROLLO SOSTENIBLE	INSTRUCTIVO PARA LA ELABORACIÓN DE ARQUITECTURAS DE SOFTWARE	
	Proceso: Gestión de Servicios de Información y Soporte Tecnológico	
Versión: 1	Vigencia: 16/06/2023	Código: I-A-GTI-03

Los componentes pueden estar formados por una o varias clases o módulos, y representan una unidad cohesionada de funcionalidad que puede desarrollarse, probarse y desplegarse de forma independiente.

El código suele representarse como cuadros de texto o fragmentos de código dentro de los cuadros de componentes en los diagramas C4.

La distinción entre componentes y código es importante porque permite trabajar sobre la funcionalidad del sistema a diferentes niveles de abstracción. Los componentes proporcionan una visión de alto nivel de la funcionalidad del sistema, mientras que el código proporciona una visión de bajo nivel de cómo se implementa esa funcionalidad.

3.8.1. Diagrama de Clases

El diagrama de Clases se utiliza para representar desde un punto de vista estático los elementos de Sistema de Información.

El diagrama de clases es orientado a objetos, este define las clases que se van a crear y como es la relación entre ellas, con este diagrama se puede representar los datos y su interacción. Los elementos de un diagrama de clases son: Clases y relaciones o interfaces.

Clase:

Una clase representa conceptos o entidades del “negocio”, además, una clase define un grupo de objetos que comparten características, condiciones y significado.

La clase está compuesta por: nombre de la clase, atributos y métodos.


Tabla 5. Estructura de una clase

NOMBRE DE LA CLASE
- Atributos:
+ Métodos ():

- **Nombre de la clase:** En caso de que la clase sea abstracta se utilizará su nombre en cursiva.
- **Atributos de la Clase:** uno por línea y con el siguiente formato:
visibilidad nombre_atributo: tipo = valor-inicial
- **Métodos de una clase:** Funciones que ofrece la clase y con el siguiente formato:
visibilidad nombre_funcion {parametros}: tipo-devuelto

Los atributos y las funciones deben incluir que visibilidad van a tener:

- (+) Pública: Se puede acceder al atributo o función desde cualquier lugar de la aplicación

MINISTERIO DE AMBIENTE Y DESARROLLO SOSTENIBLE	INSTRUCTIVO PARA LA ELABORACIÓN DE ARQUITECTURAS DE SOFTWARE	
	Proceso: Gestión de Servicios de Información y Soporte Tecnológico	
Versión: 1	Vigencia: 16/06/2023	Código: I-A-GTI-03

- (-) Privada: Se puede acceder al atributo o función únicamente desde la misma clase.
- (#) Protegida: Se puede acceder al atributo o función únicamente desde la misma clase o las que hereden de ella.

Relaciones: Las relaciones son dependencias que hay entre las clases, esta se puede dar entre dos o más clases, estas se representan con líneas y varían de acuerdo con el tipo de relación. Las relaciones en el diagrama de clases tienen las siguientes propiedades:

Multiplicidad: El número de elementos de una clase que participan en una relación. Se puede indicar un número, un rango. Se utiliza n o * para identificar un número cualquiera.

Nombre de la Asociación: Indicación de la asociación que ayuda a entender la relación que tienen dos clases, suelen utilizarse verbos.

Tipos de relaciones: Un diagrama de clases incluye los siguientes tipos de relaciones:

- **Asociación:** Es el más común y se utiliza para representar dependencia semántica. Se representa con una simple línea continua que une las clases que están incluidas en la asociación.
- **Agregación:** Es una representación jerárquica que indica a un objeto y las partes que componen ese objeto. Es decir, representa relaciones en las que un objeto es parte de otro, pero aun así debe tener existencia en sí mismo. Se representa con una línea que tiene un rombo en la parte de la clase que es una agregación de la otra clase (es decir, en la clase que contiene las otras).
- **Composición:** Representa una relación jerárquica entre un objeto y las partes que lo componen, pero de una forma más fuerte. En este caso, los elementos que forman parte no tienen sentido de existencia cuando el primero no existe. Es decir, cuando el elemento que contiene los otros desaparece, deben desaparecer todos ya que no tienen sentido por sí mismos, sino que dependen del elemento que componen. Además, suelen tener los mismos tiempos de vida y los componentes no se comparten entre varios elementos.
- **Dependencia:** Este tipo de relación para representar que una clase requiere de otra para ofrecer sus funcionalidades. Es muy sencilla y se representa con una flecha discontinua que va desde la clase que necesita la utilidad de la otra flecha hasta esta misma.
- **Herencia:** Permiten que una clase (clase hija o subclase) reciba los atributos y métodos de otra clase (clase padre o superclase). Estos atributos y métodos recibidos se suman a los que la clase tiene por sí misma.

Interfaces: Una interfaz es una entidad que declara una serie de atributos, funciones y obligaciones. Es una especie de contrato donde toda instancia asociada a una interfaz debe de implementar los servicios que indica aquella interfaz.

