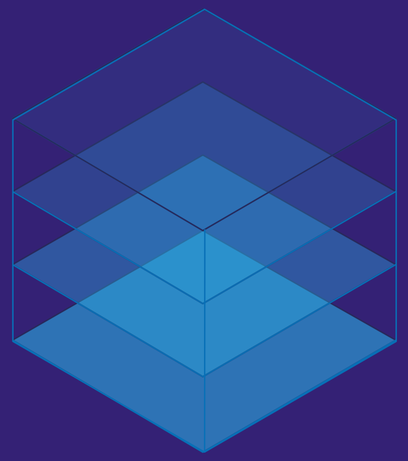


MISO

Maestría en Ingeniería de Software

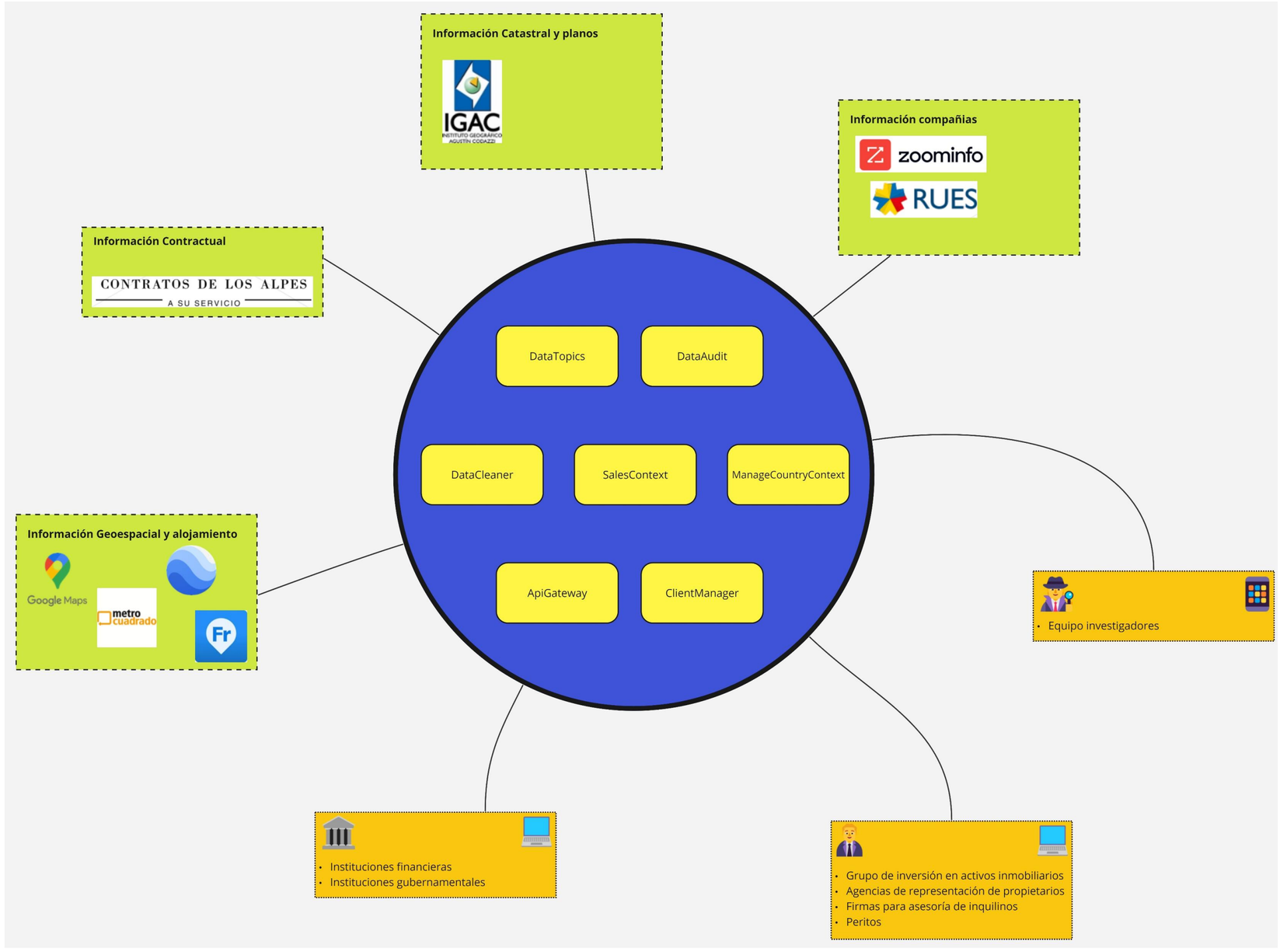
Entrega 2: Diseño Táctico

Requerimientos de calidad

Atributo de calidad	Prioridad (L/M/H)	Justificación
Rendimiento	**H**	La capacidad de escalar horizontalmente el sistema para manejar picos de carga y demanda se vuelve crítica. El sistema debe ser capaz de ajustar dinámicamente los recursos según sea necesario para garantizar un **rendimiento** óptimo y una experiencia del usuario sin problemas, especialmente al expandirse a nuevas localidades y mercados. Esta capacidad de escalabilidad aseguraría que el sistema pueda **crecer y adaptarse** a medida que la compañía expanda sus operaciones globalmente.
Mantenibilidad	**H**	Se identifica la necesidad de contar con un sistema que sea altamente **escalable**, **modificable** y extensible tanto en el aspecto tecnológico como en el de recursos humanos. Esto implica que la arquitectura del sistema debe estar diseñada de manera **modular** y clara, lo que facilitaría realizar actualizaciones, correcciones y mejoras en el sistema. Además, la facilidad para realizar pruebas y depuraciones también es importante para mantener la calidad del sistema a lo largo del tiempo.
Interoperatibilidad	**M**	Dado que la expansión a diferentes mercados implica la necesidad de integrarse con otros sistemas y servicios, la **interoperabilidad** es un requerimiento crítico. El sistema debe ser capaz de integrarse de manera efectiva con los sistemas existentes en cada mercado objetivo, así como cumplir con los estándares y protocolos de comunicación necesarios. Esto garantizaría una transferencia fluida de datos y una operación sin problemas entre los diversos sistemas involucrados en las operaciones comerciales y de investigación de la compañía.

Puntos de vista de Contexto

Proyecto	Propiedades los alpes	**ID** 2		**Elaboración**	lunijule	**Versión**	1.0
Vista	Contexto		**Tipo**	C&C			



