Taller de Sistemas de Información
Java

Iteración 2
Integración con Sistemas externos

Introducción.

El siguiente documento incluye los detalles de la Iteración 2, a realizar en la materia Taller Java.

La lógica de negocio implementada hasta el momento, solo es de utilidad cuando se expone al exterior para ser “consumida” por diferentes actores de nuestro Sistema.

Además, nuestro sistema necesita integrarse con sistemas externos para poder realizar las tareas asignadas.

La integración de sistemas es un aspecto importante de las aplicaciones empresariales.

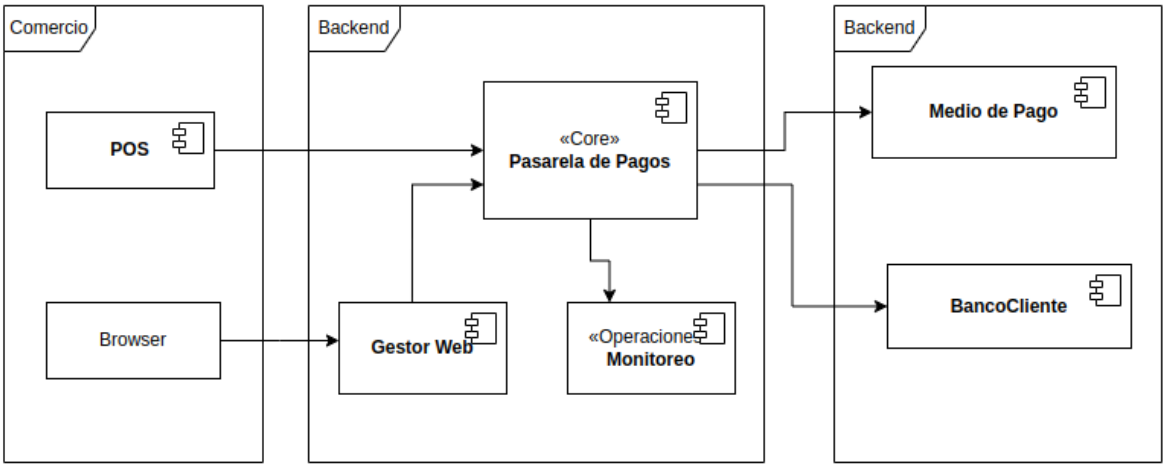
Integración de Sistemas.

En las aplicaciones empresariales, los sistemas o subsistemas muy rara vez trabajan de forma aislada. Por lo general los procesos de negocio son transversales a varios sistemas.

Cuando diferentes sistemas trabajan juntos se dice que están integrados.

Desde un punto de vista general, podemos decir que nuestro sistema cumple con dos roles:

- actúa como cliente que consume servicios externos
- actúa como servicio, consumido por otros subsistemas o componentes



Descripción del esquema de integración

Nuestro Sistema ofrece servicios al **Comercio**

- El principal servicio (Pasarela de Pagos) es realizar el cobro de una venta (mediante el POS)
- Además ofrecemos al cliente la posibilidad de consultar información vía WEB. (expondremos APIs Remotas en la Pasarela de Pagos)

Nuestro Sistema actúa como cliente de los siguientes servicios:

- Consume un servicio en el sistema Medio de Pago, que es que el verdaderamente sabe si la el pago se puede realizar o no
- Además consume el servicio del Bando del Cliente, donde notificamos que realizamos un depósito en la cuenta del cliente

Además Nuestro Sistema está dividido en tres subsistemas:

- Subsistema de Gestión Web, permite al usuario interactuar vía web (no se implementa en esta versión)
- Subsistema de Monitoreo, permite a los operarios monitorear el sistema

Tareas a realizar.

Pasarela de Pagos

Exponer API REST para Comercio

1. Exponer como API Rest la/s funcionalidad/es utilizada/s por el Comercio.
2. La API Rest deberá ser segura, es decir, se deberá implementar mecanismo de autenticación y autorización pertinente.
3. Cada comercio tendrá un **usuario y contraseña** que será utilizada cada vez que se quiera acceder a un endpoint de la API Rest.
4. Además se debe implementar un **RateLimiter** para la funcionalidad de realizarPago().

