Entrega final - ECSDI

Amazon2

Alex Huang Feng

Nico Ernesto Francisquelo Tacca

Eloi Roca Corcelles

1. Especificación del sistema

Para poder representar los elementos de nuestro sistema, utilizamos la metáfora de una tienda física, en el que los agentes son empleados y los roles son las funciones que deben hacer en su puesto de trabajo. De esta forma nos aseguramos una mayor abstracción de la tecnología en las fases de especificación y diseño.

1.1 Escenarios posibles

Nuestro sistema se puede ver afectado por los diferentes escenarios que vamos a presentar a continuación.

- **Visualización de productos disponibles**: Este escenario contempla el proceso donde el usuario externo accede al sistema con tal de conocer los productos disponibles.
- **Compra de productos**: Escenario donde el usuario quiere comprar uno o más productos de nuestro sistema y éste tiene que gestionar todo el proceso para hacer llegar el producto al usuario. Se descompone por 3 sub-escenarios para simplificar el sistema.
    - **Interacción con usuario**: Proceso donde el usuario externo accede a nuestro sistema y a través del cual obtenemos información sobre lo que está buscando y datos de compra.
    - **Efectuar cobro**: Se hace el intercambio de dinero, el pago de los productos comprados una vez sabemos que los productos han sido enviados o inmediatamente si se trata de una venta con envío externo.
    - **Envío del producto**: Este escenario contempla el envío de los productos pedidos al usuario comprador.
- **Devolución de productos**: Escenario que trata toda la gestión de devolución de un producto comprado en nuestro sistema.
- **Puesta de productos a la venta**: Puesto que nuestro sistema ofrece la posibilidad de poner a la venta productos externos, este escenario trata todo el proceso de publicación de los productos que el vendedor externo quiere vender a partir de nuestro sistema.

- **Obtención de feedback del usuario**: Dado que nuestro sistema permite al usuario escribir reseñas sobre los productos que ha comprado, este escenario contempla el proceso que sigue el sistema para obtener estas reseñas.
- **Envío de recomendaciones al usuario**: Escenario que trata las recomendaciones que se envían a los usuarios en función de los productos que han comprado.

1.2 Objetivos del sistema

Nuestro sistema se compone de 5 objetivos objetivos principales. A partir de estos, se pueden descomponer en diferentes sub-objetivos con tal de simplificar el sistema. Explicaremos uno por uno cada objetivo:

- **Visualización de productos disponibles:** Objetivo que contempla la visualización de los productos disponibles por el usuario.
- **Compra de productos:** Objetivo que consiste en gestionar la compra de uno o más productos de nuestro sistema por un usuario externo. Para entender mejor el cumplimiento de este objetivo explicaremos los tres sub-objetivos no triviales que lo forman:
  - *Interacción con usuario*: contempla la necesidad de recibir información del usuario, sería como preguntarle al cliente qué anda buscando.
  - *Entregar producto*: dado a que una de las características de nuestro sistema es la existencia de vendedores externos que pueden encargarse del envío del producto, debemos tener en cuenta que podemos *Gestionar el envío* nosotros mismos o simplemente *Comunicar envío al vendedor*.
    - En el caso que seamos nosotros los que gestionan el envío, primero tendremos que *Decidir el Almacén* desde el qual se enviará el producto, esta decisión simplemente constará de la elección del almacén más cercano al cliente y que contiene el producto. Hemos pensado añadir más características a la toma decisiones como podría ser enviar desde el sitio más barato (puede no ser el más cercano) pero creímos conveniente no añadir más complejidad. Una vez sabemos el almacén, éste debe *Negociar con los transportistas* y elegir el que ofrezca las mejores condiciones, para finalmente *Realizar el envío del producto*.

- ➢ *Efectuar cobro*: que se divide en *Cobrar el pedido,* que va a *Ingresar dinero en la tienda* una vez se haya realizado el envío o a *Cobrar comisión y pagar al vendedor* de forma inmediata si es un envío externo, y en *Emitir la factura,* que contendrá la información sobre la compra y, si lo hemos gestionado nosotros, los detalles del envío.

- ❖ **Puesta de productos a la venta:** Objetivo que consiste en que un vendedor externo pueda vender sus productos a través de nuestro sistema.

- ❖ **Devolución de productos:** Objetivo que consiste en que un usuario pueda devolver productos fallidos o erróneos que ha comprado y poder recuperar su dinero

- ❖ **Envío de recomendaciones:** Objetivo que consiste en enviar recomendaciones a los usuarios sobre productos en los que pueda estar interesado para incentivar su compra.

1.3 Roles

Dentro de nuestra plataforma, hemos definido hasta 11 roles. A continuación explicaremos cada uno de ellos de forma resumida.

- **Procesador de Productos disponibles**: Rol que trata de cumplir el objetivo de visualizar los productos disponibles del sistema. Se activa cuando hay una *Petición Productos disponibles*. Cuando se activa enviará la información de los productos disponibles del sistema.

