Programación 2 - año 2023

Trabajo Final

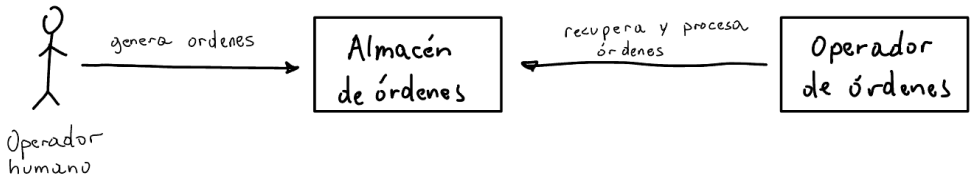
Enunciado del trabajo de regularización y aprobación de la materia

Versión 1.1 (2023-09-26)

Resumen

Servicio de compra/venta de acciones en la bolsa.

Existen servicios que serán los encargados de hacer funcionar el sistema, para la compra y venta de acciones se van a crear órdenes de compra/venta que quedarán almacenadas hasta que sean procesadas.
El procesamiento de las órdenes se realiza de forma asíncrona donde el servicio de procesamiento de órdenes se conectará con el almacén de operaciones pendientes para ser procesadas.



Enunciado

El sistema estará compuesto por cuatro componentes, 1 desarrollado por la cátedra (**servicio cátedra**) y los otros 3 por los alumnos (**procesador de órdenes**, **generador de órdenes**, y **servicios complementarios**).
Está pensado para equipos de trabajo de 1 a 3 alumnos, los alumnos deberán desarrollar si o si el servicio de procesamiento de operaciones (para el caso de un alumno individual) y los otros servicios para grupos de 2 o 3 alumnos.

El procesador de órdenes utiliza datos que se encuentran en el **generador de órdenes** y en el servidor de **servicios complementarios**. Si estos servicios no son desarrollados por el grupo, serán provistos como valores fijos desde el **servicio de cátedra**.

En caso de que el grupo decida desarrollar estos dos servicios, estos generarán los datos necesarios y el servicio del procesador de operaciones en lugar de obtener datos genéricos fijos, los obtendrá de los otros servicios desarrollados por los alumnos. Al final del documento se escribe un mapa de cómo sería cada caso.

Detalles

El sistema global (conjunto de todos los micro servicios) permite manejar la compra o venta de acciones de la bolsa.
El **servicio de cátedra** es un servicio que utilizarán todos los grupos (aquellos grupos que no desarrollen todos los servicios), por ese motivo los usuarios deberán crear un usuario e iniciar sesión para utilizar posteriormente los endopoints necesarios.

Proceso de creación de orden de compra

1- Se busca un cliente desde la lista de clientes y se selecciona.
2- Se busca una acción desde la lista de acciones y se selecciona.
3- Se consulta el valor de las acciones, el valor puede ser el valor actual o una lista de valores históricos hasta el momento actual. El rango de valores es de hasta 4 horas.
4- Se define el modo de la operación (en qué momento se hará la operación), esto puede ser:

- inmediato: a la hora que se genere el pedido de la operación, siempre y cuando estemos en el horario de operaciones entre las 09:00 y 18:00
- a la hora de inicio de operaciones (a las 09:00)
- a la hora de cierre de operaciones (a las 18:00)

5- Se genera la orden de operación para ser procesada posteriormente.
Antes de generar la orden, debe verificarse que el Id de cliente y el Id de la acción sean válidos. Para esto se debe consultar el servicio de la cátedra (o el de servicios complementarios) buscando por Id de ambos.

Proceso de la creación de orden de venta

1- Se busca un cliente desde la lista de clientes y se selecciona.

2- Se selecciona la acción desde la lista de acciones.
En caso de haber desarrollado el **generador de órdenes**, se le pide la lista de acciones que ese cliente tiene comprado.
En caso de no haber desarrollo, se obtiene el cliente de la lista de clientes.

3- Se consulta si el cliente tiene acciones de la compañía seleccionada.

4- Se recupera el valor de acciones disponibles para vender. Si el valor es 0, no se puede vender. Si el valor es un número negativo significa que no hay un seguimiento de las acciones a vender y se puede vender cualquier número.

