

# **Documento de Arquitectura Mi Mutual, Sistema de Previsión, Asistencia y Solidaridad, Cooimeva, STEF - Coomeva**

Mi Mutual Coomeva - Mi Mutual, Sistema de Previsión, Asistencia y Solidaridad, Coomeva

**Versión** del producto 1.1670c6b de 13 Oct 2023

**Presentado a**

STEF - Coomeva

**Fecha**


13 Oct 2023

Los productos de esta etapa, MiMutual - Modificación Core Unidad de Solidaridad y Seguros, Contrato XXX-2023, ([Web](#)) están basados en el resultado del proyecto Coomeva Mi Mutual en curso. [Sharepoint STEF@1670c6b](#) del October 13, 2023.

## Autores

---

- **Equipo arquitectura STEF-COOMV.**

-  Usuario [e\\_hwong](#)

- Arquitecto, Stefanini

✉ — Enviar mensajes a Equipo arquitectura STEF-COOMV. <[e\\_hwong@stefanini.com](mailto:e_hwong@stefanini.com)>.

## Objetivo del Documento

---

Descripción de los productos del trabajo de arquitectura del proyecto MI MUTUAL de la Coomeva, Contrato XXX-2023. El principal propósito de este documento es informar de las decisiones sobre la disposición lógica y física de las partes del sistema. Por tanto, el documento contiene información estratégica, siendo en algunos casos el diseño detallado. Puntualmente, el documento refleja decisiones sobre la plataforma tecnológica seleccionada, así como consideraciones importantes para el diseño y desarrollo, con procura de garantizar una solución técnicamente viable y óptima para el proyecto.

# Control de Cambios

Tema	Mi Mutual Coomeva Mi Mutual, Sistema de Previsión, Asistencia y Solidaridad, Coomeva
Palabras clave	SIU, Stefanini, Coomeva, Análisis de brecha, GAP, Comparativa
Autor	
Fuente	
Versión	1.1670c6b del 13 Oct 2023
Vínculos	<a href="#">N003a Vista Segmento Coomeva SIU</a>

# Contenidos

---

# Introducción

## Propósito

---

Este documento tiene como propósito presentar la arquitectura del aplicativo Mi Mutual para STEF - Coomeva, según los requerimientos definidos durante la etapa de preventa y luego detallados en las historias de usuario.

La arquitectura será una guía para que el diseño y la implementación de los componentes que conforman la solución sean cobijados bajo lineamientos y premisas bien definidos, permitiendo a los elementos del sistema interactuar entre sí de forma coherente. La arquitectura será tomada como un diseño estratégico que establece restricciones globales para el diseño, define un marco inicial de trabajo para la implementación de los requerimientos funcionales y no funcionales.

La definición arquitectónica de este proyecto será un proceso evolutivo como tal. Este documento puede ser susceptible a cambios a medida que se vayan agregando nuevas funcionalidades o requisitos al sistema.

Uno de los principales propósitos de este documento es hacer una representación de las decisiones de disposición lógica y física de las partes del sistema; por tanto, es un diseño estratégico, no un diseño detallado. Puntualmente, refleja decisiones sobre la plataforma tecnológica seleccionada, así como consideraciones importantes para el diseño y desarrollo, con procura de garantizar una solución técnicamente viable y óptima para el proyecto.

# Restricciones Principales de Arquitectura

Informamos de las restricciones que hacen parte de Mi Mutual, y por tanto, a considerar en el ejercicio de arquitectura del presente proyecto.

Lista de restricciones de Mi Mutual, 2023.

1. Disponibilidad. Se requiere que el sistema esté disponible 7 X 24, el servicio prestado al cliente no se limita a horarios de oficina pues las compras pueden darse en cualquier momento
2. Escalabilidad. Se requiere que el sistema pueda llegar a atender hasta 1.000 clientes, para esto se requiere que el sistema se pueda extender horizontalmente de tal manera que pueda tener instalado en varios servidores para atender esta cantidad de usuarios. Todas las aplicaciones desarrolladas podrán ser escaladas horizontalmente para atender la demanda relacionada con el crecimiento de la empresa.
3. Reutilización. Se requiere que el sistema permita reutilizar sus componentes para prestar el mismo servicio a otras aplicaciones de la compañía. Para esto se va a desarrollar la aplicación utilizando servicios, separados y con asignación de responsabilidades, propias, de tal manera de que, si se requiere exponer servicios web sobre estas funcionalidades, no requiere cambios en la aplicación.
4. Autenticación. Autenticación es el proceso para determinar que alguien o un sistema es quien dice ser. Uso de estándar OAuth2 y JSON Web Token – JWT, para gestión de autenticación de servicios de la aplicación.
5. Autorización. Autorización se refiere a la validación que realiza un sistema para determinar si un usuario puede usar cierta funcionalidad. Uso de API de seguridad de Spring (spring-security) + OAuth2
6. Interoperabilidad – Movilidad. Interoperabilidad se refiere a la habilidad de un sistema de interactuar y comunicarse con sistemas heterogéneos a través de interfaces completamente definidas. Uso de estándar de web services REST + JSON.
7. Facilidad de Uso. Se refiere a la facilidad con que las personas pueden utilizar el sistema porque facilitan la lectura de los textos, descargan rápidamente la información y presentan funciones y menús sencillos, por lo que el usuario encuentra satisfechas sus consultas y cómodo su uso.
8. Verificación (QA). Es la capacidad del producto software que hace posible que el software modificado sea probado.
9. Estándares. Los estándares seleccionados por la solución de este proyecto, (Mi Mutual, Sistema de Previsión, Asistencia y Solidaridad, Coomeva, están determinados por el uso de las plataformas específicas determinadas por la implementación (desarrollo del software).

## Restricciones Secundarias

---

Otras restricciones a detallar.

1. Repositorio de datos.
2. Memoria, disco, CPU.
3. Requerimientos de rendimiento.

# Requisitos de Arquitectura Mi Mutual Central (no funcional)

Entendemos como requisitos de arquitectura aquellos requerimientos no visibles pero estructurales, medibles, y que impactan al funcionamiento, desarrollo y mantenimiento de la solución migración Mi Mutual, objeto de este proyecto, Mi Mutual Coomeva.

Definiremos estos requisitos de la solución a tener en cuenta al momento del desarrollo.

## Requerimientos generales

---

1. **Parametrización.** Crear desarrollos parametrizables necesarios para permitir la administración de la información de uso general.
2. **Interoperabilidad.** Crear desarrollos de Mi Mutual interoperables con otros sistemas de información de la entidad según requerimientos de los procesos.
3. **Diseño.** Los desarrollos complementarios deben responder a los criterios de bajo acoplamiento y alta cohesión.
4. **Reglas de negocio.** Las soluciones deben disponer de todas las validaciones y controles que garanticen la calidad, seguridad y unicidad de la información.
5. Para los casos que aplique, la solución debe contar con una integración con el servicio de correo de la Entidad.
6. Todos los desarrollos complementarios serán en su totalidad propiedad de la entidad, para lo cual la entidad podrá modificar y/o actualizar a futuro los procesos modelados, acorde a las necesidades; por tanto, deberán entregarse los derechos intelectuales y patrimoniales como parte de la documentación y el código fuente que corresponda.

# Requisitos Particulares de Arquitectura Mi Mutual Central (no funcional)

## Consistencia Mi Mutual (lógica)

Tabla 1: Requisito no. 1, Migración Mi Mutual, Consistencia.