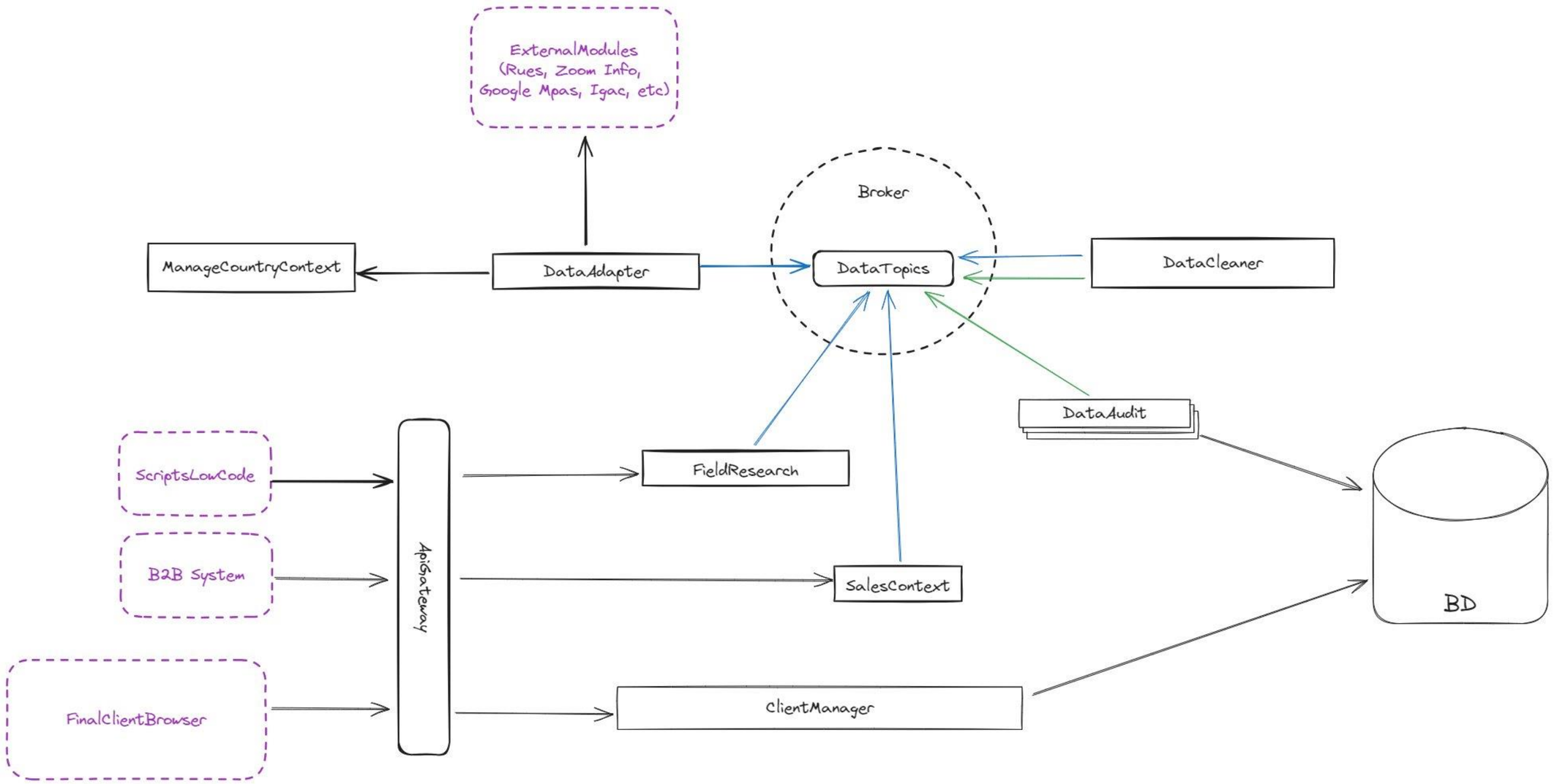
Convención
Ad-hoc
Contexto externo
Contexto interno
App web
App móvil
Usuario institucional
Usuario corporativo

Decisiones de diseño y descripción

- La arquitectura se basa en microservicios para garantizar **la modularidad** y facilitar el despliegue independiente de servicios.
- La arquitectura de microservicios también facilita **la mantenibilidad** del sistema, dado que se pueden escalar los servicios que lo requieran de forma independiente y eficiente.
- El procesamiento asíncrono se implementa para gestionar la ingesta de datos, actualizaciones en tiempo real y flujos de trabajo automatizados, garantizando un rendimiento eficiente.
- La aplicación se conecta con diversas fuentes externas, como el Registro Único Empresarial, Contratos de los Alpes y servicios de propiedades, esto para recopilar información clave.
- Este enfoque en la arquitectura permite a PDA **adaptarse fácilmente a nuevos mercados** y garantizar la calidad, precisión y actualización constante de la información inmobiliaria proporcionada a todos sus usuarios.
- **Puntos de sensibilidad:**
  - En la arquitectura de microservicios y, teniendo en cuenta que la recopilación de datos se realiza por múltiples fuentes, puede generar una dificultad a la hora de tener datos consistentes entre todos los servicios de la aplicación. Este es un desafío que se puede solucionar por medio del **DataCleaner**, especialmente utilizado para mantener orden en este sistema distribuido.
- **Puntos de Tradeoffs:**
  - El tradeoff más notorio con la arquitectura de microservicios es la complejidad operativa que genera el crecimiento de la aplicación frente a independencia de los equipos. A medida que crece la aplicación, nuevos microservicios serán agregados para poder cumplir con normativas y/o regulaciones de nuevos territorios, lo cual hará que haya mayor carga operativa (mantenimiento, implementación, etc..) para el equipo. Por otro lado, esta distribución hace que cada equipo encargado de los nuevos servicios pueda tener eficiencia e independencia a la hora innovar con nuevas funcionalidades.