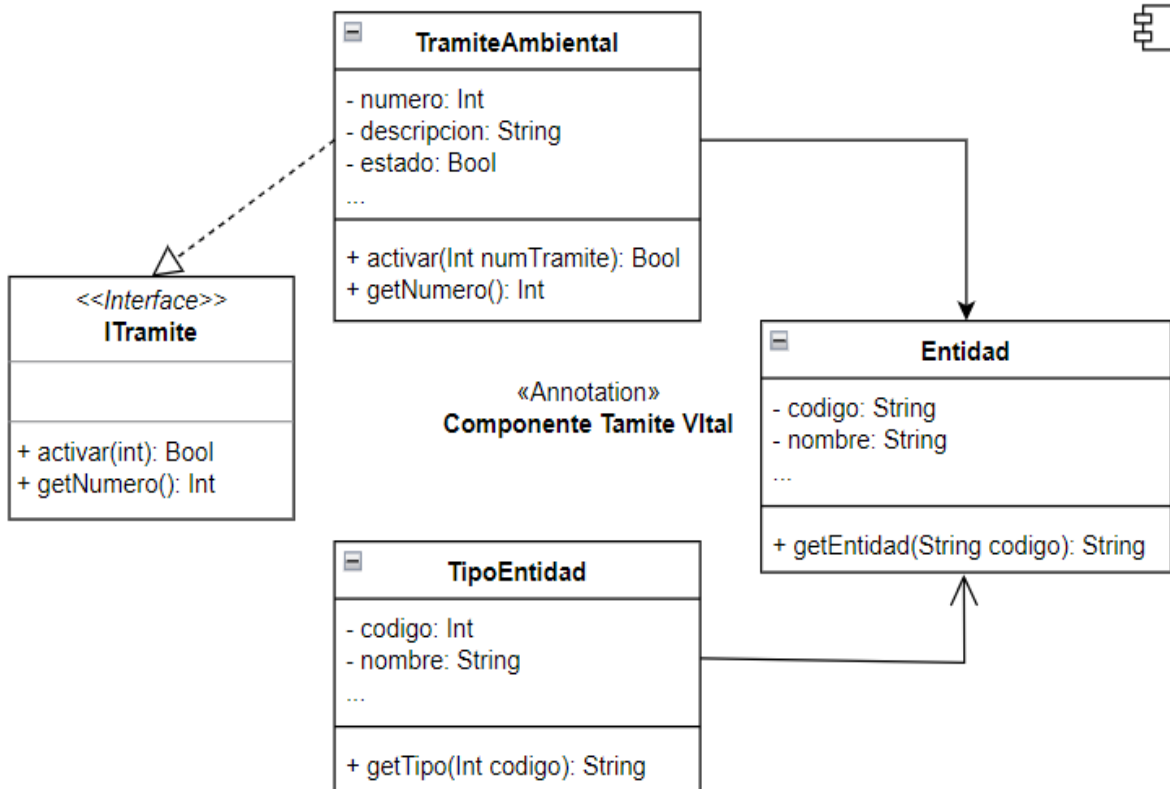
MINISTERIO DE AMBIENTE Y DESARROLLO SOSTENIBLE	INSTRUCTIVO PARA LA ELABORACIÓN DE ARQUITECTURAS DE SOFTWARE	SOMOSIG Sistema Integrado de Gestión
	Proceso: Gestión de Servicios de Información y Soporte Tecnológico	
Versión: 1	Vigencia: 16/06/2023	Código: I-A-GTI-03

Dado que únicamente son declaraciones no pueden ser instanciadas. Las interfaces se asocian a clases. Una asociación entre una clase y una interfaz representa que esa clase cumple con el contrato que indica la interfaz, es decir, incluye aquellas funciones y atributos que indica la interfaz. Su representación es similar a las clases, pero indicando arriba la palabra <<interface>>.

Tabla 6. Estructura de Interface

<< interface >>
NombreInterfaz
+ Atributos:

Imagen 7. Imagen 1 Diagrama de Clases Ecosistema VITAL - Propósito Ejemplo



MINISTERIO DE AMBIENTE Y DESARROLLO SOSTENIBLE	INSTRUCTIVO PARA LA ELABORACIÓN DE ARQUITECTURAS DE SOFTWARE	SOMOSIG Sistema Integrado de Gestión
	Proceso: Gestión de Servicios de Información y Soporte Tecnológico	
Versión: 1	Vigencia: 16/06/2023	Código: I-A-GTI-03