Requisito	Extensibilidad Mi Mutual
Descripción	Unifica las entidades de negocio Coomeva, entre las que se incluyen a conciliaciones, publicaciones de relatoría, resoluciones, en artefactos reutilizables. Distinto de que estas entidades (y su lógica de negocio) estén dispersos entre los sistemas del Mi Mutual, estarán concentradas en un único artefacto correspondiente.
Calidad sistémica	La consistencia persigue que el resultado de la lógica de negocio sea la misma entre los módulos del Mi Mutual migrado. Esto redundará a mantenibilidad y gestión: tiende a tener un solo punto de cambio y dificulta la transferencia de dependencias implícitas a otros procesos.

## Mantenibilidad Mi Mutual

Tabla 2: Requisito no. 2, Mantenibilidad Mi Mutual.

Requisito	Mantenibilidad Mi Mutual
Descripción	Evitar las dependencia transitivas de los módulos misionales del Mi Mutual a componentes y sistemas de terceros o submódulos no misionales.
Calidad sistémica	La mantenibilidad por control de dependencias que optimiza el diseño Migración Mi Mutual está dada por el control de cambios no programados sobre los componentes misionales del Mi Mutual (corrupción de componentes). Ver Patrón de Diseño Migración Mi Mutual, más adelante en el documento.

## Extensibilidad Mi Mutual

Tabla 3: Requisito no. 3, Migración Mi Mutual, Flexibilidad.

Requisito	Extensibilidad Mi Mutual
Descripción	Concentración de los componentes de negocio, misionales, del Mi Mutual protegidos de cambios provenientes de otros sistemas. Ver Patrón de Diseño Migración Mi Mutual, más adelante en el documento.
Calidad sistémica	La extensibilidad que optimiza el diseño Migración Mi Mutual está dada por el intercambio de submódulos no misionales, como el gestor documental, sin afectación de los componentes misionales que este diseño protege.



# Vistas de Arquitectura Cotizador

- [Cotizador Web](#)
  - [ArqCotizador. 1. Contexto](#)
  - [ArqCotizador. 2. Contenedores](#)
  - [ArqCotizador. 4. Aplicación](#)
  - [ArqCotizador. 4a. Aplicación. Servicios](#)
  - [ArqCotizador. 4a. Dependencias](#)
  - [ArqCotizador. 5. Físico \(despliegue\)](#)
  - [ArqCotizador. 7. Datos. Negocio](#)

# Cotizador Web

## ArqCotizador. 1. Contexto

Mi Mutual. Coomeva, 2023.

Cotizador Web Mi Mutual. Contexto  
Mi Mutual: Áreas negocio,  
componente central Mi Mutual,  
servicios y funciones.

versión 0.1

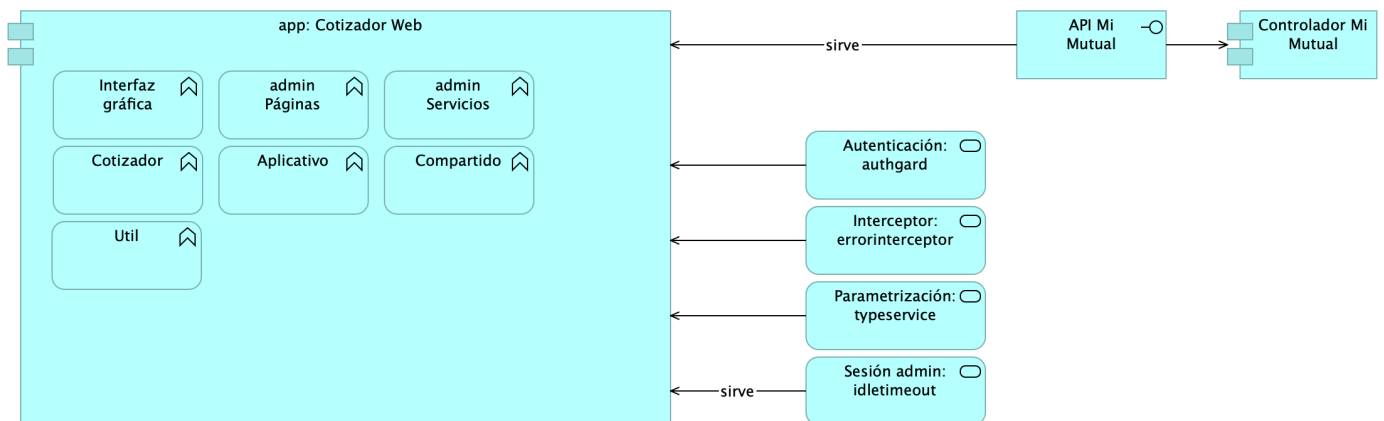


Imagen 1: Diagram: ArqCotizador. 1. Contexto

## Contexto Mi Mutual Web

La aplicación Cotizador Web hace parte de los módulos de interfaz web de Mi Mutual Central, representado por API Mi Mutual en el diagrama. Realizar cotizaciones de los planes de protección luego de la vinculación del asociado.

La estructura por módulos permite realizar aplicaciones escalables y robustas ya que permite organizar las partes de la aplicación, la organización en bloques, extender la aplicación con funcionalidades de librerías externas, proporcionar un entorno de resolución de plantillas y además permite especificar la forma de la carga de cada uno de los componentes y servicios que conforman un módulo.

## Módulos Externos

Los módulos externos son todas y cada uno de las herramientas que se utilizan para complementar con funcionalidades ya desarrolladas y tomadas desde un repositorio externo (NPM).

- TranslateModule: Manejo de internacionalización. Documentación: <https://github.com/ngx-translate/core>
- NgxMaskModule: Manejo de máscaras de input text. Documentación: <https://github.com/JsDaddy/ngx-mask>
- JwtModule: Manejo de token. Documentación: <https://github.com/auth0/angular2-jwt>
- sweetalert2: Manejo de alertas de mensajes. Documentación: <https://sweetalert2.github.io/>
- ngx-ui-loader: Manejo de Spinner para control de peticiones asíncronas. Documentación: <https://github.com/t-ho/ngx-ui-loader>
- Ngprime: Manejo de componentes visuales Documentación: <https://www.primefaces.org/primeng/#/>

- chart.js: componente utilizado para el manejo de graficas Documentación: <https://www.chartjs.org/docs/latest/>
- classlist.js: componete para el manejo de listas de datos en las gráficas Documentación: <https://www.chartjs.org/docs/latest/>
- cronstrue: componente para traducir una expresión cron a palabras Documentación: <https://github.com/bradymholt/cronstrue>
- file-saver: componente para descargar un archivo desde los bytes Documentación: <https://github.com/eligrey/FileSaver.js#readme>
- ngx-tinymce: Editor html para generación de plantillas para cartas Documentación: <https://cipchk.github.io/ngx-tinymce/#/>
- quill: componente para editor html Documentación: <https://quilljs.com/>

## Servicios Transversales

- AuthGuard: Validación de existencia de autenticación
- DeactiveGuard: Validación de salida de un componente
- ErrorInterceptor: Interceptor de Errores del back
- JwtInterceptor: Interceptor para inyectar el token
- AuthenticationService: Métodos para completar la autenticación
- TypesService: Consumo de servicios de parametrización
- IdleTimeoutService: Verificación de timeout del token

