**DEFINICIÓN DE TRABAJO 2**

**DOCUMENTO 2**

**Materia: Tópicos especiales en telemática**

**Profesor: Edwin Nelson Montoya**

**Estudiantes: Diego Alejandro Pérez**

**Daniel Hoyos Ospina**

**Edwin Montoya Jaramillo**

**Daniela Serna Escobar**

**Fecha: 10 de Septiembre de 2017**

**Universidad EAFIT**

**Contenido del documento 2:**

**1. Síntesis de la fundamentación de los QA Seleccionados (2 a 3 páginas por QA, de forma muy sintética y clara, respondiendo a:**

**Marco de referencia:**

1. **¿Qué es?**
2. **¿Qué patrones se pueden emplear?**
3. **Especificación mediante escenarios**
4. **¿Qué tácticas se pueden emplear?**
5. **Qué herramientas se pueden utilizar para lograrlo**

**2. Análisis: Mediante escenarios y/o propuesta en marco teórico.**

**3. Diseño: En Aplicación y en Sistema.**

1. **Vistas de arquitectura.**
2. **Patrones de arquitectura.**
3. **Best Practices.**
4. **Tácticas.**
5. **Herramientas.**

**Desarrollo**

**1. Síntesis de la fundamentación de los QA Seleccionados (2 a 3 páginas por QA, de forma muy sintética y clara**

**Disponibilidad:**

La disponibilidad de software siempre ha sido un aspecto importante del proceso arquitectónico. La alta disponibilidad es la capacidad de un sistema para ser continuamente disponible para los usuarios sin ninguna pérdida de servicio. Disponibilidad indica el tiempo total en que la aplicación o servicio está disponible para cumplir con los requisitos del usuario final. La alta disponibilidad es fundamental para las operaciones y la continuidad del negocio, así como para alcanzar los objetivos finales de las aplicaciones de software, como los ingresos en línea y la satisfacción del cliente. Para lograr una disponibilidad de cinco nueves (99.999%), el tiempo de inactividad máximo permitido es de 300 s por año. Debido a estos factores, la alta disponibilidad es ahora un requisito crítico e "imprescindible" para la mayoría de las aplicaciones empresariales. También afecta la lealtad del cliente y mejora la ventaja competitiva.

**Patrones de alta disponibilidad**

* Failover
* Failback
* Replication: Active replication, Passive replication
* Redundancy
* Virtualization
* Continuous maintenance: Corrective maintenance, Preventive maintenance
* Perfective maintenance

**Patrones de alta disponibilidad del software**

* Graceful and stepwise functionality degradation pattern
* Asynchronous and services-based integration with external interfaces
* Stateless and lightweight application components
* Continuous incremental code and data replication
* Availability trade-off using the CAP theorem
* No Consistency but required Availability and Partition Tolerance
* No Partition Tolerance but required Availability and Consistency
* No Availability but required Partition Tolerance and Consistency

**Escenarios:**

1. Fuente del estímulo. Distinguimos entre orígenes internos y externos de fallas porque la respuesta del sistema deseada puede ser diferente.
2. Estímulo: Se produce una falla de una de las clases siguientes

* Omisión. Un componente no responde a una entrada.
* Choque. El componente sufre repetidamente fallos de omisión.
* Sincronización. Un componente responde pero la respuesta es temprana o tardía.
* Respuesta. Un componente responde con un valor incorrecto.

1. Artefacto. Esto especifica el recurso que se requiere que esté altamente disponible, como un procesador, un canal de comunicación, un proceso o un almacenamiento.
2. Ambiente. El estado del sistema cuando el fallo ocurre también puede afectar la respuesta del sistema deseada. Por ejemplo, si el sistema ya ha visto algunos fallos y está funcionando en otro modo que el normal, puede ser deseable apagarlo totalmente. Sin embargo, si este es el primer fallo observado, puede preferirse alguna degradación del tiempo o función de respuesta.
3. Respuesta. Hay una serie de posibles reacciones a una falla del sistema. Primero, la falla debe ser detectada y aislada (correlacionada) antes de que cualquier otra respuesta sea posible. (Una excepción a esto es cuando se evita el fallo antes de que se produzca.) Después de que se detecta el fallo, el sistema debe recuperarse de él. Las acciones asociadas a estas posibilidades incluyen registrar el fallo, notificar a usuarios seleccionados u otros sistemas, tomar acciones para limitar el daño causado por la falla, cambiar a un modo degradado con menos capacidad o menos función, apagar sistemas externos o quedar indisponible durante reparar.
4. Medida de respuesta. La medida de respuesta puede especificar un porcentaje de disponibilidad, o puede especificar un tiempo para detectar el fallo, el tiempo para reparar el fallo, los tiempos o intervalos de tiempo durante los cuales el sistema debe estar disponible, o la duración para la cual el sistema debe estar disponible.