- **Procesador de solicitudes de compra** Se encarga de tratar las peticiones de compra de productos. Pretende cumplir el objetivo de *Interactuar con el usuario* y cuando se activa devuelve una confirmación de que la compra se está efectuando.

- **Procesador de cobro**: Rol que se ocupa de procesar todos los cobros y cobrar una comisión e ingresar el dinero en la cuenta del vendedor si se trata de un producto externo. Pretende cumplir el objetivo de *Efectuar cobro*. Se activa cuando el sistema recibe tiene un proceso de compra y se necesita realizar un cobro. Cuando se activa, cobra el pedido y emite una factura.

- **Procesador de devoluciones**: Rol que se ocupa de procesar todas las devoluciones de productos comprados en nuestro sistema. Pretende cumplir el objetivo *Devolución de productos* y se activa cuando recibe una petición de devolución de producto externa. Cuando se activa, se confirma la devolución del producto si éste ha sido efectuado con éxito.

- **Decididor de almacén**: Rol que se encarga de escoger el almacén más cercano al cliente para efectuar el envío del pedido. Quiere cumplir el objetivo de *Decidir Almacén* y se activa cuando dentro del sistema surge una necesidad de nuevo envío de paquete.

- **Procesador de recomendaciones**: Rol encargado de dar recomendaciones de productos a los clientes. Se activa cada día con la percepción *Toca enviar recomendación* y envía una recomendación de productos que al usuario le pueda interesar. Pretende cumplir el objetivo *Envío de recomendaciones al usuario*.

- **Procesador de envio de productos**: Rol que pretende cumplir el objetivo de *Realizar envío producto*. Se activa cuando dentro del sistema recibe una petición de envío de producto. Cuando se activa a partir de la petición interna, avisa al transportista de la existencia de un nuevo envío.

- **Procesador de puesta a la venta**: Rol que se ocupa de gestionar las peticiones externas de *Usuario quiere vender producto*. Pretende cumplir el objetivo de *Puesta de productos a la venta*. Su trabajo se limita a recibir las peticiones externas y actualizar la lista de productos disponibles dentro del sistema.

- **Procesador de feedback de usuarios**: Rol encargado de recibir las reseñas sobre productos del sistema de los usuarios. Recibe peticiones externas de *Usuario quiere escribir una reseña* y pretende cumplir el objetivo de *Obtención feedback de usuario*.

- **Procesador de envío externo**: Rol que comunica al transportista la necesidad de realizar un nuevo envío. Pretende cumplir el objetivo de *Realizar envío de producto* y cuando se activa, avisa al transportista externo de que tiene un nuevo envío que efectuar.

- **Negociador de transportistas**: Rol ocupado de negociar los precios y condiciones de transporte con los transportista. Cumple el objetivo de *Negociar con transportistas* y se activa cuando en el proceso de compra de un producto, se necesita negociar el precio

del envío. Cuando se activa, envía una petición a los transportistas para saber sus precios y condiciones.

1.4 Análisis externo

En una visión general del sistema, identificamos 4 actores externos: la entidad bancaria, el transportista, el cliente de nuestro servicio y el vendedor externo. Obviamos nuestro propio sistema como actor, ya que interviene en todos los escenarios de nuestro sistema.

La **entidad bancaria** és la empresa que se ocupa de hacer todas las transacciones de dinero entre nuestros clientes y nuestra empresa. Se asegura que todas estas transferencias se efectúan con éxito. Interactúa con nuestro sistema para hacer las transacciones de dinero.

El **transportista** es un actor del sistema encargado de asegurar que la entrega de los paquetes es efectuada con éxito. Interactúa en el proceso de compra de un producto.

El **cliente** es el actor que accede a nuestro sistema para utilizar nuestras funcionalidades. Interactúa con el agente Cliente para efectuar una compra, visualizar los productos disponibles dentro de nuestro sistema, devolver un producto comprado o escribir una reseña de algún producto comprado anteriormente.

El **vendedor externo** es el actor que utiliza nuestro sistema para vender productos propios a través de nuestro sistema. Utiliza el agente vendedor externo para poner a la venta un producto propio.

2. Diseño de la arquitectura

2.1 Acoplamiento de datos

El sistema que proponemos está formado por varios elementos de almacenaje de datos que serán accedidos por los agentes ya sea para hacer consultas o actualizar la información.

Estos elementos son:

- **Info Productos:** Contiene información sobre todos los productos que se venden en el sistema. Es consultado por el rol *Procesador de cobro* para obtener el precio del producto, y por el *Procesador de productos disponibles* para obtener los productos disponibles del sistema. Es actualizado por el rol *Procesador de puesta a la venta* para añadir nuevos productos y por el rol *Procesador de solicitudes de compra* para decrementar los productos que se han vendido.

- **Info Vendedores Externos:** Contiene información sobre los vendedores externos al sistema. Es consultado por el rol *Procesador de envío externo* para poder avisar al vendedor que debe encargarse del envío. Es actualizado por el rol *Procesador de puesta a la venta* para añadir la información si se añade un producto de un vendedor nuevo.