## Catálogo de Elementos

Name	Type	Description	Properties
<b>Controlador Mi Mutual</b>	application-component	Los componentes de este tipo se encargan de controlar los servicios rest de la aplicación, además en estos componentes se define la forma como se reciben y envían los datos de los servicios rest y la seguridad de cada uno de los métodos.	<i>modulo:</i> mimutual
<b>app: Cotizador Web</b>	application-component	pkg: MiMutualWeb	<i>modulo:</i> cotizador
<b>Aplicativo</b>	application-function		<i>modulo:</i> cotizador
<b>Compartido</b>	application-function		<i>modulo:</i> cotizador
<b>Cotizador</b>	application-function		<i>modulo:</i> cotizador
<b>Interfaz gráfica</b>	application-function		<i>modulo:</i> cotizador

Name	Type	Description	Properties
<b>Util</b>	application-function	En la Utilidades se especifican las clases que complementan una funcionalidad de un componente o servicio. * FormValidate: Clase que implementa un disparador de validación de todos los campos de un formulario. * CustomValidators: Creación de validaciones de campos.	<i>modulo: cotizador</i>
<b>admin Páginas</b>	application-function		<i>modulo: cotizador</i>
<b>admin Servicios</b>	application-function		<i>modulo: cotizador</i>
<b>API Mi Mutual</b>	application-interface		<i>modulo: mimutual</i>
<b>Autenticación: authgard</b>	application-service		
<b>Interceptor: errorinterceptor</b>	application-service		
<b>Parametrización: typeservice</b>	application-service		
<b>Sesión admin: idletimeout</b>	application-service		

## ArqCotizador. 2. Contenedores

Mi Mutual. Coomeva, 2023.

Estructura de componentes principales, Cotizador Web, Mi Mutual. Roles de componentes, separación responsabilidades.

versión 0.2

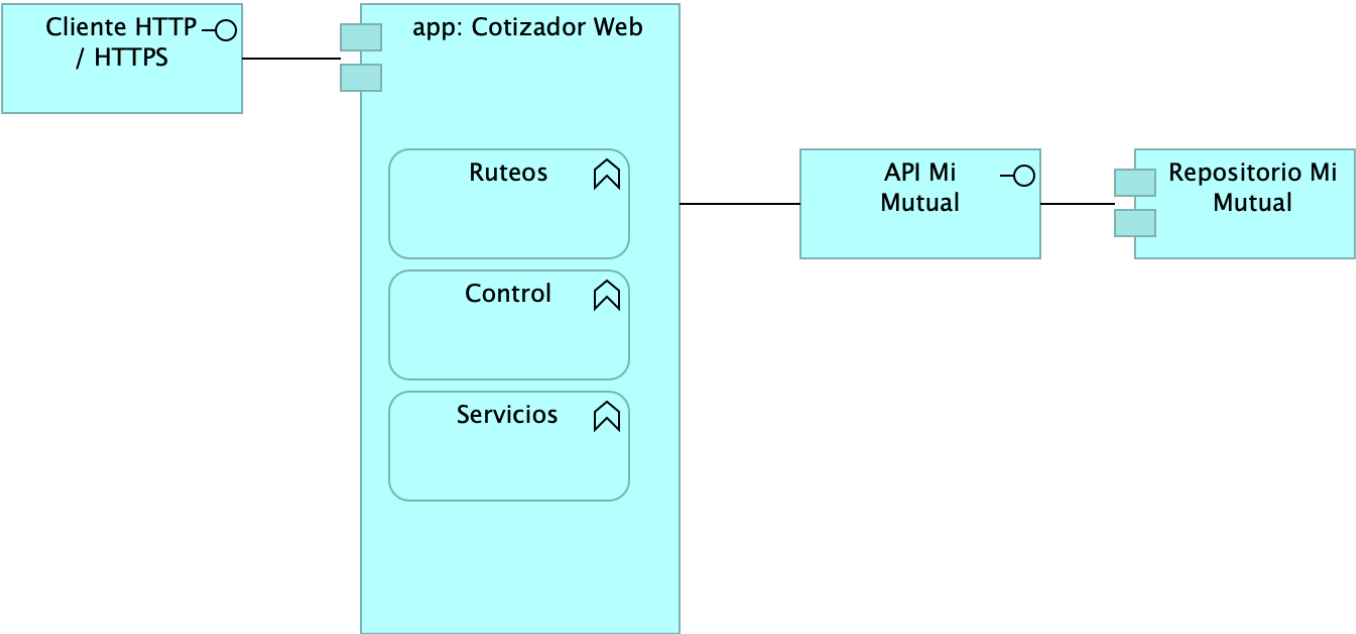


Imagen 2: Diagram: ArqCotizador. 2. Contenedores

Catálogo de Elementos

Name	Type	Description	Properties
Repositorio Mi Mutual	application-component	Antes SIPAS, Mi Mutual es una aplicación web compuesta por distintos módulos de software con arreglo a todas las actividades necesarias que soportan la operación de los productos y servicios que ofrece la Unidad de Solidaridad y Seguros de la Cooperativa. Para el manejo de la persistencia de datos se utilizará Spring Data el cual se apoya en la especificación de JPA y en la implementación de HIBERNATE además de complementar esta capa de persistencia con nuevas funcionalidades que facilitan el acceso a datos.	modulo: mimutual
app: Cotizador Web	application-component	pkg: MiMutualWeb	modulo: cotizador



Name	Type	Description	Properties
<b>Controlador Mi Mutual</b>	application-component	Los componentes de este tipo se encargan de controlar los servicios rest de la aplicación, además en estos componentes se define la forma como se reciben y envían los datos de los servicios rest y la seguridad de cada uno de los métodos.	<i>modulo: mimutual</i>
<b>MOD0.JwtModule</b>	application-component	Manejo de token. Documentación: <a href="https://github.com/auth0/angular2-jwt">https://github.com/auth0/angular2-jwt</a>	
<b>MOD0.Ngprime</b>	application-component	Manejo de componentes visuales Documentación: <a href="https://www.primefaces.org/primeng/#/">https://www.primefaces.org/primeng/#/</a>	
<b>MOD0.NgxMaskModule</b>	application-component	Manejo de máscaras de input text. Documentación: <a href="https://github.com/JsDaddy/ngx-mask">https://github.com/JsDaddy/ngx-mask</a>	
<b>MOD0.TranslateModule</b>	application-component	Manejo de internacionalización. Documentación: <a href="https://github.com/ngx-translate/core">https://github.com/ngx-translate/core</a>	
<b>MOD0.chart.js</b>	application-component	Componente utilizado para el manejo de graficas Documentación: <a href="https://www.chartjs.org/docs/latest/">https://www.chartjs.org/docs/latest/</a>	
<b>MOD0.classlist.js</b>	application-component	Componete para el manejo de listas de datos en las gráficas Documentación: <a href="https://www.chartjs.org/docs/latest/">https://www.chartjs.org/docs/latest/</a>	
<b>MOD0.cronstrue</b>	application-component	Componente para traducir una expresión cron a palabras Documentación: <a href="https://github.com/bradymholt/cronstrue">https://github.com/bradymholt/cronstrue</a>	
<b>MOD0.file-saver</b>	application-component	Componente para descargar un archivo desde los bytes Documentación: <a href="https://github.com/eligrey/FileSaver.js#readme">https://github.com/eligrey/FileSaver.js#readme</a>	
<b>MOD0.ngx-tinymce</b>	application-component	Editor html para generación de plantillas para cartas Documentación: <a href="https://cipchk.github.io/ngx-tinymce/#/">https://cipchk.github.io/ngx-tinymce/#/</a>	