**Tácticas:**

1. Detectar Fallos

Antes de que cualquier sistema pueda tomar acción con respecto a un fallo, la presencia del fallo debe ser detectada o anticipada. Las tácticas de esta categoría incluye las siguientes

* Ping/echo
* Monitor
* Heartbeat
* Time stamp
* Sanity checking
* Condition monitoring
* Voting: Replication, Functional redundancy, Analytic redundancy
* Exception detection: System exceptions,The parameter fence tactic,Parameter typing,Timeout
* Self-test

1. Recuperación de fallos

Las tácticas de recuperación de fallas se refinan en tácticas de preparación y reparación y tácticas de reintroducción. Estos últimos se ocupan de reintroducir un componente fallido (pero rehabilitado) de nuevo en funcionamiento normal.

Tácticas de preparación y reparación:

* Active redundancy (hot spare)
* Passive redundancy (warm spare)
* Spare (cold spare)
* Exception handling
* Rollback
* Software upgrade
* Retry
* Ignore faulty behavior
* The degradation
* Reconfiguration

Reintroducción:

* The shadow
* State resynchronization
* Escalating restart
* Non-stop forwarding (NSF)

1. Prevenir fallas

En lugar de detectar fallos y luego tratar de recuperarse de ellos, ¿qué pasa si su sistema podría evitar que ocurra en primer lugar?

* Removal from service
* Transactions
* Predictive model
* Exception prevention
* Increase competence set

**Rendimiento:**

Se trata de tiempo y la capacidad del sistema de software para cumplir con los requisitos de tiempo. Cuando ocurren eventos -interrupciones, mensajes, peticiones de usuarios u otros sistemas, o eventos de reloj que marcan el paso del tiempo- el sistema, o algún elemento del sistema, debe responder a ellos en el tiempo. Caracterizar los eventos que pueden ocurrir (y cuándo pueden ocurrir) y la respuesta basada en el tiempo del sistema o del elemento a esos eventos es la esencia que está discutiendo el rendimiento.

Los eventos del sistema basados ​​en Web vienen en forma de solicitudes de usuarios (numeración en decenas y decenas de millones) a través de sus clientes, como los navegadores web. En un sistema de control para un motor de combustión interna, los eventos provienen de los controles del operador y del paso del tiempo; el sistema debe controlar tanto el encendido del encendido cuando un cilindro está en la posición correcta y la mezcla del combustible para maximizar la potencia y la eficiencia y reducir al mínimo la contaminación.

WPO está esencialmente relacionado con los métodos y técnicas para optimizar la velocidad de las páginas web, lo que implica analizar todos los componentes de la página web para optimizar el tiempo de respuesta de la página. Esto también implica varios otros factores que afectan directa o indirectamente a la velocidad, como componentes de infraestructura, componentes de supervisión y mantenimiento, estrategia de gestión de contenido y el marco de almacenamiento en caché.  
  
El rendimiento de la página web se considera un requisito crítico y no funcional. Tiene un gran impacto en los siguientes factores:

* Ventaja competitiva: A medida que aumentan las expectativas de velocidad de los usuarios en línea, los sitios web más lentos se perderán frente a la competencia.
* Experiencia del usuario: El rendimiento es el principal factor que influye en la experiencia general de los usuarios. Cuanto mejor sea la velocidad, más leales serán los clientes.
* Uso del sitio: los clientes sólo se adhieren a los sitios más rápidos y receptivos; si el sitio es más lento, cambiarán a los sitios de los competidores. Además, la mayoría de los motores de búsqueda ahora están factorizando la velocidad de la página para la clasificación de las páginas, y por lo tanto la velocidad también afecta el tráfico enrutado desde los motores de búsqueda.
* Impacto financiero: el ratio de conversión (relación visitante-comprador) se ve afectado si la página no se carga rápidamente; esto demuestra el impacto directo de la velocidad de la página en los ingresos en línea.
* Estrategia omnidireccional: una página web óptima se adapta mejor a la representación multidispositivo.