Exponer API REST para Medio de Pago

1. Exponer una API Rest la operación que recibe la notificación de una transferencia de dinero desde el medio de pago hacia el Banco de la Pasarela.

Consumir API REST MOCK del Medio de Pago

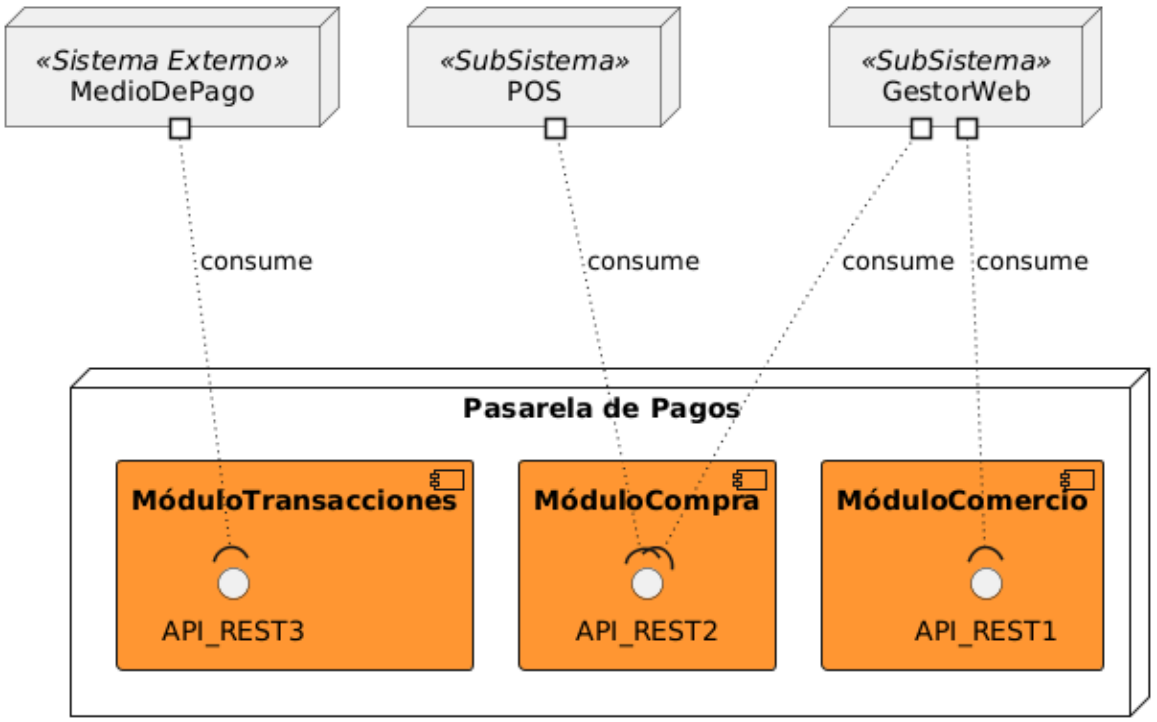
1. Cuando la pasarela de pagos recibe la petición de procesar un pago, debe de invocar al sistema de Medio de Pagos utilizando la API Rest expuesta por el mismo.

Consumir API SOAP MOCK del Banco del Cliente

1. Cuando recibimos la notificación del Medio de pago, indicando que se ha depositado el dinero de una compra, debemos invocar vía SOAP a la API Remota del Banco para que realice el depósito en la cuenta del Comercio.