Name	Type	Description	Properties
<b>MOD0.ngx-ui-loader</b>	application-component	Manejo de Spinner para control de peticiones asíncronas. Documentación: <a href="https://github.com/t-ho/ngx-ui-loader">https://github.com/t-ho/ngx-ui-loader</a>	
<b>MOD0.quill</b>	application-component	Ccomponente para editor html Documentación: <a href="https://quilljs.com/">https://quilljs.com/</a>	
<b>MOD0.sweetalert2</b>	application-component	Manejo de alertas de mensajes. Documentación: <a href="https://sweetalert2.github.io/">https://sweetalert2.github.io/</a>	
<b>Repositorio Mi Mutual</b>	application-component	Antes SIPAS, Mi Mutual es una aplicación web compuesta por distintos módulos de software con arreglo a todas las actividades necesarias que soportan la operación de los productos y servicios que ofrece la Unidad de Solidaridad y Seguros de la Cooperativa. Para el manejo de la persistencia de datos se utilizará Spring Data el cual se apoya en la especificación de JPA y en la implementación de HIBERNATE además de complementar esta capa de persistencia con nuevas funcionalidades que facilitan el acceso a datos.	<i>modulo: mimutual</i>
<b>app: Cotizador Web</b>	application-component	pkg: MiMutualWeb	<i>modulo: cotizador</i>
<b>app: Implementación de Servicios</b>	application-component	Los componentes de este tipo se encargan de controlar y almacenar toda la lógica del negocio, validaciones y todo lo referente a procesamiento de datos.	<i>modulo: mimutual</i>
<b>app: Mi Mutual Central</b>	application-component	Antes SIPAS, Mi Mutual es una aplicación web compuesta por distintos módulos de software con arreglo a todas las actividades necesarias que soportan la operación de los productos y servicios que ofrece la Unidad de Solidaridad y Seguros de la Cooperativa.	<i>modulo: mimutual</i>



Name	Type	Description	Properties
<b>Interfaz transporte</b>	application-interface	Feign Client. Integración con otros sistemas para facilitar los procesos de vinculación, retiro, reactivación o fallecimiento de asociados.	
<b>Application Service</b>	application-service	Otros servicios del contexto de Mi Mutual Central.	
<b>Autorizaciones</b>	application-service	Autorizaciones: Administración de peticiones de autorización y sus correspondientes aprobaciones usando el servicio del flujo de procesos.	
<b>Certificados</b>	application-service	Certificados: Permite la generación de los certificados de valores de protección y contribuciones pagadas, de retención en la fuente, de pagos de perseverancia y de cobertura de auxilio funerario.	
<b>Configuración</b>	application-service	Configuración o parametrización de factores para realizar los cálculos de las contribuciones de los asociados a la Cooperativa para cada uno de los productos adquiridos.	
<b>Facturación y Recaudo</b>	application-service	Administración de la facturación y recaudo diario de los productos	
<b>Gestión de Beneficiarios</b>	application-service	Gestión de Beneficiarios: Permite administrar la información relacionada con los beneficiarios del Asociado, permitiendo ejecutar operaciones de consulta, inserción y modificación	

Name	Type	Description	Properties
<b>Gestión de Productos</b>	application-service	Gestión de productos del fondo mutual y auxilio funerario que involucran lo relacionado a las siguientes coberturas: * Fondo de Solidaridad: Incapacidades temporales, Incapacidades Permanentes (total, parcial), Perseverancia 60, 62, 65, 70 años, Perseverancias Anticipadas, Fallecimiento Asociado (Auxilio por muerte), Desempleo, Disminución de ingresos y enfermedades graves; Rentas por hospitalización, Enfermedades de Alto Costo, Pólizas de seguros personales y patrimoniales, Planes educativos, Segunda opinión médica, Asistencias. * Auxilio Funerario: Fallecimiento de familiares directos (inscritos) del Asociado.	
<b>Gestión de Usuarios</b>	application-service	Gestión de Usuarios: Administración de la información relacionada con los usuarios del sistema. Este componente se comunica con el servicio unificado de autenticación y autorización que devuelve los permisos que un usuario posee sobre las opciones que proporciona el sistema.	
<b>SS02.reporte - cotizacion</b>	application-service		
<b>SS02.reporte - estado - cotizacion</b>	application-service		
<b>Simuladores</b>	application-service	Simuladores: Funcionalidades que permiten generar las simulaciones de los diferentes planes o modificaciones (incrementos y disminuciones) a los productos del Asociado.	
<b>Analistas</b>	business-role	Analistas y auxiliares de servicio regional y nacional, agentes del centro de contacto, auditores médicos, analistas de operaciones (aseguramiento y facturación) y jefes.	

Name	Type	Description	Properties
<b>Asesores</b>	business-role	Asesores integrales	
<b>Auxiliares servicio</b>	business-role	Analistas y auxiliares de servicio regional y nacional, agentes del centro de contacto, auditores médicos, analistas de operaciones (aseguramiento y facturación) y jefes.	
<b>Módulos Externos</b>	grouping		
<b>Unidad de Solidaridad y Seguros</b>	grouping	La Unidad de Solidaridad y Seguros cuenta con un software integrado para su core de negocio denominado SIPAS (Sistema de Previsión, Asistencia y Solidaridad)	
<b>Servicio de Almacenamiento de Datos</b>	technology-service		
<b>Servicio de Red</b>	technology-service		
<b>Servicio de archivos</b>	technology-service		

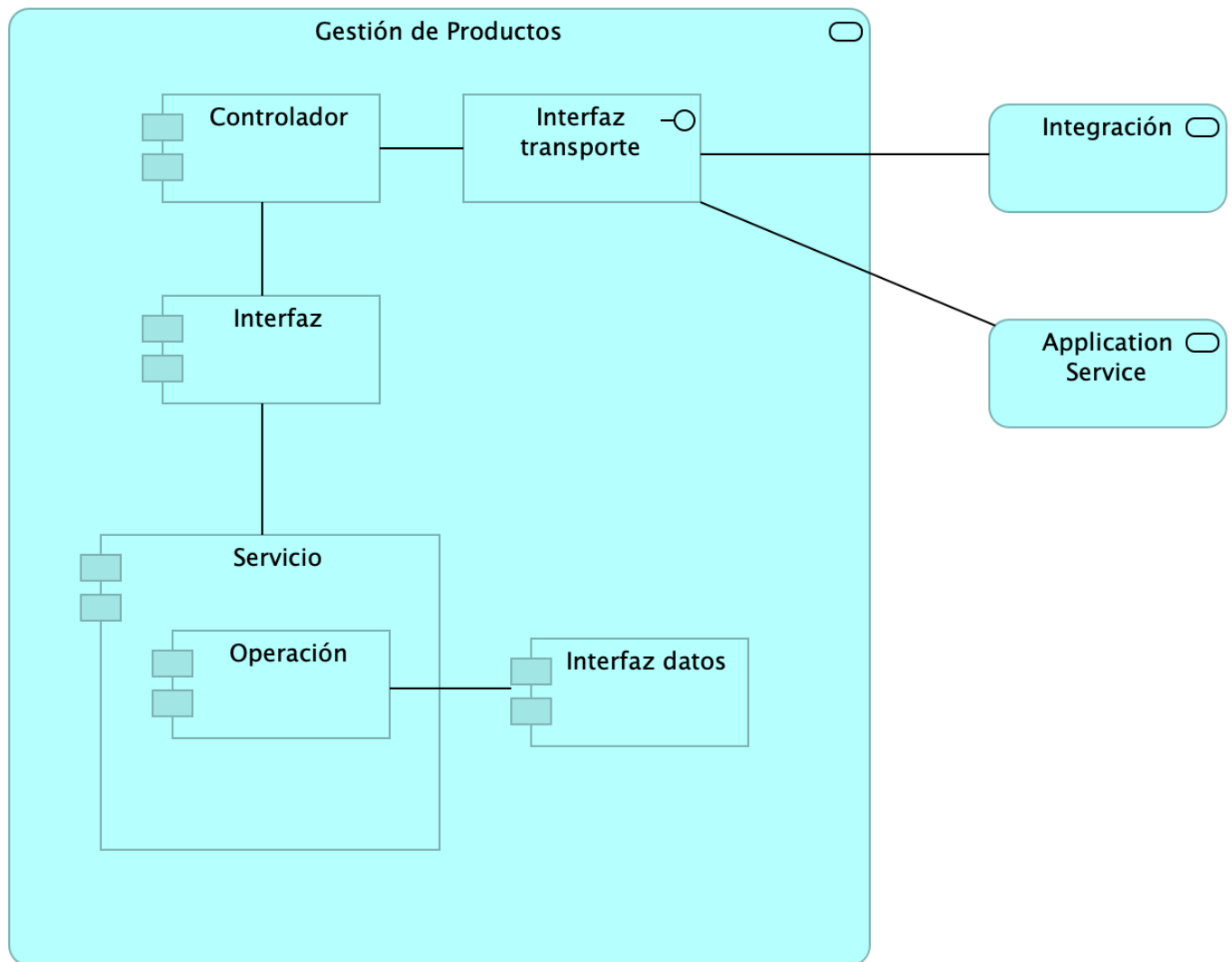
## ArqCotizador. 4a. Aplicación. Servicios