El rendimiento suele estar relacionado con la escalabilidad, es decir, aumentar la capacidad de trabajo de su sistema, mientras sigue funcionando bien. Técnicamente, la escalabilidad hace que su sistema sea fácil de cambiar de una manera particular, y también lo es una especie de modificabilidad.

**Especificación de Escenario**

1. Fuente del estímulo: Los estímulos llegan de fuentes externas (posiblemente múltiples) o internas.
2. Estímulo: Los estímulos son las llegadas del evento. El patrón de llegada puede ser periódico, estocástico o esporádico, caracterizado por parámetros numéricos.
3. Artefacto: El artefacto es el sistema o uno o más de sus componentes.
4. Ambiente: El sistema puede estar en varios modos operativos, tales como normal, emergencia, carga de pico o sobrecarga.
5. Respuesta: El sistema debe procesar los eventos que llegan. Esto puede causar un cambio en el entorno del sistema (por ejemplo, desde el modo normal al sobrecargado).
6. Medida de respuesta: Las medidas de respuesta son el tiempo que se tarda en procesar los eventos que llegan (latencia o plazo), la variación en este tiempo (jitter), el número de eventos que se pueden procesar dentro de un intervalo de tiempo determinado (throughput) o una caracterización de los eventos que no pueden ser procesados (tasa de errores).

**Tácticas**

1. Control de demanda de recursos:

Una manera de aumentar el rendimiento es administrar cuidadosamente la demanda de recursos. Esto puede hacerse reduciendo el número de eventos procesados aplicando una tasa de muestreo o limitando la velocidad a la que el sistema responde a los eventos. Además, hay una serie de técnicas para asegurar que los recursos que tiene son aplicados con juicio:

* Controlar la demanda de recursos.
* Administrar recursos.
* Administrar la frecuencia de muestreo.
* Limitar la respuesta al evento
* Priorizar eventos.
* Reduzca los gastos generales.
* Tiempos de ejecución vinculados.
* Administrar recursos

1. Administración de recursos:

Incluso si la demanda de recursos no es controlable, la gestión de estos recursos puede ser. A veces un recurso puede ser intercambiado por otro. Por ejemplo, los datos intermedios pueden mantenerse en un caché o pueden regenerarse dependiendo de la disponibilidad de recursos de tiempo y espacio. Esta táctica se aplica generalmente al procesador pero también es eficaz cuando se aplica a otros recursos como un disco. Estas son algunas tácticas de administración de recursos:

* Aumentar los recursos.
* Introducir la concurrencia.
* Mantener múltiples copias de los cálculos.
* Mantener múltiples copias de datos.
* Tamaños de cola enlazados.
* Programar recursos.

**Seguridad:**

La seguridad es una medida de la capacidad del sistema de proteger los datos y la información de accesos no autorizados al mismo tiempo que proporciona acceso a las personas y los sistemas que están autorizados. Una acción contra un sistema computacional con la intención de causar daño, es llamada ataque y puede tomar numerosas formas. Este puede ser un intento no autorizado de acceder a los datos o servicios o para modificar datos, o puede ser un intento de denegar los servicios a usuarios legítimos.

El enfoque más simple para caracterizar la seguridad tiene 3 características (CIA):

1. Confidencialidad: es la propiedad de que los datos o servicios están protegidos de accesos no autorizados.
2. Integridad: es la propiedad de que los datos o servicios no están sujetos a manipulaciones no autorizadas.
3. Disponibilidad: es la propiedad de que el sistema estará disponible para uso legítimo.

otras características que son usadas para soportar CIA son estas:

1. Autenticación: verifica la identidad de las partes en una transacción y verifica si ellos realmente son quienes dicen ser .
2. No rechazo: garantiza que el remitente de un mensaje no puede negar más tarde el envío del mensaje y que el destinatario no puede negar haber recibido el mensaje.
3. Autorización: le garantiza al usuario los privilegios de llevar a cabo una tarea.

Los enfoques para lograr la seguridad pueden caracterizarse como aquellos que detectan ataques, resisten ataques, reaccionan a los ataques y los que se recuperan de ataques exitosos. Los objetos que están siendo protegidos de los ataques son datos en reposo, datos en tránsito y procesos computacionales.