3.8.2. Diagrama Modelo De Datos

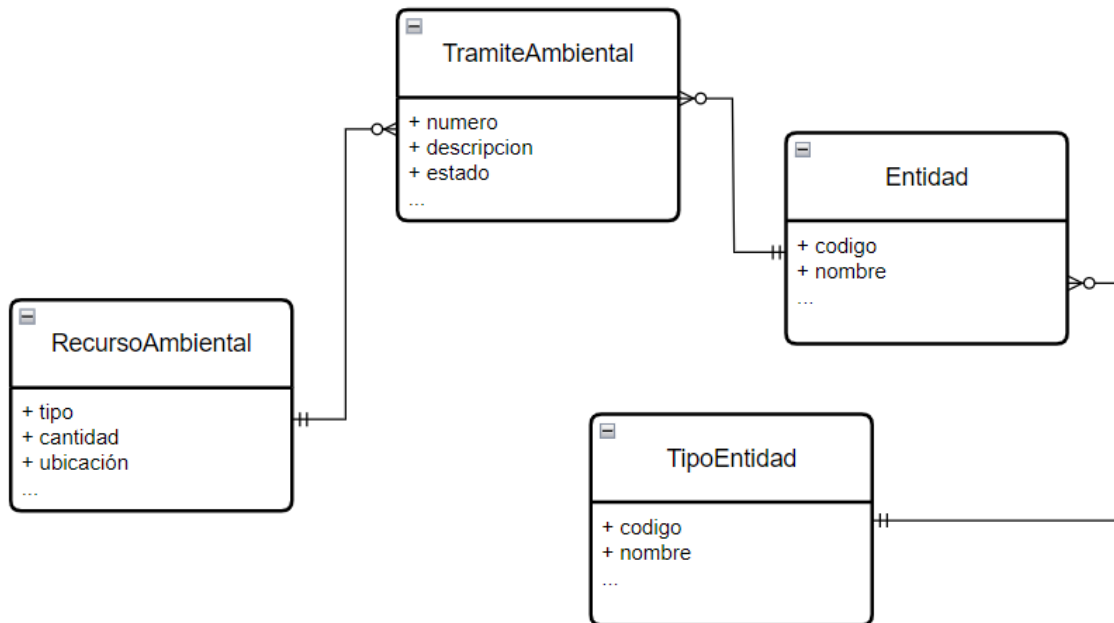
El modelado de datos permite la representación visual de los datos y hace cumplir las reglas que se tengan definidas dentro del Ministerio de Ambiente y Desarrollo Sostenible, el cumplimiento normativo y las políticas gubernamentales sobre los datos.


El modelo de datos diseña la base de datos en los niveles: físico y lógico, además, La estructura del modelo de datos ayuda a definir las tablas relacionales, las claves primarias y externas y los procedimientos almacenados.

Modelo de Datos Lógicos: El Modelo de Datos Lógico define **Cómo** se debe implementar el sistema independientemente del DBMS, es un mapa técnico de reglas y estructuras de datos y relaciones entre ellos.

Modelo de Datos Físicos: Este Modelo de datos describe **Cómo** se implementará el sistema real utilizando un sistema DBMS específico.

Imagen 8. Modelo de Datos - Ecosistema VITAL - Propósito Ejemplo



MINISTERIO DE AMBIENTE Y DESARROLLO SOSTENIBLE	INSTRUCTIVO PARA LA ELABORACIÓN DE ARQUITECTURAS DE SOFTWARE	 Sistema Integrado de Gestión
	Proceso: Gestión de Servicios de Información y Soporte Tecnológico	
Versión: 1	Vigencia: 16/06/2023	Código: I-A-GTI-03

4. TÉRMINOS Y/O CONCEPTOS DEL INSTRUCTIVO

- **Arquitectura de software:** es el diseño de alto nivel de la estructura de un sistema de información, definida a partir de modelos y estándares.
- **Atributo** es una especificación que define una propiedad de un objeto elemento o archive. también puede referirse o establecer el valor específico para una instancia determinada de los mismos. Sin embargo, actualmente, el término atributo se considera como si fuera una propiedad dependiendo de la tecnología que se use.

Para mejor entendimiento, los atributos deben ser considerados más correctamente como metadatos.
- **Base de datos:** recopilación organizada de información o datos estructurados, que normalmente se almacena de forma electrónica en un sistema informático.
- **Modelo C4:** Es una forma sencilla y eficaz de representar la arquitectura de software en diferentes niveles de abstracción. Consta de cuatro niveles: contexto del sistema, contenedores, componentes y código (Brown, 2013). El modelo C4 aborda las limitaciones de los enfoques de modelado existentes, proporcionando una notación estandarizada que es simple, intuitiva y lo suficientemente flexible como para representar sistemas de software de diferentes tamaños y complejidad.
- **Software:** Es el conjunto de los programas de cómputo, procedimientos, reglas, documentación y datos asociados, que forman parte de las operaciones de un sistema de información.

5. REFERENCIAS DEL INSTRUCTIVO

Bass, L., Clements, P., & Kazman, R. (2003). Software architecture in practice. Addison-Wesley Professional.

Brown, S. (2013). Software architecture for developers. Coding the Architecture.

Clements, P., Garlan, D., Little, R., Nord, R., & Stafford, J. (2003). Documenting software architectures: views and beyond. In 25th International Conference on Software Engineering (pp. 740-741). IEEE.

García-Holgado, A., Vázquez-Ingelmo, A., & García-Peñalvo, F. J. (2020). C4 model in a Software Engineering subject to ease the comprehension of UML and the software development process.