---

Mi Mutual. Coomeva, 2023.

Especificaciones de Servicios.  
Aplicación Cotizador Web, Mi Mutual.  
Estado Actual. Estructura interna,  
comunicación e interfaces.

versión 0.1



**Imagen 4:** Diagram: ArqCotizador. 4a. Aplicación. Servicios

Composición interna de los servicios de Mi Mutual Central, Mi Mutual Web, Cotizador Web.

## Catálogo de Elementos

Name	Type	Description	Properties
Controlador	application-component	Controlador interno del servicio. Punto de entrada a la lógica de expuesta.	

Name	Type	Description	Properties
<b>Interfaz</b>	application-component	Interfaz de inversión de dependencia a las clases del servicio.	
<b>Interfaz datos</b>	application-component	Acceso a datos del modelo del contexto de Mi Mutual Central.	
<b>Operación</b>	application-component		
<b>Servicio</b>	application-component	Exposición de componentes de negocio.	
<b>Interfaz transporte</b>	application-interface	Feign Client. Integración con otros sistemas para facilitar los procesos de vinculación, retiro, reactivación o fallecimiento de asociados.	
<b>Application Service</b>	application-service	Otros servicios del contexto de Mi Mutual Central.	
<b>Gestión de Productos</b>	application-service	Gestión de productos del fondo mutual y auxilio funerario que involucran lo relacionado a las siguientes coberturas: * Fondo de Solidaridad: Incapacidades temporales, Incapacidades Permanentes (total, parcial), Perseverancia 60, 62, 65, 70 años, Perseverancias Anticipadas, Fallecimiento Asociado (Auxilio por muerte), Desempleo, Disminución de ingresos y enfermedades graves; Rentas por hospitalización, Enfermedades de Alto Costo, Pólizas de seguros personales y patrimoniales, Planes educativos, Segunda opinión médica, Asistencias. * Auxilio Funerario: Fallecimiento de familiares directos (inscritos) del Asociado.	
<b>Integración</b>	application-service		

## ArqCotizador. 4a. Dependencias

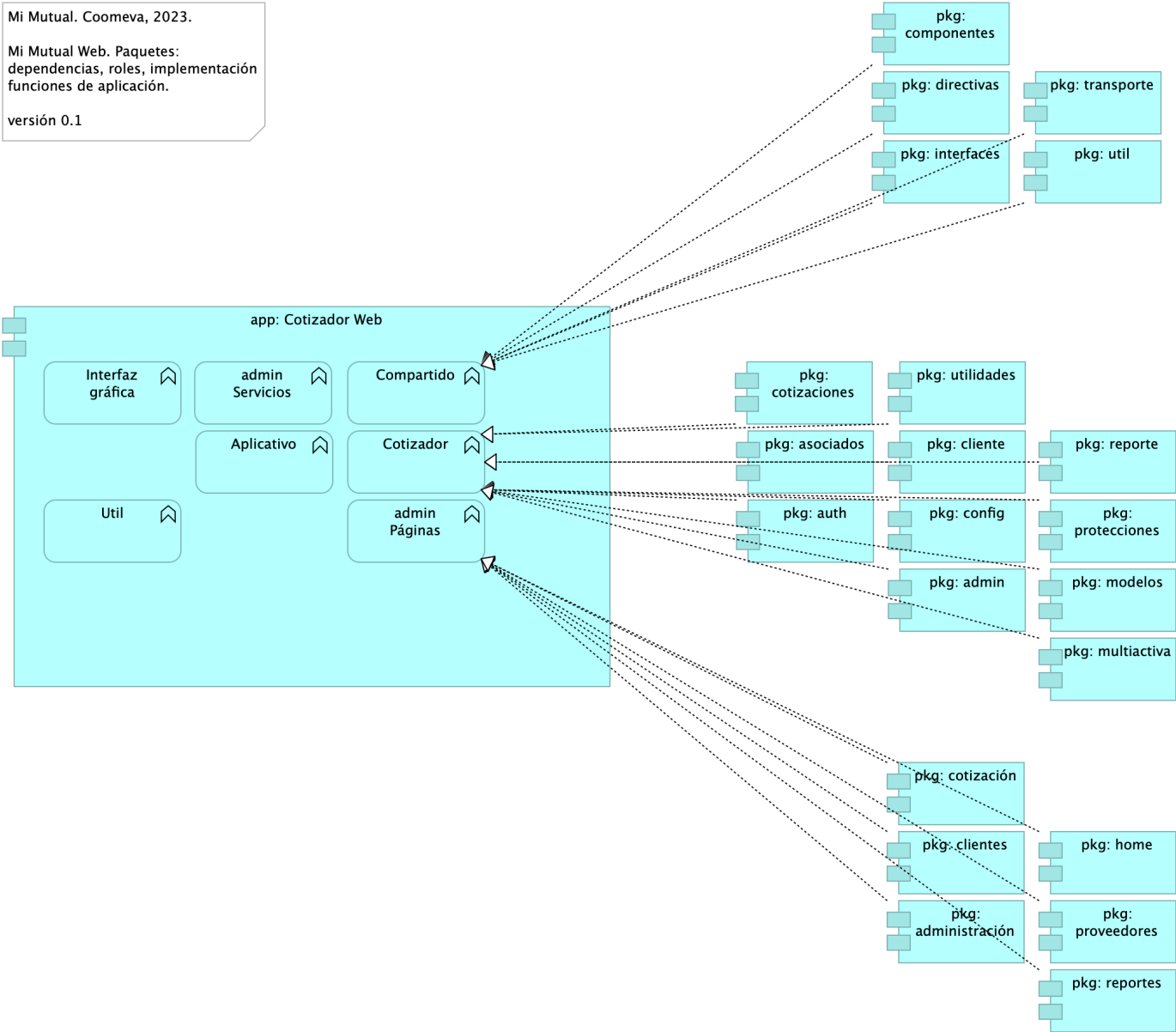


Imagen 5: Diagram: ArqCotizador. 4a. Dependencias

## Paquetes y Dependencias Cotizador Web

Módulos y componentes que hacen parte de la estructura de la aplicación Cotizador Web (basado en Angular 12 [1](#)).