5- Se define el modo de la operación (en qué momento se hará la operación), esto puede ser:

- inmediato: a la hora que se genere el pedido de la operación, siempre y cuando estemos en el horario de operaciones entre las 09:00 y 18:00
- a la hora de inicio de operaciones (a las 09:00)
- a la hora de cierre de operaciones (a las 18:00)

6- Se genera la orden de operación para ser procesada posteriormente.
Antes de generar la orden, debe verificarse que el Id de cliente y el Id de la acción sean válidos. Para esto se debe consultar el servicio de la cátedra (o el de servicios complementarios) buscando por Id de ambos.

Almacenamiento de órdenes

Cada orden que se genera quedará almacenada en el servidor de **generación de órdenes** hasta que el servidor de **procesamiento de órdenes** haga una consulta, en ese caso se armará una lista de órdenes que serán enviadas al procesador.
Si el procesador hace una nueva consulta y no hay órdenes nuevas creadas, la lista estará vacía.
Las nuevas órdenes se irán acumulando hasta tanto el procesador de órdenes haga una nueva consulta.
Las órdenes que han sido entregadas al procesador de órdenes deberán ser actualizadas indicando que ya han sido pedidas.

Procesamiento de órdenes

Las órdenes a procesarse se almacenan en el servidor de la cátedra (si no hay desarrollo de alumnos) o en el servidor de **generador de órdenes**.
En caso de usar el servidor de la cátedra, la lista de operaciones siempre será la misma. Existirá otro endpoint adicional que hará la retransmisión de un JSON que nosotros le inyectamos como valores de entrada. La idea es crear nuestra propia lista de operaciones manualmente, la inyectamos al endpoint y el endpoint nos devolverá el objeto creado.

1- El procesador de órdenes se conecta al servidor y recupera la lista de órdenes pendientes.

2- Analiza una por una las órdenes y:

- Verifica si es posible de realizar la operación de acuerdo a las condiciones de procesamiento que se encuentran más abajo (condiciones de procesamiento).
- Si NO es posible realizar la operación, se almacenará la operación en una lista de operaciones fallidas y continuamos con la siguiente.
- Si es posible de realizar la operación, realizamos el procesamiento de órdenes (explicado más abajo)
- Las órdenes de compra o venta cuyo modo es inmediato se ejecutan en el momento del análisis.
- Las órdenes de compra o venta cuyo modo sea al final del día, o a principio del día se deberán programar para ser ejecutadas cuando corresponda.
- Se debe verificar que el Id de cliente y el Id de la acción sean válidos. Para esto se debe consultar el servicio de la cátedra (o el de servicios complementarios) buscando por Id de ambos.

El resultado del análisis es una lista de órdenes que pueden procesarse y una lista de órdenes que no pueden procesarse.

Todas las órdenes que se reciben deben almacenarse en la base de datos local del servicio, cada operación debe tener información del estado de la misma para saber si se pudo ejecutar con éxito (o no), si está programada o si la órden tiene errores.

3- Cada orden procesada se notifica al servidor de **servicios de la cátedra** o al de **servicios complementarios** según corresponda (ver almacenamiento de órdenes exitosas a continuación).

Condiciones de procesamiento

- Una orden instantánea no puede ejecutarse fuera del horario de transacciones, antes de las 09:00 y después de las 18:00.
- Una orden debe tener asociado un cliente y una acción de una compañía.
- Una orden no puede tener un número de acciones <=0. Para verificar este punto se deberá hacer una consulta al servidor de **servicios de la cátedra** o al servidor de **servicios complementarios.**

Procesamiento de órdenes

El procesamiento de órdenes debería ser un proceso de llamado a un servicio externo donde realizamos la compra o venta de acciones de acuerdo a la acción definida.
Lo que hay que implementar es un servicio de procesamiento que tendrá un método de compra que devuelve un valor verdadero simulando el proceso de compra, y otro método para la venta que también devolverá un valor verdadero.

Almacenamiento de órdenes exitosas

Las órdenes ejecutadas con éxito se almacenan en el **servidor de la cátedra** o en el servidor de **servicios complementarios**.
Cuando se reporta una órden (compra o venta) se analiza si hay operaciones previas para el cliente seleccionado y para las acciones de la compañía seleccionada. Hay un solo registro donde se totaliza la cantidad de acciones, si la acción es una compra se suman las acciones al valor anterior, si la acción es una venta se restan las acciones vendidas al total.
Este punto sirve para verificar durante la creación de órdenes de venta y en el procesamiento de órdenes.