**Especificación mediante escenarios:**

Partes del escenario:

1. Fuente de estímulo: El origen del ataque puede ser un humano u otro sistema. Puede haber sido previamente identificado o puede ser actualmente desconocido. Un atacante humano desde afuera de la organización o desde adentro.
2. Estímulo: El estímulo es un ataque. Caracterizamos esto como un intento no autorizado de mostrar , cambiar o eliminar datos, acceder a los servicios del sistema o reducir su disponibilidad.
3. Artefacto: El objetivo del ataque puede ser los servicios del sistema, la información que hay en el, o los datos producidos o consumidos por el sistema.
4. Ambiente: El ataque puede ocurrir cuando el sistema esté en línea o no, conectado o desconectado de la red, detrás de un firewall o abierto a la red, completamente operacional, parcialmente operacional o no operacional.
5. Respuesta: El sistema debe garantizar que las transacciones se efectúen de manera que los datos o servicios estén protegidos contra el acceso no autorizado; datos o servicios no están siendo manipulados sin autorización; las partes en una transacción se identifican con garantía; las partes en la transacción no pueden rechazar sus compromisos; y los datos, recursos y servicios del sistema estarán disponibles para uso legítimo.

El sistema también debe rastrear las actividades dentro de él, mediante registro de acceso o modificación; intentos de acceso a datos, recursos o servicios; y notificar a las entidades apropiadas (personas o sistemas) cuando se produce un ataque aparente.

1. Medida de respuesta: incluyen cuánto de un sistema se ve comprometido cuando un determinado componente o valor de datos se ve comprometido, cuanto tiempo paso antes de que un ataque fuera detectado, cuantos ataques fueron resistidos, cuanto tiempo tomo para recuperarse de un ataque exitoso, y cuantos datos eran vulnerables a un ataque en particular.

**Tácticas para la seguridad**

1. Detectar ataques: la categoría de detectar ataques consiste en cuatro tácticas:
2. Detectar intrusión.
3. Detectar negación de servicios.
4. Verificar la integridad del mensaje
5. Detectar retraso del mensaje
6. Resistir ataques: Hay una serie de bien conocidos medios de resistencia a un ataque:
7. Identificar actores
8. Autenticar actores
9. Autorizar actores
10. Limitar acceso
11. Limitar la exposición
12. Encriptar los datos
13. Entidades separadas
14. Cambiar ajustes por defecto
15. Reaccionar a ataques: varias tácticas se han creado para responder a un potencial ataque:
16. Revocar acceso.
17. Bloquear computador
18. informar a los actores
19. Recuperarse de ataques: una vez que un sistema ha detectado e intentado resistir a un ataque, este necesita recuperarse. Parte de la recuperación es la restauración de los servicios.

Además de las tácticas de disponibilidad que permiten la restauración de los servicios, debemos mantener una pista de auditoria. Revisamos para ayudar a rastrear las acciones de, e identificar, a un atacante. Podemos analizar pistas de auditoria para intentar procesar a los atacantes, o para crear mejores defensas en el futuro

**2. Análisis: Mediante escenarios y/o propuesta en marco teórico.**

***Disponibilidad*:**

**Escenario 1:**

* **Fuente:** Heartbeat.
* **Estimulo:** Servidor de aplicación 1 no responde.
* **Artefacto:** Balanceador de cargas.
* **Ambiente:** Normal.
* **Respuesta:** Aislar el servidor malo y redirigir todo el tráfico al otro servidor de aplicación 2.
* **Medida de Respuesta:** No *downtime*.

**Escenario 2:**

* **Fuente:** Heartbeat.
* **Estimulo:** Servidor de aplicación 1 caido vuelve a estar disponible.
* **Artefacto:** Balanceador de cargas.
* **Ambiente:** Servidor de aplicación 1 caido.
* **Respuesta:** Reincorporar el servidor de aplicación 1 y redistribuir las cargas.
* **Medida de Respuesta:** No *downtime.*

***Rendimiento*:**

**Escenario 1:**

* **Fuente:** Usuarios
* **Estimulo:** Baja el rendimiento debido al exceso de pedidos al servidor de aplicación 1.
* **Artefacto:**  Balanceador de cargas.
* **Ambiente:** Normal.
* **Respuesta:** El balanceador de cargas comienza a redirigir tráfico hacia el servidor de aplicación 2.
* **Medida de Respuesta:** Los usuario notan una mejoría de hasta un minuto menos en el tiempo de descarga.