Puntos de vista Funcional - Módulo

Proyecto	Propiedades los alpes	ID	3	Elaboración	lunijule	Versión	1.0
Vista	Funcional		Tipo	Módulo			



Convención

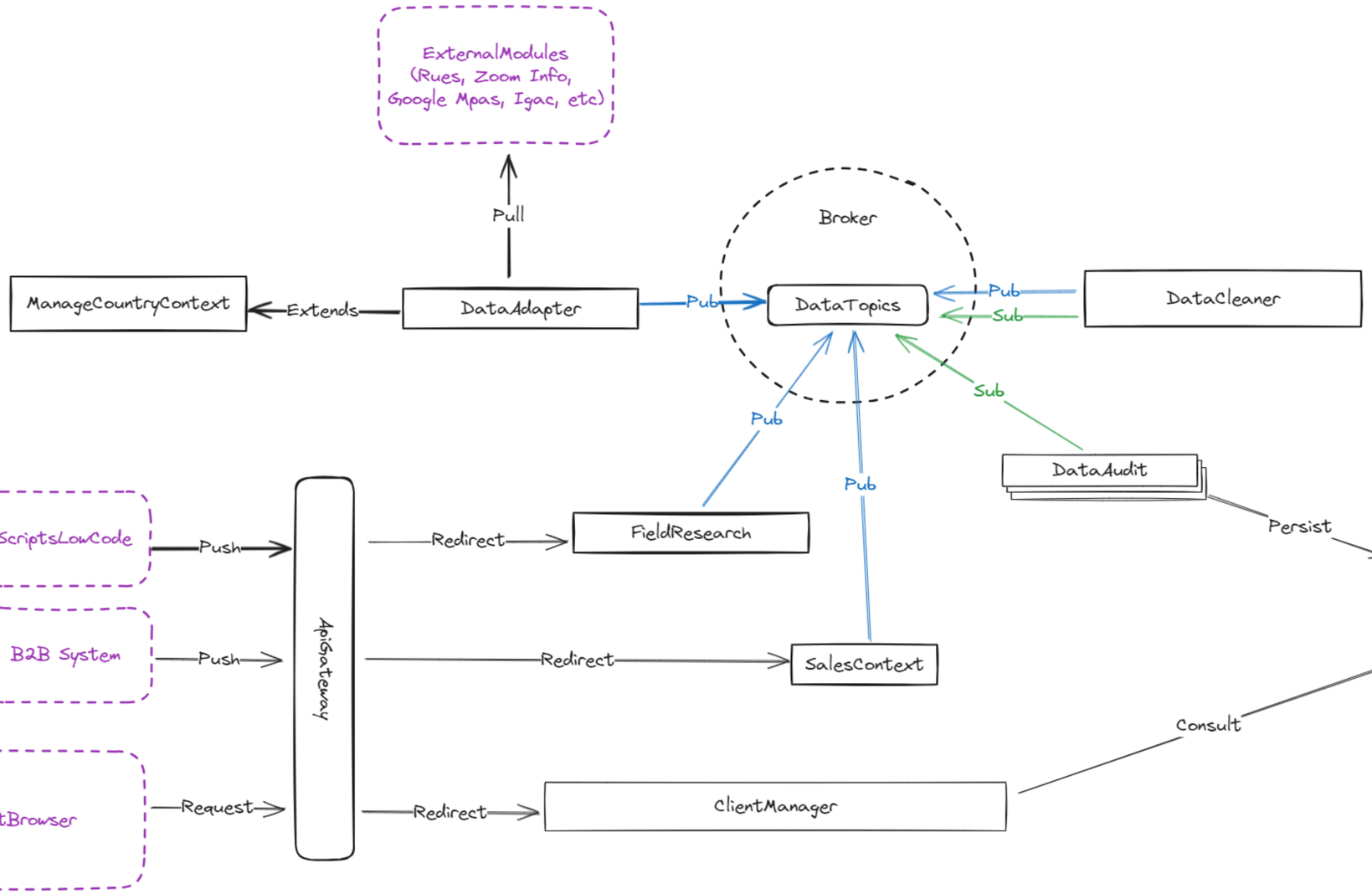
UML 2.0 Modificado

Decisiones de diseño y descripción

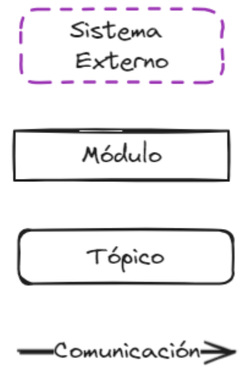
- **Arquitectura modulada (Microservicios):**
- Los microservicios son una excelente elección para modularizar la aplicación en componentes más pequeños y autónomos, lo que facilita la **escalabilidad**, el despliegue y la **interoperatibilidad**.
- **Identificación de un posible cuello de botella en el servicio de DataAudit:**
- Es fundamental identificar y abordar posibles cuellos de botella en los servicios críticos como el de DataAudit. Se deben implementar estrategias de **escalabilidad**, **redundancia** y optimización para garantizar la disponibilidad y el **rendimiento** del módulo. Esto podría incluir la optimización de consultas, la distribución de la carga y el almacenamiento en caché, entre otras técnicas.
- **Utilización de un ApiGateway para redireccionar el tráfico y balancear las cargas:**
- La introducción de un ApiGateway puede mejorar la gestión del tráfico y la seguridad al actuar como un punto de entrada centralizado para los clientes. Además, puede proporcionar capacidades de balanceo de carga para **distribuir** las solicitudes entre los microservicios subyacentes, lo que ayuda a evitar la congestión y mejora la **disponibilidad**

Puntos de vista Funcional – Componente & Conector

Proyecto	Propiedades de los alpes	ID	4	Elaboración	Lunijule	Versión	1.0
Vista	Funcional		Tipo	C&C			