Funcionalidades por servicios

A continuación se describe en detalle la funcionalidad y endpoints del servidor de la cátedra y la lista de las funcionalidades de los servicios que deben desarrollar los alumnos.

Notas respecto del desarrollo:

- Se debe desarrollar utilizando JHipster.
- El acceso a los endpoints deben realizarse usando seguridad con tokens JWT, de la misma forma que los servicios acceden al servicio de la cátedra.
- Los servicios deben desarrollarse junto con sus pruebas (tests)

Cátedra

Esta es la descripción de los endpoints y funcionalidades del servicio de la cátedra. Cualquier duda o falta me avisan.

La referencia a cliente es a cualquier aplicación que permita realizar peticiones HTTP al servidor de la cátedra, puede ser cURL, postman, insomnia o la aplicación que los alumnos estén realizando.

En cada caso se especifica la URL, el método de acceso, los datos que se envían, los datos que se reciben y el código de respuesta HTTP de la petición.

En cada llamado se especifica la ruta sin incluir el nombre del servidor, por ejemplo:
Si la ruta completa es http://servidor:puerto/api/authenticate se va a especificar sólo /api/authenticate.

Cada endpoint tiene verificaciones de consistencia de datos, en caso de detectarse un error, el servidor devuelve el mensaje 400 Bad Request y una breve explicación de cuál es el error.

En caso de haber un error interno que no haya sido contemplado el servicio falla (sin dejar de funcionar) y devuelve un error 500, en ese caso por favor mandarme mensaje con la información detallada de lo que hicieron, los datos enviados y la fecha de ejecución para poder revisarlo.

Registro

Para que un equipo de trabajo pueda consumir los recursos de la cátedra deberá crear un usuario e iniciar sesión para obtener un token de acceso.

Proceso de registro

Crear usuario

Se debe crear el usuario accediendo a un endpoint con la siguiente información:

Cliente → Servidor: POST /api/register

```
{
  "login": "juanperez",
  "email": "juan@perez.org",
  "password": "juan123",
  "langKey": "es"
}
```

Respuesta (Servidor → Cliente): HTTP 201 Created

Activar usuario

En el paso anterior se crea el usuario en la base de datos pero no se puede iniciar sesión hasta que no se haya activado. Para activar hay que enviar este mensaje:

Cliente → Servidor: POST /api/activar

```
{
  "login": "juanperez",
  "email": "juan@perez.org",
  "nombres": "Gomez Juan, Tapia Ricardo",
  "descripcion": "Grupo 20 - blah blah blah"
}
```

Respuesta (Servidor → Cliente): HTTP 200 OK

Inicio de sesión / obtención de token de acceso

Para obtener el token de acceso debemos iniciar sesión en el servidor con el siguente mensaje:
Cliente → Servidor: POST /api/authenticate

```
{
  "username": "juanperez",
  "password": "juan123",
  "rememberMe": false
}
```

Respuesta (Servidor → Cliente): HTTP 200 OK

La respuesta es un objeto JSON con el id_token conteniendo el token.

```
{
  "id_token": "eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOi...GckUzA"
}
```

Nota: Todos los accesos posteriores deben realizarse utilizando el token.

Nota: Al registrarse un nuevo grupo, el sistema hará una copia de un listado genérico de usuarios que podrán utilizar durante el proceso de desarrollo.

Clientes

Los endpoints de clientes permitirán trabajar con la lista de clientes de cada grupo de desarrollo.

Endpoints

Lista de todos los clientes

Este endpoint devuelve la lista de todos los clientes asignados al grupo.

Cliente → GET /api/clientes/

Respuesta (Servidor → Cliente): 200 OK. La respuesta es un objeto JSON con el listado de clientes.


```
{
 "clientes": [{
     "id": 1101,
     "nombreApellido": "María Corvalán",
     "empresa": "Happy Soul"
   },
   {
     "id": 1102,
     "nombreApellido": "Ricardo Tapia",
     "empresa": "Salud Zen"
   }
 ]
}
```


Buscar cliente por nombre y/o empresa

Este endpoint permite buscar una lista de clientes filtrando por nombre y/o empresa. Ejemplos:

Cliente → GET /api/clientes/buscar?nombre=ricardo
Cliente → GET /api/clientes/buscar?empresa=mo
Cliente → GET /api/clientes/buscar?empresa=o&nombre=to

La respuesta es un objeto JSON con el listado de clientes.

