****

**DISEÑO DE SISTEMAS**

***Ingeniería en Sistemas de Información***

**TRABAJO PRÁCTICO**

***Guía Nro. 3***

**“Arquitectura”**

**> Equipo Nro. 2 <**

**Ahmed, Farid 155946-1**

**Chaín, Ezequiel 160065-5**

**Galvan, Santiago 147569-1**

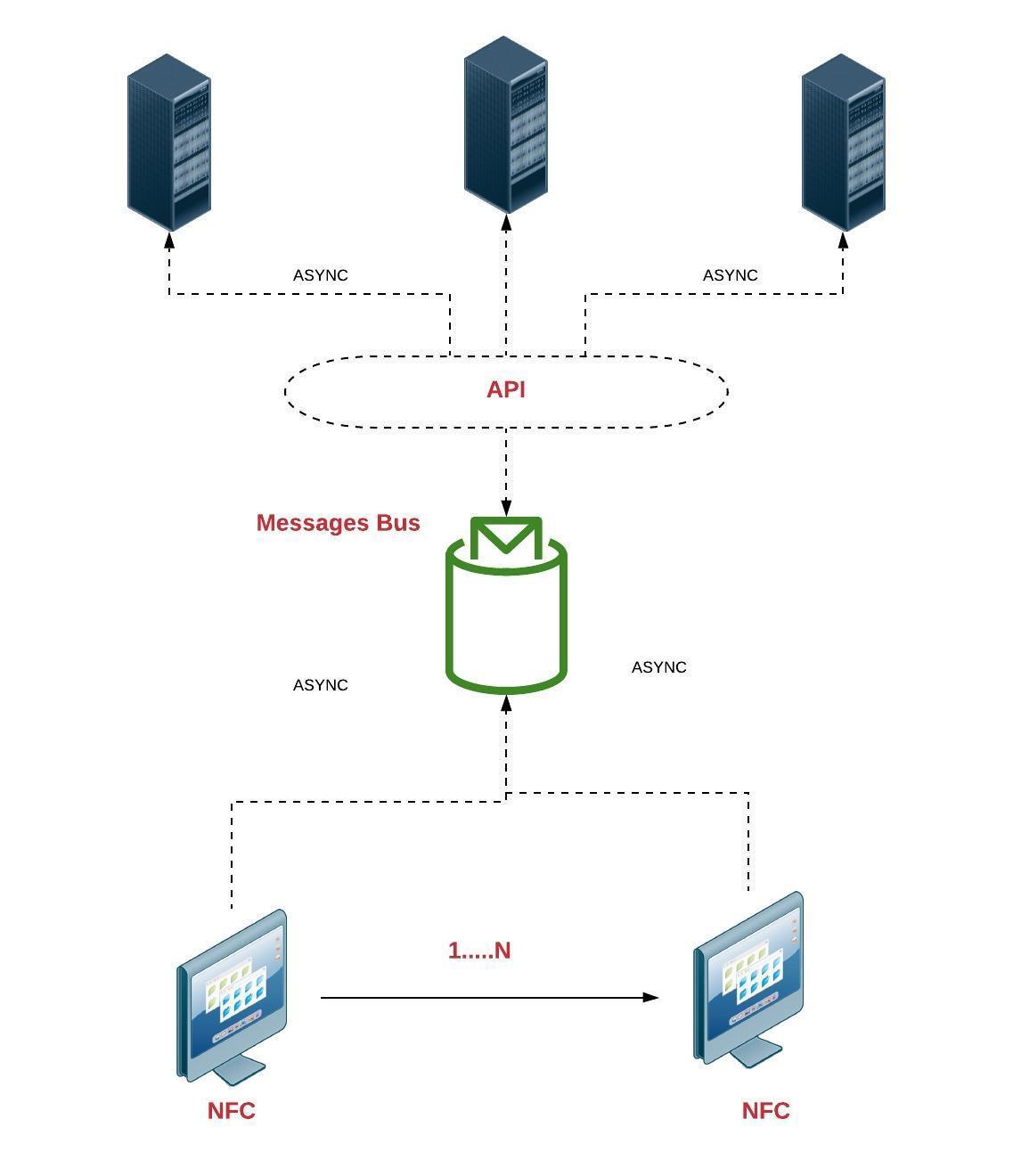
**Sanchez, Etienne 152852-0**

**Virgili, Jose Maria 147264-1**

**Curso: K3051**

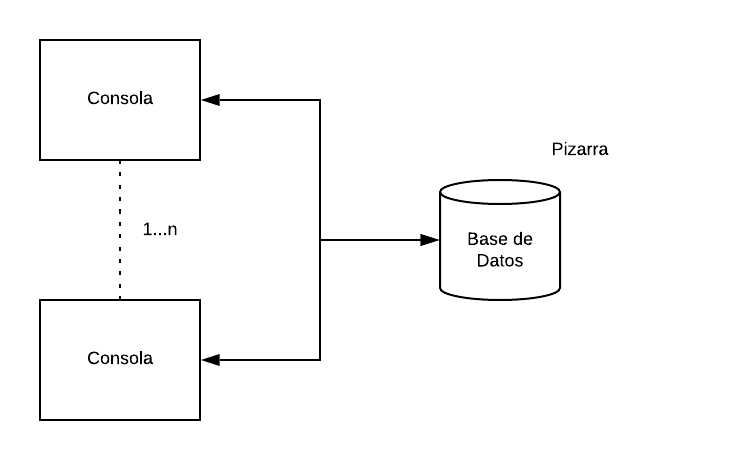
**Ejercicio 1:** NFC

1. Para esta solución se ha seleccionado un estilo Cliente-Servidor. Se puede considerar un Cliente-Servidor de 3 capas o uno de 2 capas más una API como adaptador para diferentes tipos de requests y tipos de datos. Se selecciona este estilo debido a que el sistema consiste simplemente de N clientes enviando requests hacia un servidor, sin consumir servicios específicos.
2. Para la integración, los patrones recomendados para fusionarlos con el estilo Cliente-Servidor (ya que por si mismo no satisface los requerimientos) son los de Eventos y Bus de mensajes. Estos son los seleccionados debido a la capacidad de poder actuar asincrónicamente como el enunciado lo pide (“no quedarse bloqueada a la espera de una confirmación”) y también debido a la alta transferencia de datos, frecuencia y alta capacidad de procesamiento.
3. Los atributos de calidad mas relevantes de esta especificación se consideran Disponibilidad y Rendimiento. Para la Disponibilidad se ha elegido comunicación asincrónica para que los clientes nunca se queden bloqueados esperando respuesta y pueda seguir funcionando. Para el rendimiento se han elegido esos patrones de integración debido al alto rendimiento, frecuencia, y procesamiento de datos que brindan. Y como tercer atributo de calidad también tenemos la Integración, que no deja de ser importante debido al requerimiento de poder aceptar distintos fabricantes de terminales y soportar múltiples formatos de datos, por ende se ha seleccionado trabajar con una API Rest para poder funcionar comúnmente para múltiples clientes y que pueda normalizar los tipos de datos.



**Ejercicio 2:** Tráfico aéreo

1. Para esta solución se ha seleccionado una arquitectura de centralización de datos. Un estilo como el Pizarra, ya que el objetivo principal es la consistencia de datos.
2. Para la integración, será la de base de datos compartida, ya que el usuario está dispuesto a que haya bloqueos para la sincronización con la imagen del servidor, será enviando una consulta al servidor.