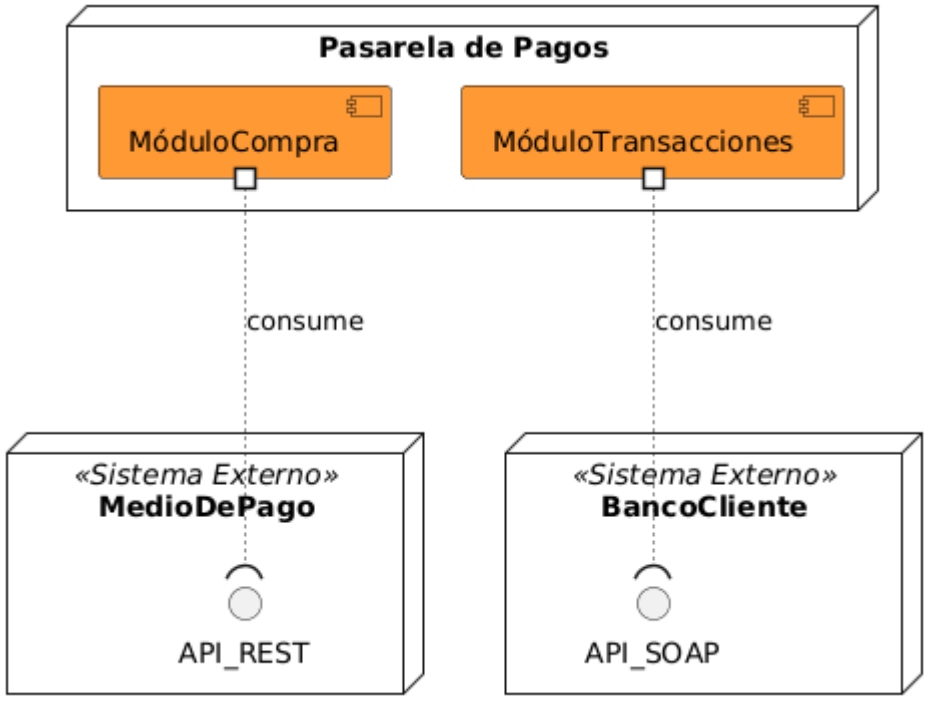
Nota: por razones de tiempo y simplicidad no configuraremos ssl para la comunicación https.

APIs que se exponen



En este caso los módulos son servicios ofrecidos con APIs Remotas.

APIs que consumen



En este caso los módulos son “clientes” de APIs.

Sistema Externo de Medio de Pago

API REST MOCK de sistema Medio de Pago

1. Crear un proyecto web nuevo que se llame ServicioMedioPagoMock
2. Exponer una API Rest con funcionalidad que simule la autorización o no de un pago.
3. Con el objetivo de facilitar las pruebas de integración, es posible implementar que para ciertos números de tarjetas el pago siempre se retorne ok o error dependiendo del caso.
4. Contemple:
    a. retornar un código de autorización cuando del pago es confirmado o un código de error cuando el pago es rechazado
    b. rechazar o confirmar pagos de forma randómica
    c. priorizar los pagos aceptados sobre los pagos rechazados, por ejemplo en un relación de 5 a 1

Sistema Banco Cliente

API SOAP MOCK de sistema Banco Cliente

1. Crear un proyecto web nuevo que se llame BancoClienteMock
2. Exponer una API SOAP con funcionalidad que reciba la notificación de la transferencia de dinero desde la Pasarela hacia la cuenta del Cliente.
3. Contemple:
    a. retornar un código de confirmación

Aclaraciones varias.

Dependencias.

Tenga en cuenta que para implementar las nuevas funcionalidades, deberá incluir dependencias en tiempo de compilación y ejecución (servidor). Analice los pom.xml de los proyectos de referencia utilizados en clases.

Capa de persistencia.

Si no está implementada aún, en esta iteración se deberá comenzar a implementar la persistencia del Sistema Pasarela de Pagos.

Nota: las aplicaciones MOCK no implementan persistencia.

Entrega.

Se debe entregar el link a un fork o branch del repositorio github utilizado.

Se debe entregar documentación de lo realizado; utilizar el archivo README.md del repositorio. La documentación se debe agregar a la ya realizada en la iteración 1.

Aplicar los siguientes criterios para decidir que es importante y relevante documentar:

1. priorice diagramas uml sobre texto (por ejemplo, diagrama de integración entre sistemas)
2. contextualice la información ofrecida
3. sea prolijo y mantenga uniformidad de formato
4. jerarquice la información subtitulando contenido