Respuesta (Servidor → Cliente): 200 OK.


```
{
 "clientes": [
   {
     "id": 1211,
     "nombreApellido": "Victoria Vazquez",
     "empresa": "Cool fit"
   },
   {
     "id": 1212,
     "nombreApellido": "Guadalupe Prieto",
     "empresa": "Sabor Natural"
   }
 ]
}
```


Acciones

Los endpoints de acciones permitirán acceder a información de las acciones. Las acciones son iguales para todos los grupos.

Endpoints

Lista de todas las acciones

Este endpoint devuelve la lista de todas las acciones.

Cliente → GET /api/acciones

Respuesta (Servidor → Cliente): 200 OK.
La respuesta es un objeto JSON con el listado de las acciones.


```
{
 "acciones": [
   {
     "id": 1,
     "codigo": "AAPL",
     "empresa": "Apple Inc."
   },
   {
     "id": 2,
     "codigo": "GOOGL",
     "empresa": "Alphabet Inc. (google)"
   },
   {
     "id": 3,
     "codigo": "INTC",
     "empresa": "Intel Corporation"
   }
 ]
}
```


Buscar acciones por nombre o código (ticker)

Este endpoint permite buscar una lista de acciones filtrando por nombre de la empresa o por código. Ejemplo:

Cliente → GET /api/acciones/buscar?empresa=coca
Cliente → GET /api/acciones/buscar?codigo=o
Cliente → GET /api/acciones/buscar?empresa=o&codigo=o

Respuesta (Servidor → Cliente): 200 OK.
La respuesta es un objeto JSON con el listado de las acciones.


```
{
 "acciones": [
   {
     "id": 2,
     "codigo": "GOOGL",
     "empresa": "Alphabet Inc. (google)"
   },
   {
     "id": 4,
     "codigo": "KO",
     "empresa": "The Coca-Cola Company"
   },
   {
     "id": 12,
     "codigo": "GLOB",
     "empresa": "Globant SA"
   }
```

```
 ]
}
```

Recuperar último valor de una acción

Este endpoint devuelve el último valor de una acción (basado en la fecha y hora del pedido).

Cliente → GET /api/acciones/ultimovalor/CODIGO

Respuesta (Servidor → Cliente): 200 OK.
La respuesta es un objeto JSON con el último valor de la acción.

Ejemplo:

Cliente → GET /api/acciones/ultimovalor/AAPL

Respuesta:


```
{
 "codigo": "AAPL",
 "empresa": "Apple Inc.",
 "ultimoValor": {
   "fechaHora": "17:59:00",
   "valor": 23.14
 },
 "valores": null
}
```


Recuperar valores históricos de una acción

Este endpoint devuelve una lista de valores de una acción especificando un rango de fechas. Si el rango de fechas arranca antes de la apertura de mercado o termina después del cierre del mercado, los valores no serán incluidos en la lista. Si la ventana de consulta se encuentra íntegramente antes de la apertura o después de ella, el listado se encontrará vacío.

Cliente → GET /api/acciones/valores/CODIGO/FECHA_INICIO/FECHA_FIN

Respuesta (Servidor → Cliente): 200 OK.
La respuesta es un objeto JSON con el último valor de la acción y el listado de los valores en el rango de fecha especificado.

La fecha se especifica como: AAAA-MM-DDTHH:mm:SSz

Ejemplo:

Cliente → GET /api/acciones/valores/AAPL/2023-12-10T09:00:00z/2023-12-10T11:00:00z

Respuesta:


```
{
 "codigo": "AAPL",
 "empresa": "Apple Inc.",
```

```
 "ultimoValor": {
   "fechaHora": "2023-12-10T11:00:00",
   "valor": 23.1
 },
 "valores": [
   {
     "fechaHora": "2023-12-10T09:00:00",
     "valor": 23.1
   },
   {
     "fechaHora": "2023-12-10T09:01:00",
     "valor": 23.1
   },
   {
     "fechaHora": "2023-12-10T09:02:00",
     "valor": 23.1
   },
   {
     "fechaHora": "2023-12-10T09:03:00",
     "valor": 23.11
   }
 ]
}
```


Buscar Órdenes

Permite recuperar un conjunto de órdenes necesarias para el servicio de procesamiento de órdenes.