3. Los atributos de calidad que destacan de esta especificación son la integridad y escalabilidad y ya que la solución es desarrollada por el mismo fabricante, no debería haber problema con estructuras de datos, ya que es lo que acomplejaría al implementar este tipo de arquitectura

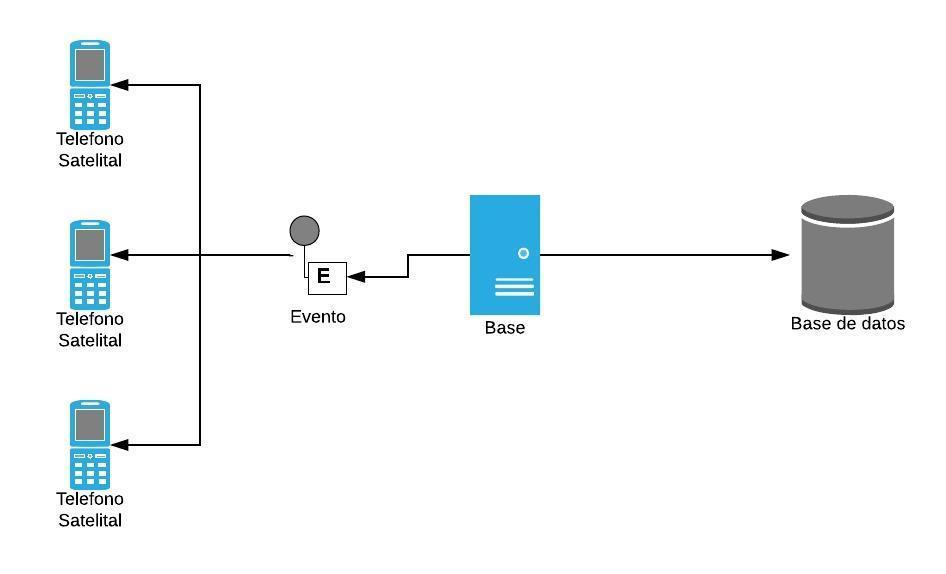


**Ejercicio 3:** Marina Mercante

1. Para esta solución usamos la arquitectura Cliente-Servidor, ya que vamos a tener un servidor al que se van a conectar múltiples clientes simultáneamente. Además pensamos usar una arquitectura de tres capas siendo la primera utilizada para el acceso del cliente, la segunda para la lógica de negocio, la cual beneficiará gestionará la concurrencia de clientes, y la tercera capa para el acceso de datos.
2. El patrón de integración que recomendamos usar es el de eventos ya que varios subsistemas van a procesar los mismos eventos en tiempo real asincrónicamente ya que necesitan la respuesta de estos eventos si o si para continuar su ejecución.
3. Los atributos de calidad más relevantes que definen a la solución son:

Conexiones concurrentes: Ya que el sistema necesita múltiples conexiones de clientes en simultáneo.

Integración: Ya que el sistema debe garantizar la integridad para las mediciones y los informes de los buques.



**Ejercicio 4:** FinTech

1. Para esta solución se ha seleccionado una arquitectura orientado a servicios.Ya que dependemos de sistemas externos, que son servicios que vamos a consumir.  
   Se deberá implementar una capa de presentación para la interacción con el usuario y una capa de servicios principalmente en el cual estará definido la interacción con los diferentes servicios. A su vez, podemos crear nuestros propios servicios, que pueden ser reemplazables en un futuro.
2. El patrón de integración que se debería usar es el Call and Return, por medio de un servicio web, de tipo REST o SOAP por medio del protocolo HTTP, ya que no tiene problemas con el firewall. En lugar de usar .NET (en caso de usar C#) o RMI (Java), ya que tienen alto acoplamiento y son de tipo binario y están orientado a objetos. En cambio SOAP o REST utilizan documentos como JSON o XML y no necesitan saber el IP del servidor.
3. Los atributos de calidad que destacan de esta especificación son flexibilidad y escalabilidad. Ya que son adaptables ante cambio funcional y que la implementación de un nuevo servicio no es compleja.