## Módulos Cotizador Web

La estructura por módulos actual apunta a la escalabilidad y mantenimiento del Cotizador en términos de: organizar las partes de la aplicación, organización los bloques, extender la aplicación con librerías externas, proporcionar un entorno de resolución de plantillas y además, distribuir las cargas de los componentes y servicios que usa la aplicación.

## Catálogo de Elementos

Name	Type	Description	Properties
app: Cotizador Web	application-component	pkg: MiMutualWeb	modulo: cotizador

Name	Type	Description	Properties
<b>pkg: admin</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: administración</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: asociados</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: auth</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: cliente</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: clientes</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: componentes</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: config</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: cotizaciones</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: cotización</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: directivas</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: home</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: interfaces</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>

Name	Type	Description	Properties
<b>pkg: modelos</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: multiactiva</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: protecciones</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: proveedores</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: reporte</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: reportes</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: transporte</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: util</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: utilidades</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>Aplicativo</b>	application-function		<i>modulo: cotizador</i>
<b>Compartido</b>	application-function		<i>modulo: cotizador</i>
<b>Cotizador</b>	application-function		<i>modulo: cotizador</i>
<b>Interfaz gráfica</b>	application-function		<i>modulo: cotizador</i>



Name	Type	Description	Properties
Util	application-function	<p>En la Utilidades se especifican las clases que complementan una funcionalidad de un componente o servicio.</p> <p>* FormValidate: Clase que implementa un disparador de validación de todos los campos de un formulario.</p> <p>* CustomValidators: Creación de validaciones de campos.</p>	<i>modulo:</i> cotizador
admin Páginas	application-function		<i>modulo:</i> cotizador
admin Servicios	application-function		<i>modulo:</i> cotizador

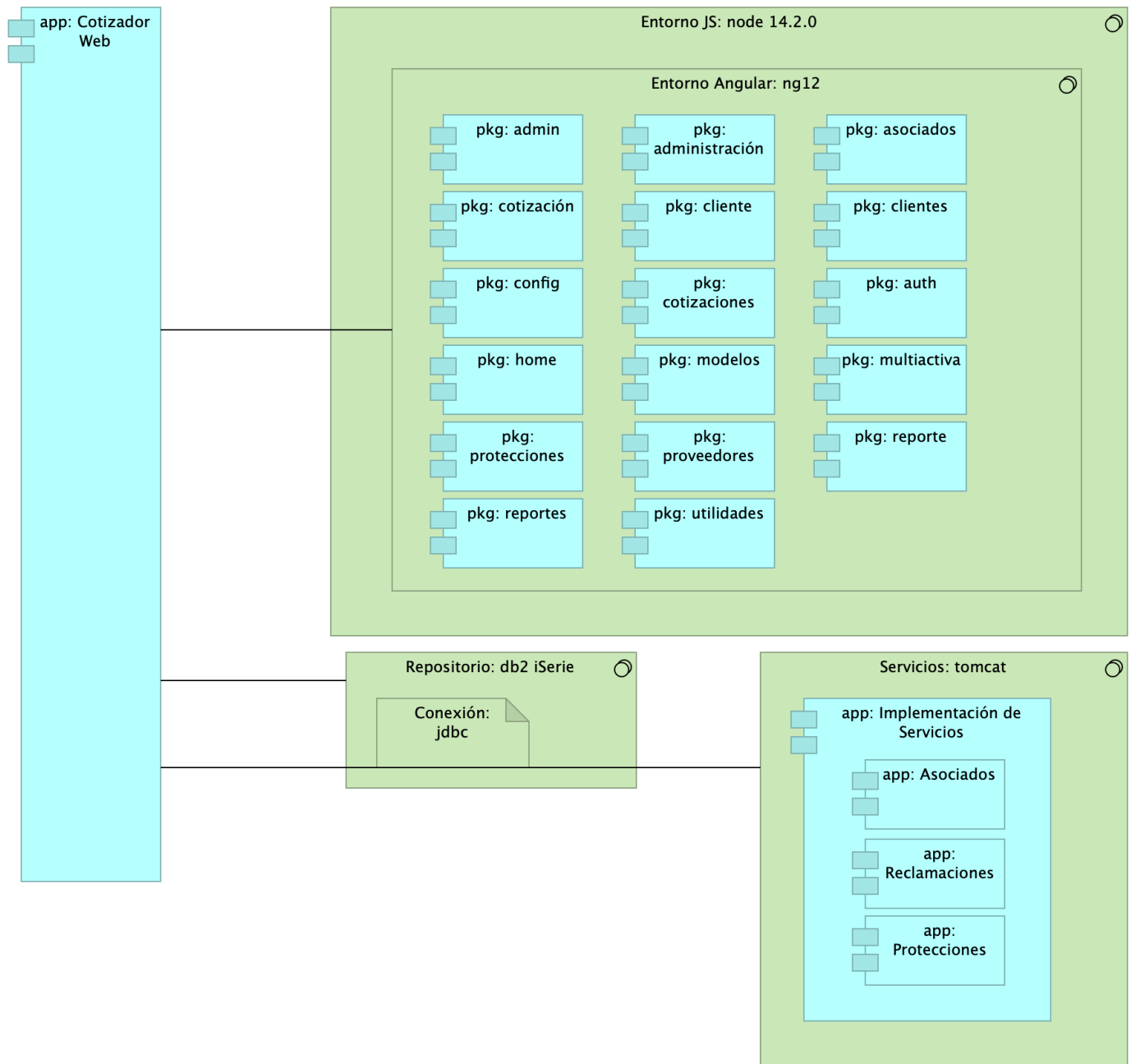
## ArqCotizador. 5. Físico (despliegue)

---

Mi Mutual. Coomeva, 2023.

Distribución física Cotizador Web, Mi Mutual. Estado actual, 2023.

versión 0.1.1



**Imagen 6:** Diagram: ArqCotizador. 5. Físico (despliegue)

## Especificaciones de Despliegue Cotizador Web

Detalles de configuración del proyecto Mi Mutual en el espacio de trabajo local (2022).

### Recursos Requeridos

- Git. Se debe instalar git para poder realizar la clonación de cada uno de los proyectos mas adelante.
- Instalación SmartGit. Se debe instalar Smartgit para poder realizar la clonación de cada uno de los proyectos mas adelante, este es opcional ya que es una interfaz gráfica de git mas amigable para el

usuario en caso que no desee trabajar con la consola.