Obtener orden fija

Este endpoint devuelve siempre la misma lista de órdenes, se utiliza durante el desarrollo para poder tener la posibilidad de consultar un servicio y que devuelva una lista de órdenes con las cuales podemos trabajar.

Cliente → GET /api/ordenes/ordenes

Respuesta (Servidor → Cliente): 200 OK.
La respuesta es un objeto JSON con la lista de órdenes a procesar.


```
{
 "ordenes": [
   {
     "cliente": 1101,
     "accionId": 3,
     "accion": "INTC",
     "operacion": "COMPRA",
     "precio": null,
     "cantidad": 10,
     "fechaOperacion": "2023-09-25T03:00:00Z",
     "modo": "AHORA"
   },
   {
     "cliente": 1101,
     "accionId": 3,
```

```
      "accion": "INTC",
      "operacion": "COMPRA",
      "precio": null,
      "cantidad": 10,
      "fechaOperacion": "2023-09-25T03:00:00Z",
      "modo": "AHORA"
    },
    {
      "cliente": 1102,
      "accionId": 2,
      "accion": "GOOGL",
      "operacion": "VENTA",
      "precio": null,
      "cantidad": 5,
      "fechaOperacion": "2023-09-25T03:00:00Z",
      "modo": "AHORA"
    }
  ]
}
```


Espejo de orden personalizada

Este endpoint recibirá un objeto JSON representando una lista de órdenes que serán retransmitidas.
Cliente → Servidor: POST /api/ordenes/espejo


```
{
  "ordenes": [
    {
      "cliente": 1202,
      "accionId": 2,
      "accion": "GOOGL",
      "operacion": "COMPRA",
      "cantidad": 10,
      "fechaOperacion": "2023-09-25T03:00:00Z",
      "modo": "AHORA"
    },
    {
      "cliente": 1201,
      "accionId": 3,
      "accion": "INTC",
      "operacion": "VENTA",
      "cantidad": 20,
      "fechaOperacion": "2023-09-25T03:00:00Z",
      "modo": "FINDIA"
    },
    {
      "cliente": 1201,
      "accionId": 3,
      "accion": "INTC",
      "operacion": "VENTA",
      "cantidad": 25,
      "fechaOperacion": "2023-09-25T03:00:00Z",
      "modo": "FINDIA"
    }
  ]
```

```
}
```

Respuesta (Servidor → Cliente): HTTP 200 OK

```
{
 "ordenes": [
   {
     "cliente": 1202,
     "accionId": 2,
     "accion": "GOOGL",
     "operacion": "COMPRA",
     "cantidad": 10,
     "fechaOperacion": "2023-09-25T03:00:00Z",
     "modo": "AHORA"
   },
   {
     "cliente": 1201,
     "accionId": 3,
     "accion": "INTC",
     "operacion": "VENTA",
     "cantidad": 20,
     "fechaOperacion": "2023-09-25T03:00:00Z",
     "modo": "FINDIA"
   },
   {
     "cliente": 1201,
     "accionId": 3,
     "accion": "INTC",
     "operacion": "VENTA",
     "cantidad": 25,
     "fechaOperacion": "2023-09-25T03:00:00Z",
     "modo": "FINDIA"
   }
 ]
}
```

Registro/Reporte de operaciones

Permite registrar y consultar las operaciones que se han ejecutado en el procesador de órdenes.

Registro de operaciones

Este endpoint permite registrar el resultado de la operación sobre las órdenes. Puede recibir múltiples órdenes en un solo llamado.

Cliente → Servidor: POST /api/reporte-operaciones/reportar

```
{
 "ordenes": [
   {
     "cliente": 1102,
     "accionId": 1,
     "accion": "AAPL",
     "operacion": "COMPRA",
```

```
    "cantidad": 10,
    "precio": 12.20,
    "fechaOperacion": "2023-09-25T03:00:00Z",
    "modo": "AHORA",
    "operacionExitosa": true,
    "operacionObservaciones": "ok"
  },
  {
    "cliente": 1103,
    "accionId": 4,
    "accion": "KO",
    "operacion": "VENTA",
    "cantidad": 5,
    "precio": 14.55,
    "fechaOperacion": "2023-09-25T03:00:00Z",
    "modo": "FINDIA",
    "operacionExitosa": true,
    "operacionObservaciones": "ok"
  }
 ]
}
```


Respuesta (Servidor → Cliente): HTTP 200 OK
La respuesta muestra el texto “aceptado”

Consulta de operaciones históricas

Este endpoint nos permite recuperar la lista de operaciones registradas en el servidor. Esto es útil para saber si las operaciones que estamos registrando han sido ejecutadas correctamente. Opcionalmente se puede buscar por rango de fecha, id de cliente, id de acción.