***Seguridad*:**

**Escenario 1:**

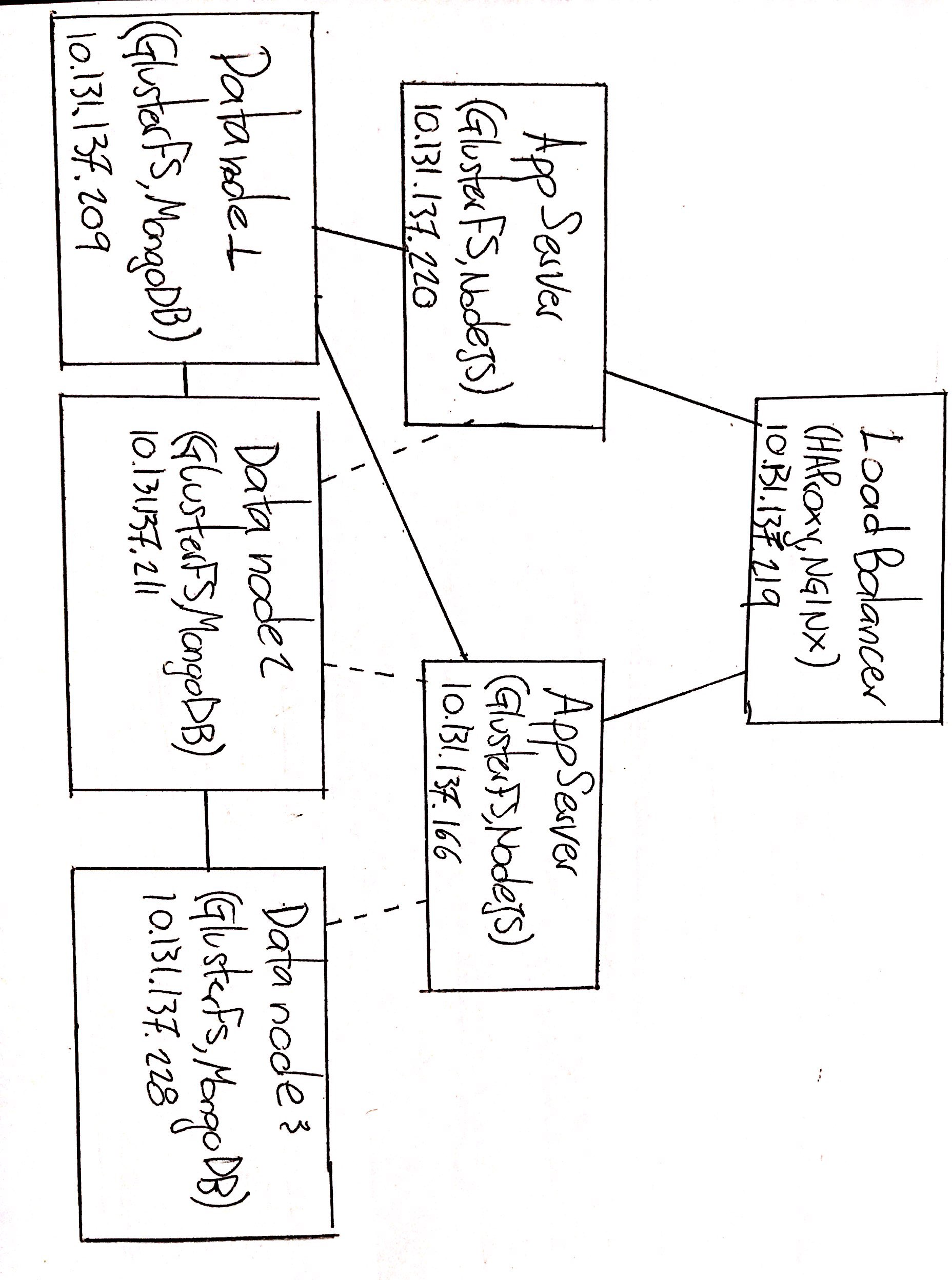
* **Fuente:** Usuario no autorizado.
* **Estimulo:** Intento de logueo sin un usuario ya registrado en google.
* **Artefacto:** Servidores de Aplicación.
* **Ambiente:** Normal.
* **Respuesta:** La aplicación no lo deja ingresar y no muestra datos privados de los usuarios.
* **Medida de Respuesta:** El usuario no autorizado no logra ingresar a la aplicación.