Convención
UML 2.0 Modificado



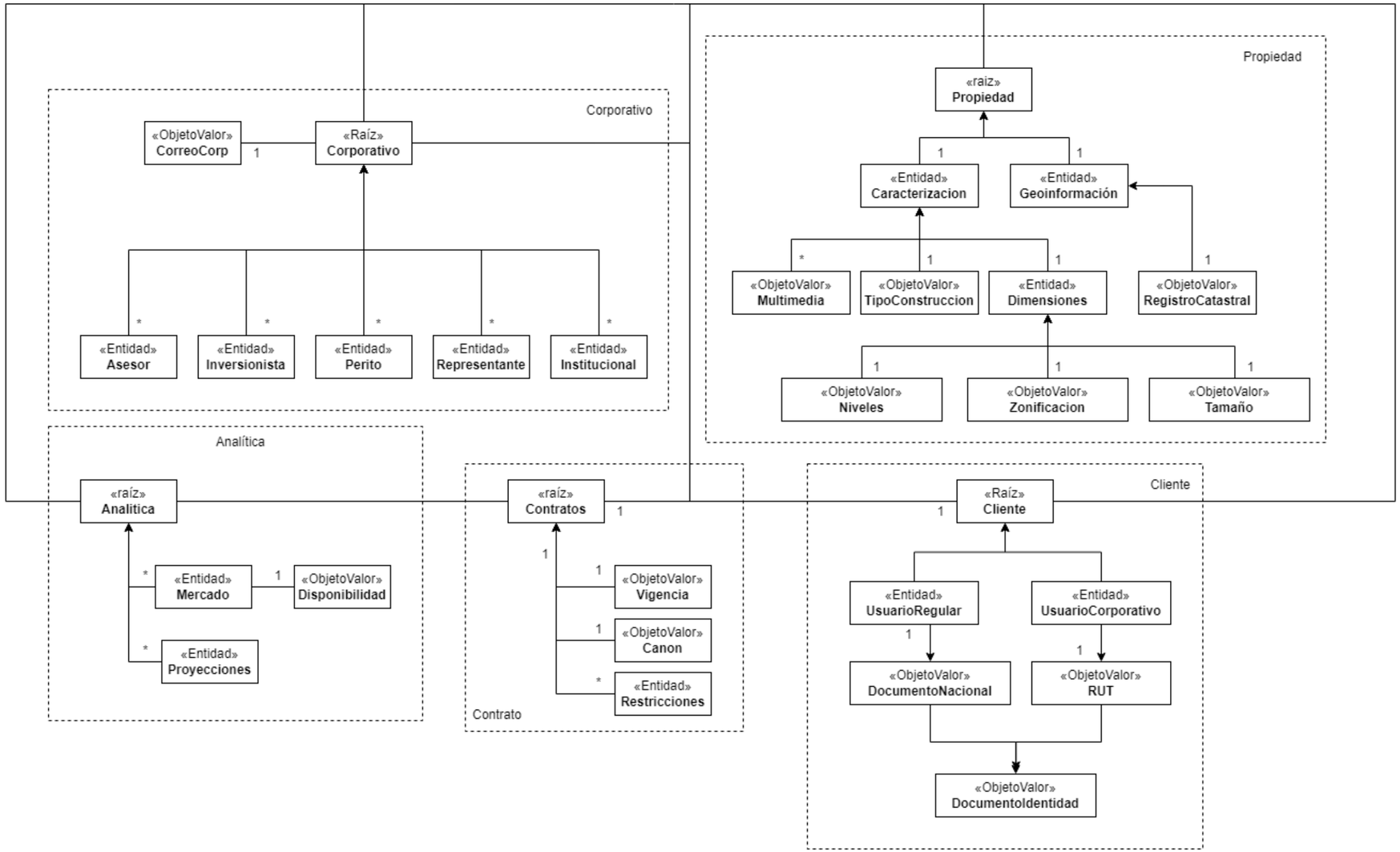
Decisiones de diseño y descripción

- **Comunicación síncrona (SOA) y asíncrona (Pub&Sub):**
- La combinación de comunicación síncrona y asíncrona proporciona flexibilidad en el diseño del sistema. La comunicación síncrona, como la arquitectura orientada a servicios (SOA), puede ser útil para operaciones que requieren respuestas inmediatas, mientras que la comunicación asíncrona, como el patrón de publicación-suscripción (Pub&Sub), es ideal para escenarios donde se necesita desacoplar los componentes y manejar grandes volúmenes de datos.
- Con un enfoque de Microservicios se ve favorecida la **modularidad** para implementar interfaces o adaptadores que permitan la comunicación con servicios externos de los cuales no hay un estándar establecido de comunicación. (**Interoperabilidad**)
- Al manejar una comunicación asíncrona se permite gestionar el flujo de la información a partir de eventos que son desencadenados por actualizaciones de la información.

- **Conectores específicos para comunicación:**
- En el diagrama, los conectores específicos pueden representar los mecanismos de comunicación utilizados entre los microservicios. Por ejemplo, conexiones HTTP para comunicación síncrona o canales de mensajería para comunicación asíncrona.

Puntos de vista de Información

Proyecto	Propiedades de los Alpes	ID		Elaboración		Versión	