Cliente → Servidor: GET /api/reporte-operaciones/consulta

Respuesta (Servidor → Cliente): HTTP 200 OK
La respuesta es un objeto JSON con la lista de operaciones.


```
[
 {
   "clienteId": 1101,
   "cliente": "María Corvalán",
   "accionId": 2,
   "accion": "GOOGL",
   "operacion": "COMPRA",
   "cantidad": 10,
   "precio": null,
   "fechaOperacion": "2023-09-25T03:00:00Z",
   "modo": "AHORA",
   "operacionExitosa": true,
   "operacionObservaciones": "ok"
 },
 {
   "clienteId": 1101,
   "cliente": "María Corvalán",
```

```
    "accionId": 1,
    "accion": "AAPL",
    "operacion": "VENTA",
    "cantidad": 5,
    "precio": null,
    "fechaOperacion": "2023-09-25T03:00:00Z",
    "modo": "FINDIA",
    "operacionExitosa": true,
    "operacionObservaciones": "ok"
  },
  {
    "clienteId": 1102,
    "cliente": "Ricardo Tapia",
    "accionId": 1,
    "accion": "AAPL",
    "operacion": "COMPRA",
    "cantidad": 10,
    "precio": null,
    "fechaOperacion": "2023-09-25T03:00:00Z",
    "modo": "AHORA",
    "operacionExitosa": true,
    "operacionObservaciones": "ok"
  }
]
```


Ejemplos de búsquedas:

Cliente → Servidor: GET /api/reporte-operaciones/consulta
Cliente → Servidor: GET /api/reporte-operaciones/consulta?clienteId=1102
Cliente → Servidor: GET /api/reporte-operaciones/consulta?accionId=1
Cliente → Servidor: GET /api/reporte-operaciones/consulta?clienteId=1101&accionId=1
Cliente → Servidor: GET /api/reporte-operaciones/consulta?clienteId=1101&accionId=1&fechaInicio=2023-09-24T02:00:00Z&fechaFin=2023-09-25T04:00:00Z

Procesador de órdenes

Funcionalidades que debe cumplir el servicio:

- Obtener listado de órdenes.
- Procesar órdenes inmediatas.
- Programar órdenes programadas.
- Registrar resultado de órdenes.
- Logs de actividad relevante (órdenes recuperadas del servidor, órdenes procesadas, ordenes programadas)
- Pruebas (testing).
- Seguridad
- Reportes
    - Lista de órdenes procesadas, filtrar por (el filtrado puede ser por múltiples campos):
        - cliente
        - acción

        - fecha
    - Lista de órdenes no procesadas y mostrar motivo

Generador de órdenes

Funcionalidades que debe cumplir el servicio:

- Crear órdenes de compra
- Crear órdenes de venta
- Buscador de órdenes de compra
- Buscador de órdenes de venta
- Anular orden de compra (si la órden todavía no fue procesada)
- Anular orden de venta (si la órden todavía no fue procesada)
- Logs de actividad relevante (órdenes creadas, órdenes rechazadas, órdenes anuladas)
- Pruebas (testing)
- Seguridad
- Reportes
    - Lista de órdenes creadas, filtrar por (el filtrado puede ser por múltiples campos):
        - cliente
        - acción
        - tipo de operación
        - fecha

Servicios complementarios

Funcionalidades que debe cumplir el servicio:

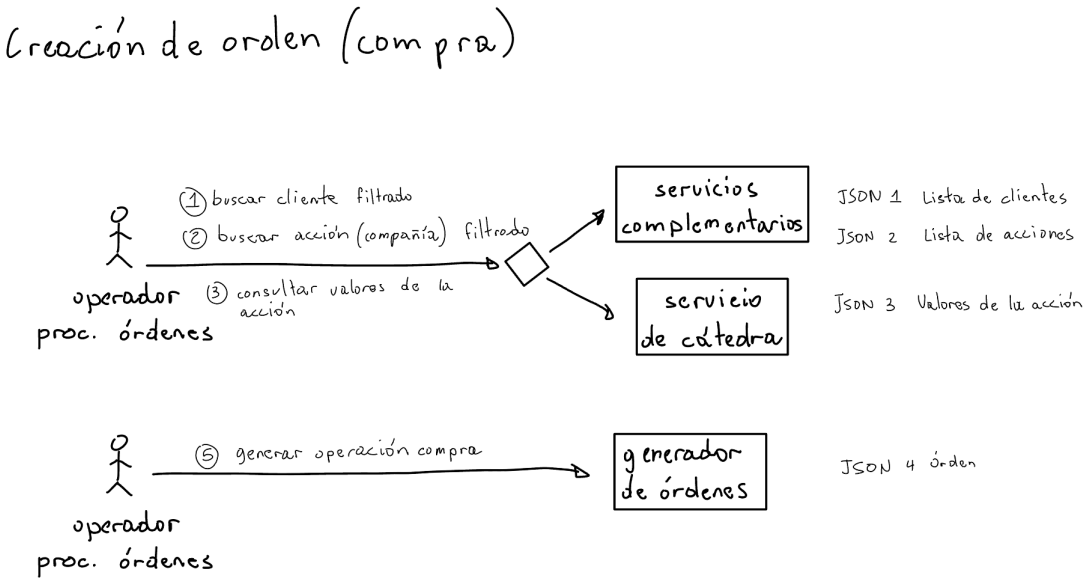
- Registro resultado de órdenes. Debe implementar la funcionalidad de almacenamiento temporal de las órdenes y cada 1 hora reportar las órdenes al servidor de la cátedra. El procesador de órdenes deberá reportar en este servicio en vez de hacerlo al servidor principal.
- Proxy de usuarios. Los servicios que obtienen la información de usuarios desde el servicio de la cátedra, deberán enviarle las peticiones al servidor de servicios complementarios y este redirigirá la petición al servicio de la cátedra.
- Proxy de acciones. Los servicios que obtienen la información de acciones desde el servicio de la cátedra, deberán enviarle las peticiones al servidor de servicios complementarios y este redirigirá la petición al servicio de la cátedra.
- Pruebas
- Logs de actividad
- Seguridad

Flujo de operaciones

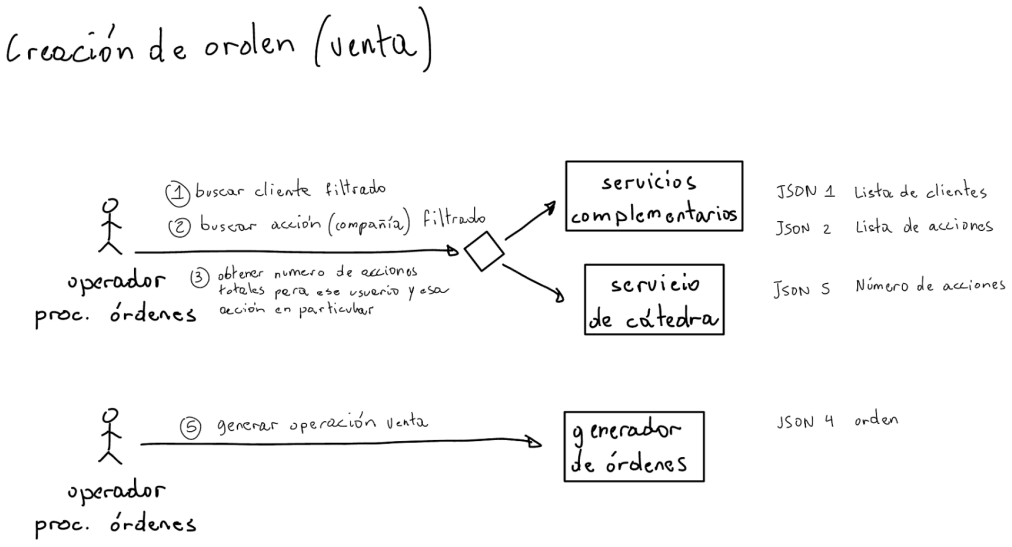
A continuación hay unas imágenes que explican gráficamente las operaciones.

Proceso de creación de orden de compra

Este es el proceso de creación de las órdenes de compra, esto lo realiza el operador interactuando manualmente con el **servicio de cátedra** o el **servicio complementario** según corresponda.



Proceso de creación de orden de venta



Procesamiento de órdenes