**Escenario 2:**

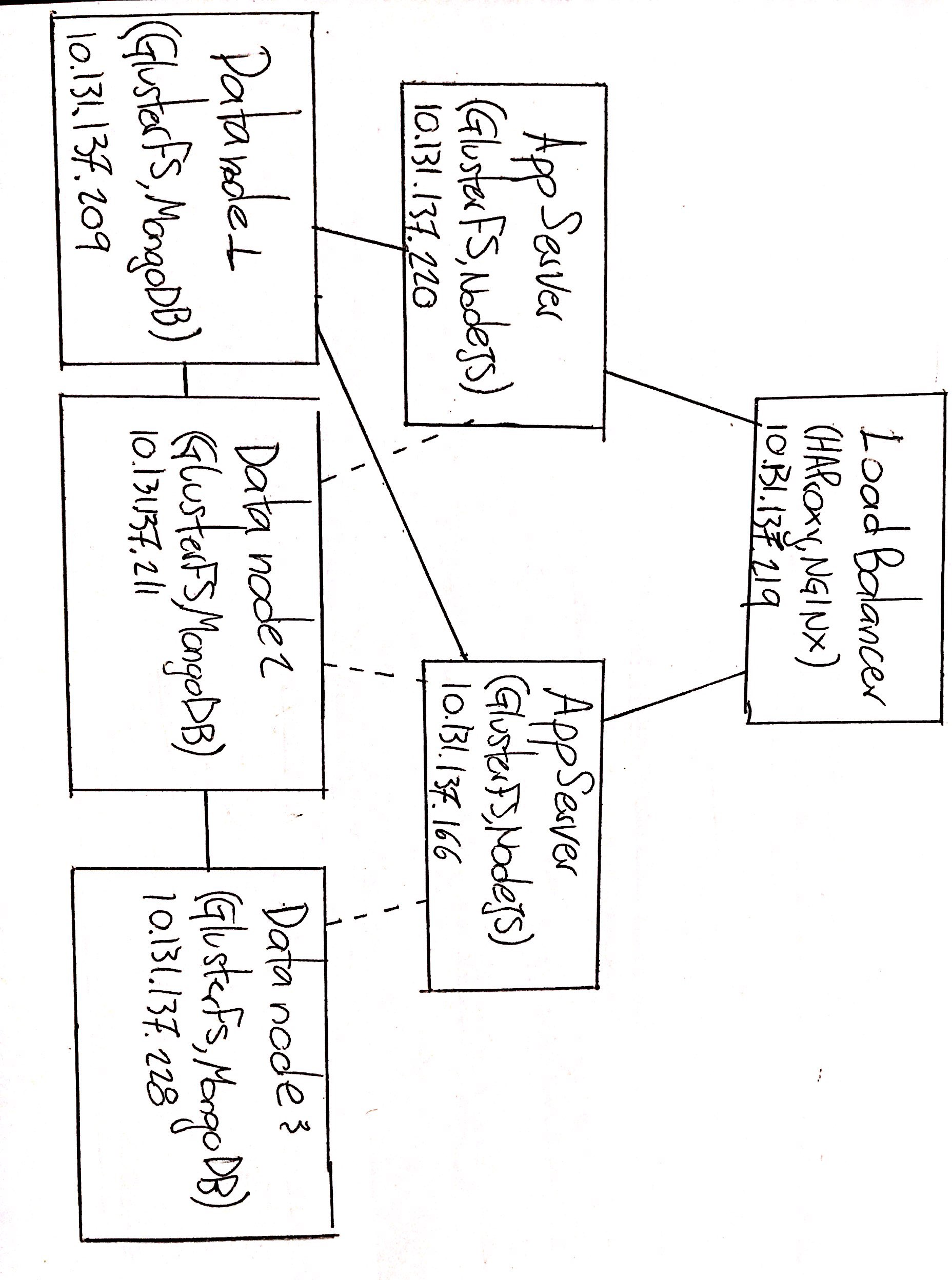
* **Fuente:** Usuario no autorizado.
* **Estimulo:**  Busca una pelicula privada que no le pertenece.
* **Artefacto:** Servidores de aplicación.
* **Ambiente:** Normal.
* **Respuesta:** El sistema no le devuelve ningún resultado.
* **Medida de Respuesta:** El usuario no autorizado no logra ver el contenido.