Vista	Información		Tipo	Módulo			



Convención
UML
Agregaciones

Decisiones de diseño y descripción

Las agregaciones identificadas son:

- **Corporativo**
- **Propiedad**
- **Cliente**
- **Contrato**
- **Analítica**

- A partir de las agregaciones identificadas se consideran independientes cada una de ellas lo que facilitaría estrategias de persistencia relacionadas con creación, actualización o eliminación sin impactar en la complejidad o rendimiento

- A partir de las prácticas de DDD la identificación de contextos limitados garantiza una coherencia interna y que pueda permitirse evolucionar de manera independiente.

- Las entidades tendrán un identificador único y seguirán un ciclo de vida, mientras que los objetos de valor serán tratados como inmutables y definidos por sus atributos. Esta distinción es fundamental para mantener la integridad y consistencia del dominio.

Puntos de sensibilidad:

- Debemos monitorear continuamente la complejidad de la agregación de Contratos para evitar la sobrecarga de responsabilidades y mantener la facilidad de mantenimiento. Se considerará descomponer esta agregación si su complejidad crece demasiado.

Puntos de tradeoff:

- Mientras aplicamos DDD, debemos equilibrar la complejidad cognitiva con la flexibilidad y modificabilidad del sistema. Esto implica una revisión constante de nuestras prácticas y un enfoque en la simplicidad de diseño donde sea posible.