- DBeaver. Se debe instalar DBeaver para poder acceder a la base de datos.
- Instalación Maven. Se debe instalar maven para poder compilar los proyectos, nos debemos asegurar de instalar la versión 3.6.3, en caso que no se encuentra en la página oficial copiar la carpeta que esta en el repositorio a archivo de programas.
- Java 8. Se debe instalar Java para poder desplegar los proyectos mas adelante, nos debemos asegurar de instalar la versión 8.
- STS. Se debe instalar el IDE para realizar modificaciones a los proyectos back mas adelante en este caso Spring Tools 4 for Eclipse. La carpeta que genera el instalador la copiamos a archivos de programa.
- Instalación Lombok. Se debe instalar el lombok seleccionando el IDE que acabamos de instalar en este caso el STS.
- Postman. Se debe instalar el postman para poder consumir los servicios del backend mas adelante cuando ya se hayan desplegado.
- Node Js. Se debe instalar Node Js para configurar el proyecto front mas adelante, nos debemos asegurar de instalar la versión v14.2.0.
- Visual Studio Code. Se debe instalar el IDE para realizar modificaciones al proyecto front mas adelante en este caso Visual Studio code.

k. Angular 9.1.12 o superior.

## Catálogo de Elementos

Name	Type	Description	Properties
<b>app: Asociados</b>	application-component	Contiene todas las funcionalidades relacionadas con consulta y creación de asociados y beneficiarios.	<i>modulo:</i> mimutual
<b>app: Cotizador Web</b>	application-component	pkg: MiMutualWeb	<i>modulo:</i> cotizador
<b>app: Implementación de Servicios</b>	application-component	Los componentes de este tipo se encargan de controlar y almacenar toda la lógica del negocio, validaciones y todo lo referente a procesamiento de datos.	<i>modulo:</i> mimutual
<b>app: Protecciones</b>	application-component	Contiene todas las funcionalidades relacionadas con la gestión y configuración de productos y protecciones.	<i>modulo:</i> mimutual
<b>app: Reclamaciones</b>	application-component	Contiene todas las funcionalidades relacionadas con la gestión de reclamaciones, liquidaciones y pagos.	<i>modulo:</i> mimutual
<b>pkg: admin</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo:</i> cotizador

Name	Type	Description	Properties
<b>pkg: administración</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: asociados</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: auth</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: cliente</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: clientes</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: config</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: cotizaciones</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: cotización</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: home</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: modelos</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: multiactiva</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: protecciones</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: proveedores</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>

Name	Type	Description	Properties
<b>pkg: reporte</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: reportes</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>pkg: utilidades</b>	application-component	controller: Almacenan todas las clases que constituyen los servicios rest de la aplicación.	<i>modulo: cotizador</i>
<b>Conexión: jdbc</b>	artifact		<i>modulo: cotizador</i>
<b>Entorno Angular: ng12</b>	system-software		<i>modulo: cotizador</i>
<b>Entorno JS: node 14.2.0</b>	system-software		<i>modulo: cotizador</i>
<b>Repositorio: db2 iSerie</b>	system-software		<i>modulo: cotizador</i>
<b>Servicios: tomcat</b>	system-software		<i>modulo: mimutual</i>

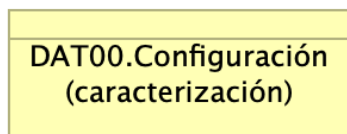
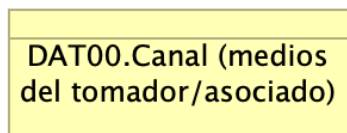
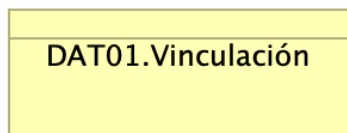
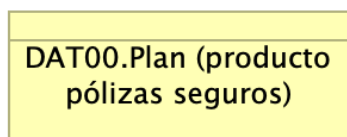
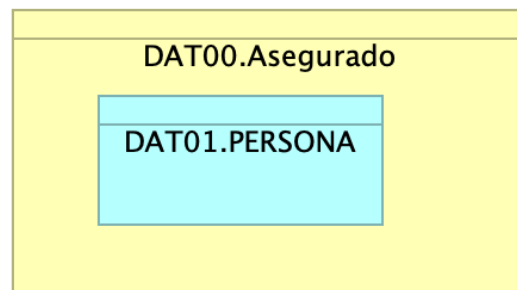
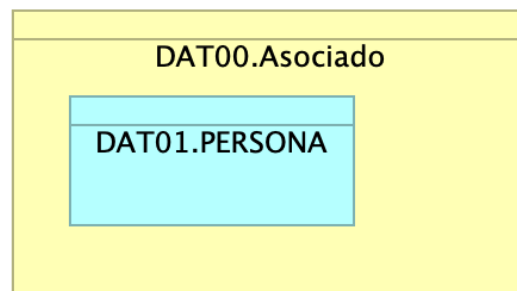
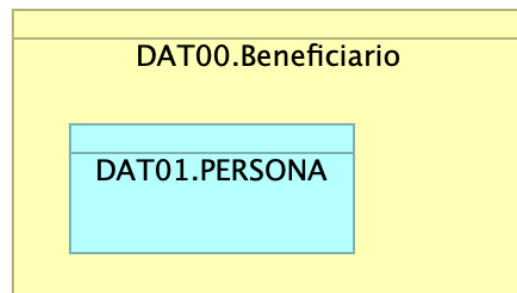
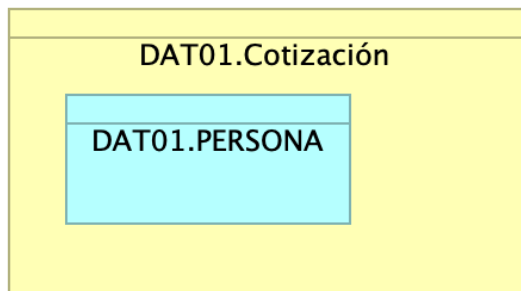
## ArqCotizador. 7. Datos. Negocio

---

Mi Mutual. Coomeva, 2023.

Mi Mutual Web. Entidades:  
Estructuras, objeto, relaciones con  
aplicación.

versión 0.1



**Imagen 7:** Diagram: ArqCotizador. 7. Datos. Negocio

## Entidades de Negocio Mi Mutual

Dominios de datos de negocio. Entidades independiente de la plataforma y de la tecnología.

- Configuración (caracterización de productos, plan)
- Plan (producto pólizas seguros)
- Canal (medios del tomador/asociado)
- Parametros globales (catálogos)
- Portafolio de asociado
- Asociado

- Facturación
- Beneficiario

## Catálogo de Elementos

Name	Type	Description	Properties
DAT00.Asegurado	business-object		
DAT00.Asociado	business-object		
DAT00.Beneficiario	business-object		
DAT00.Canal (medios del tomador/asociado)	business-object		
DAT00.Configuración (caracterización)	business-object	Caracterización de productos, planes, parámetros	
DAT00.Plan (producto pólizas seguros)	business-object		
DAT01.Cotización	business-object		
DAT01.Vinculación	business-object		
DAT01.PERSONA	data-object		
DAT01.PERSONA	data-object		
DAT01.PERSONA	data-object		
DAT01.PERSONA	data-object		

Generated on: Fri Oct 13 2023 14:46:27 GMT-0500 (COT)

## Requerimientos de Administración Mi Mutual Central

1. Las soluciones deben permitir la administración de los Roles de Usuarios: esta funcionalidad debe permitir configurar los diferentes roles de los usuarios funcionales de los procesos.
2. Administrar los Perfiles de acceso por rol: Esta funcionalidad permitirá configurar a que funcionalidades u opciones de la solución puede entrar un usuario con un rol específico.  
Administrar los Usuarios de la Solución: Esta funcionalidad debe permitir configurar, activar, desactivar usuarios de las soluciones desarrolladas.
3. Para los casos que aplique se debe asociar el desarrollo con el mecanismo de Firmas (digital, electrónica o mecánica): Esta funcionalidad debe permitir configurar los usuarios que tienen permitida la aprobación de documentos desde la solución implementada, a través del tipo de firma que corresponda.
4. Administrar los Permisos de acceso: Esta funcionalidad permite definir específicamente a que servicios de la solución puede ingresar un usuario (CRUD).
5. En los desarrollos se debe contar con un módulo de auditoría que permita generar consultas para conocer quién y cuándo se ha realizado una actuación determinada dentro de procesos críticos, almacenando el código del usuario la actuación, la acción, la fecha, la hora, y la dirección IP de la máquina.
6. Las soluciones deben permitir la configuración de permisos de consulta con diferentes alcances para cada tipo de usuario.
7. Desde la interfaz de usuario se debe poder crear, modificar o inactivar usuarios, perfiles o roles, permisos a las diferentes funcionalidades de la solución.
8. Las soluciones deben permitir la definición de varios tipos de usuario.