**3. Diseño: En Aplicación y en Sistema.**

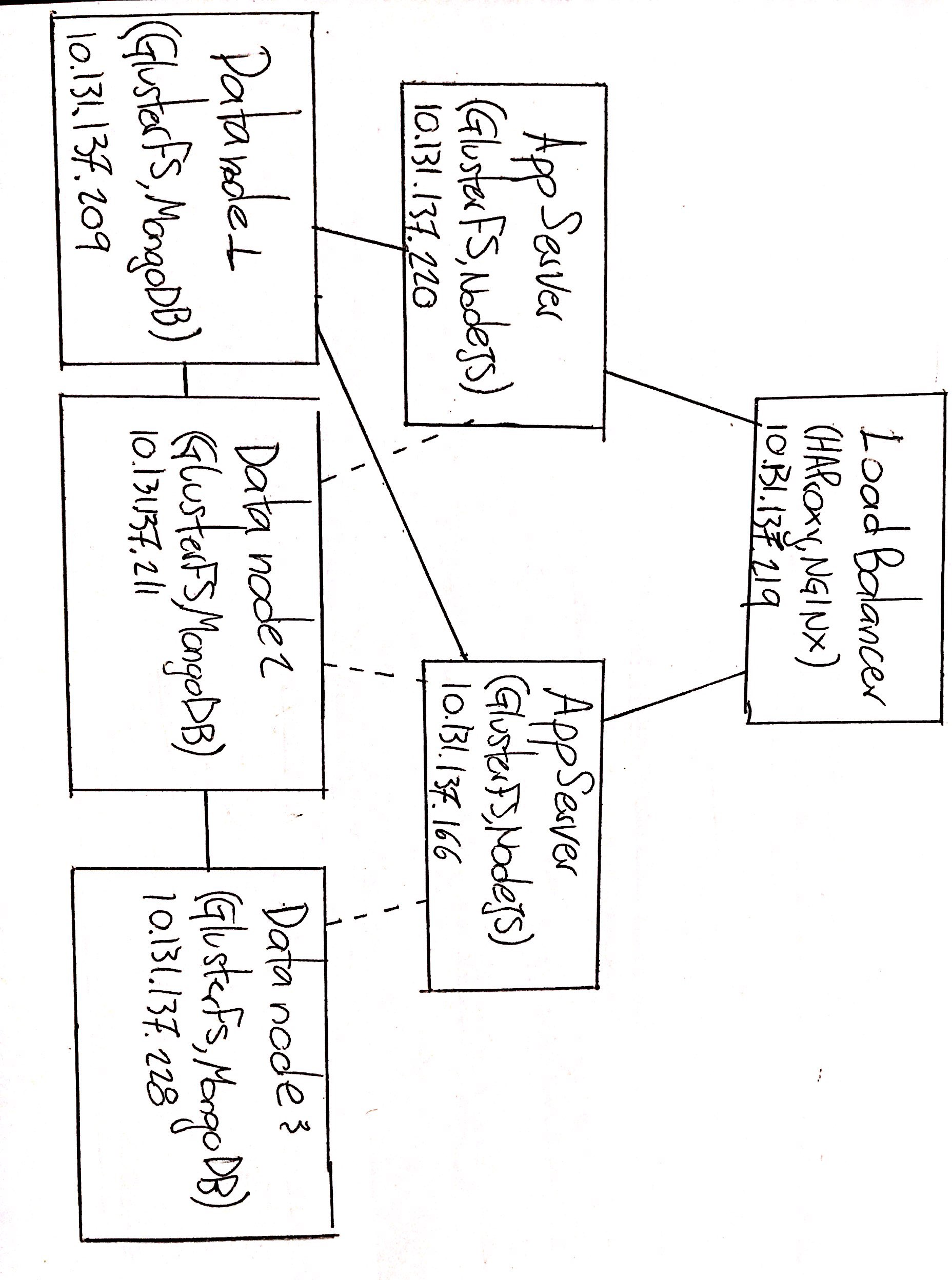
**Disponibilidad:**

1. **Vistas de arquitectura.**
2. **Patrones de arquitectura.**
   1. Failover
   2. Failback
   3. Replication: Active replication, Passive replication
   4. Redundancy
   5. Virtualization
   6. Continuous maintenance: Corrective maintenance, Preventive maintenance
   7. Perfective maintenance
3. **Best Practices.**
   1. Graceful and stepwise functionality degradation pattern
   2. Asynchronous and services-based integration with external interfaces
   3. Stateless and lightweight application components
   4. Continuous incremental code and data replication
   5. Availability trade-off using the CAP theorem
   6. No Consistency but required Availability and Partition Tolerance
   7. No Partition Tolerance but required Availability and Consistency
4. **Tácticas.**
   1. Detectar Fallos
      1. Ping/echo
      2. Monitor
      3. Heartbeat
      4. Time stamp
      5. Sanity checking
      6. Condition monitoring
      7. Voting: Replication, Functional redundancy, Analytic redundancy
      8. Exception detection: System exceptions,The parameter fence tactic,Parameter typing,Timeout
      9. Self-test
   2. Recuperación de fallos
      1. Active redundancy (hot spare)
      2. Passive redundancy (warm spare)
      3. Spare (cold spare)
      4. Exception handling
      5. Rollback
      6. Software upgrade
      7. Retry
      8. Ignore faulty behavior
      9. The degradation
      10. Reconfiguration
   3. Reintroducción:
      1. The shadow
      2. State resynchronization
      3. Escalating restart
      4. Non-stop forwarding (NSF)
   4. Prevenir fallas:
      1. Removal from service
      2. Transactions
      3. Predictive model
      4. Exception prevention
      5. Increase competence set
5. **Herramientas.**
   1. Detectar Fallos, Recuperación de fallos y Reintroducción:
      1. HAProxy
      2. GlusterFS
      3. MongoDB Replication Set