9. Las soluciones deben permitir la parametrización de los consecutivos que maneja la entidad para los diferentes documentos generados por las soluciones.
10. Debe permitir parametrizar la vinculación del consecutivo a un documento en forma manual o automática.
11. Las soluciones deben permitir que se configure la autenticación de forma interna integrándose con LDAP el acceso de los usuarios y actores de las diferentes dependencias de la entidad que interactúen con los demás sistemas.

## **Requerimientos de Seguridad Mi Mutual Central**

---

1. Las soluciones deben dar cumplimiento a las políticas institucionales del sistema de gestión de seguridad de la información establecidas por la entidad que busca garantizar la confidencialidad, integridad y disponibilidad de la información que se genera, procesa, almacena y/o transmite en los sistemas de Información de la Entidad.
2. Las soluciones de automatización de procesos a implementar deben permitir la Gestión de Seguridad de Usuarios, grupos de usuarios y asignación de Roles y perfiles de usuarios, permitiendo asociar las acciones disponibles en la solución con respecto a roles de usuario, permitiendo parametrizar las funcionalidades que cada actor puede usar en la solución.
3. Un usuario puede estar asociado a uno o más roles, de tal manera que los menús de navegación de la solución se muestran o despliegan dependiendo de las acciones asociadas a cada rol de usuario, permitiendo así que cuando el usuario es autenticado correctamente, la solución verifica los roles que tiene activos para otorgarle únicamente las acciones autorizadas.
4. El diseño de la solución debe definir los criterios necesarios para asegurar la trazabilidad y auditoría sobre las acciones de creación, actualización, modificación o borrado de los componentes de información, de tal manera que la solución debe permitirle al administrador de la solución parametrizar las tablas y eventos que pueden auditarse.
5. Las soluciones deben tener en cuenta mecanismos que aseguren el registro histórico para poder mantener la trazabilidad de las acciones realizadas por los usuarios, contemplando el registro de auditoría que contiene información de fecha y hora, identificación del registro, tabla afectada, descripción del evento, tipo de evento, usuario que realiza la acción, identificación de sesión y dirección IP del usuario que efectuó la transacción.
6. La solución debe proveer una consulta que permita a un usuario con los privilegios asignados, consultar los registros de auditoría, aplicando criterios de filtro (usuario, maquina, rango de fechas y tipo de operación).
7. Las soluciones deben integrarse con LDAP – (Lightweight Directory Access Protocol) para los procesos de inicio de sesión y autenticación. La solución debe soportar la integración Nativa con Active Directory de Microsoft. Para usuarios externos el mecanismo de autorización, autenticación y acceso será controlado a través del modelo de seguridad de la solución (no habrá autenticación para usuarios externos).
8. Las soluciones deben cumplir con los lineamientos de seguridad relacionados a su utilización a través de redes públicas y privadas, garantizando la confidencialidad e integridad de la información y acceso a ella.
9. Debe evidenciar que, a través de pruebas de vulnerabilidad, garantiza la seguridad de la información. Estas pruebas deben suministrar evidencia de que se usaron umbrales de seguridad para establecer niveles mínimos aceptables de calidad de la seguridad y de la privacidad.
10. Debe incluir un mecanismo de cifrado de los datos que se transportan entre los diferentes componentes tecnológicos y los datos sensibles de la base de datos que representen un alto nivel de confidencialidad.
11. A nivel de la base de datos debe poder definirse reglas de validación de integridad de datos (unicidad, referencial y negocio).



12. Debe contemplar el cumplimiento de la normatividad vigente en cuanto a protección de datos personales y debe permitir el manejo de excepciones.
13. Para los casos que aplique se debe permitir el manejo de certificados y/o firmas digitales en los documentos que así se definan para efectos de aprobación y digitalización.
14. Debe contemplar las prácticas de desarrollo seguro de aplicaciones y/o implementación segura de productos, para su naturaleza Web based.
15. Debe funcionar sobre protocolo SSL (certificados internos de la entidad cuando los sistemas de información sean internas y certificados validos públicamente cuando los sistemas de información estén expuestas a internet).
16. Debe entregar un procedimiento para el respaldo de la información de acuerdo con las necesidades de la entidad.
17. Debe incluir uso de criptografía para transacciones y/o campos sensibles según lo indiquen las normas vigentes y las necesidades específicas del negocio de acuerdo como lo determine la entidad.
18. Debe contemplar un modelo de datos que garantice base de datos única para evitar que se pueda presentar duplicidad de información.
19. En la información confidencial solo puede ser consultada por los perfiles autorizados e igualmente restringir documentos de consulta según los privilegios o permisos asociados.
20. A nivel de la base de datos debe poder definirse reglas de validación de integridad de datos (unicidad, referencial y negocio).
21. Debe cerrar las transacciones luego de máximo 10 minutos de inactividad.
22. Debe incluir controles de bloqueo de cuenta después de un máximo de 5 intentos erróneos a fin de evitar ataques de fuerza bruta.
23. Debe evidenciar el resultado positivo frente apruebas de ethical hacking, análisis de vulnerabilidades, carga, estrés y desempeño antes de la puesta en operación de acuerdo con los lineamientos de la entidad.
24. Debe cumplir con todos los lineamientos de desarrollo seguro establecidos en The OWASP Foundation recomendados en la "Guía de desarrollo OWASP" y "OWAS Cheat Sheet".

# Referencias

[1] [2] [3] [eservices5-23?] [eservices6-12?] [eservices7-23?] [bptrends07?]

1. **Stefanini. Proyecto de mejoramiento SIU de coomeva. Fase i**  
Stefanini, Coomeva  
(2022-06) <https://hwong23.github.io/fna-devdoc-f1/v/6497aef0f15c3591f0728e4c42cb2c26c13b43aa/>
  2. **Procuraduría general de la nación. Anexo - especificaciones técnicas 19-05-2023**  
Coomeva  
(2023-05) <https://hwong23.github.io/fna-devdoc-f1/v/6497aef0f15c3591f0728e4c42cb2c26c13b43aa/>
  3. **Coomeva manual técnico sharepoint, versión 1**  
Stefanini, Coomeva  
(2022-05) <https://hwong23.github.io/fna-devdoc-f1/v/6497aef0f15c3591f0728e4c42cb2c26c13b43aa/>
- 

1. Angular 2 tiene una arquitectura Modelo Vista Controlador (MVC) con el fin de hacer el desarrollo gestionado.↩