**Rendimiento:**

1. **Vistas de arquitectura.**
2. **Patrones de arquitectura.**
   1. Control de demanda de recursos:
      1. Controlar la demanda de recursos.
      2. Administrar recursos.
      3. Administrar la frecuencia de muestreo.
      4. Limitar la respuesta al evento
      5. Priorizar eventos.
      6. Reduzca los gastos generales.
      7. Tiempos de ejecución vinculados.
      8. Administrar recursos
   2. Administración de recursos:
      1. Aumentar los recursos.
      2. Introducir la concurrencia.
      3. Mantener múltiples copias de los cálculos.
      4. Mantener múltiples copias de datos.
      5. Tamaños de cola enlazados.
      6. Programar recursos.
3. **Best Practices.**
   1. Graceful and stepwise functionality degradation pattern
   2. Asynchronous and services-based integration with external interfaces
   3. Stateless and lightweight application components
   4. Continuous incremental code and data replication
   5. Partition trade-off using the CAP theorem
   6. No Consistency but required Availability and Partition Tolerance
   7. No Partition Tolerance but required Availability and Consistency
4. **Tácticas.**
   1. WPO está esencialmente relacionado con los métodos y técnicas para optimizar la velocidad de las páginas web, lo que implica analizar todos los componentes de la página web para optimizar el tiempo de respuesta de la página. Esto también implica varios otros factores que afectan directa o indirectamente a la velocidad, como componentes de infraestructura, componentes de supervisión y mantenimiento, estrategia de gestión de contenido y el marco de almacenamiento en caché.
5. **Herramientas.**
   1. Control de demanda de recursos:
      1. HAProxy
   2. Administración de recursos:
      1. GlusterFS

**Seguridad:**

1. **Vistas de arquitectura.**
2. **Patrones de arquitectura.**
   1. Confidencialidad: es la propiedad de que los datos o servicios están protegidos de accesos no autorizados.
   2. Integridad: es la propiedad de que los datos o servicios no están sujetos a manipulaciones no autorizadas.
   3. Disponibilidad: es la propiedad de que el sistema estará disponible para uso legítimo.
3. **Best Practices.**
   1. Autenticación: verifica la identidad de las partes en una transacción y verifica si ellos realmente son quienes dicen ser .
   2. No rechazo: garantiza que el remitente de un mensaje no puede negar más tarde el envío del mensaje y que el destinatario no puede negar haber recibido el mensaje.
   3. Autorización: le garantiza al usuario los privilegios de llevar a cabo una tarea.
4. **Tácticas.**
   1. Detectar ataques: la categoría de detectar ataques consiste en cuatro tácticas:
      1. Detectar intrusión.
      2. Detectar negación de servicios.
      3. Verificar la integridad del mensaje
      4. Detectar retraso del mensaje
   2. Resistir ataques: Hay una serie de bien conocidos medios de resistencia a un ataque:
      1. Identificar actores
      2. Autenticar actores
      3. Autorizar actores
      4. Limitar acceso
      5. Limitar la exposición
      6. Encriptar los datos
      7. Entidades separadas
      8. Cambiar ajustes por defecto
   3. Reaccionar a ataques: varias tácticas se han creado para responder a un potencial ataque:
      1. Revocar acceso.
      2. Bloquear computador
      3. informar a los actores
   4. Recuperarse de ataques: una vez que un sistema ha detectado e intentado resistir a un ataque, este necesita recuperarse. Parte de la recuperación es la restauración de los servicios.
5. **Herramientas.**
   1. Resistir Ataques:
      1. Autenticación contra Open ID
      2. Encriptar el tráfico http mediante https