- **Info Envíos Almacén:** Contiene información sobre el historial de envíos realizados por el almacén. Es actualizado por el rol *Procesador de envío de productos* cada vez que se produce un envío.

- **Info Clientes:** Contiene información sobre los clientes, datos de contacto, datos bancarios, entre otros. Es consultado por los roles *Procesador de solicitudes de compra* y *Procesador de recomendaciones*. Es actualizado por el rol *Procesador de solicitudes de compra* por ejemplo si necesita añadir una nueva tarjeta para el pago.

- **Info Almacén:** Contiene información sobre los almacenes del sistema. Es consultado por el rol *Decididor Almacén* para poder elegir el almacén más cercano al cliente desde el que enviar el producto.

- **Historial Compras:** Contiene información sobre el historial de compras realizadas a través del sistema. Es consultado por el rol *Procesador de devoluciones* para comprobar que el cliente ha comprado el producto que solicita devolver y por el rol

Procesador de recomendaciones para generar recomendaciones basadas en compras de productos similares. Es actualizado por el rol *Procesador de solicitudes de compra*.

- **Feedback de productos:** Contiene información sobre opiniones y valoraciones de los productos por parte de los clientes. Es consultado por el rol *Procesador de recomendaciones* para generar recomendaciones basadas en productos que le han gustado al cliente. Es actualizado por el rol *Procesador de feedback de usuarios*.

- **Info Transportistas Almacén:** Contiene información sobre los transportistas que trabajan con un determinado almacén del sistema. Es consultado por el rol *Negociador de transportistas*.

2.2 Agrupación de roles en agentes

En el sistema propuesto hemos definido toda una serie de roles, los cuales tienen que ser efectuados por algún agente, en nuestro sistema hemos identificado a los siguientes:

- **Agente Cliente:** Es el agente representante del **cliente** que quiere comprar un producto en nuestro sistema. Agrupa los roles *Procesador de Productos Disponibles, Procesador de solicitudes de compra, Procesador de devoluciones* y *Procesador de cobro*. Hará toda la elección según los criterios del cliente.

- **Agente Vendedor Externo:** Es el agente representante de los vendedores externos, su único rol en el sistema es el de *Procesador de puesta a la venta*.

- **Agente Recomendador:** Es el agente encargado de la gestión del feedback y la recomendación de productos. Sus dos roles son *Procesador feedback usuario* y *Procesador de recomendaciones*.

- **Agente Logístico:** Es el agente que se encarga de la gestión de los envíos de los productos. Los roles que efectúa son *Procesador de envio externo* y *Decididor de almacén*.

- **Agente Almacén:** Es el agente que representa cada uno de nuestros almacenes. Los roles que efectuará son *Negociador con transportistas* y *Procesador de envío de productos*.

Esta agrupación de roles en agentes se ha hecho teniendo en cuenta principalmente los escenarios en los que interviene cada rol, los actores externos involucrados en cada

escenario y los objetivos que persigue cada rol. También se ha tenido en cuenta la distribución de estos agentes físicamente para hacer la agrupación.

2.3 Interacción entre agentes

El sistema propuesto está formado por varios agentes, algunos de los cuales son independientes de los demás, es decir no requieren de ningún tipo de interacción con otros agentes del sistema para su funcionamiento, sin embargo en nuestro diseño existen 3 agentes que deben establecer una comunicación para poder cumplir los objetivos de los roles que tienen asignados.

Para la correcta comunicación entre estos agentes, hemos diseñado el *Protocolo Envio Producto* basado en un intercambio de mensajes a 3 bandas.

Este protocolo se inicia con un ***Mensaje de Solicitud de Envío*** generado por el *Agente Cliente* y dirigido al *Agente Logístico* para pedirle que se encargue de enviar el o los productos al cliente que los ha comprado.

En caso de que el envío de los productos lo deba gestionar el vendedor externo, acabaremos el protocolo con un ***Mensaje de Confirmación de Envío Externo*** hacia el *Agente Cliente*.

Alternativamente, si el envío está a cargo de nuestro sistema, el *Agente Logístico,* después de decidir desde qué almacén se debe efectuar el envío, genera un ***Mensaje de Realiza Envío*** dirigido al *Agente Almacén* para avisarle que debe encargarse del envío de esos productos. Llegados a este punto, el *Agente Almacén* deberá negociar con los transportistas y una vez haya cerrado el envío, emitirá un ***Mensaje de Confirmación de Envío*** para notificar al *Agente Cliente* que ya se ha efectuado el envío.

2.4 Visión general del sistema

La visión general del sistema describe las funcionalidades de cada agente, las percepciones que recibe, las acciones que realiza y los mensajes que se intercambian entre ellos.

Agente Mostrar Productos

Se encarga de mostrar los productos que coincidan con los filtros del usuario. Primero recibe la percepción del cliente para ver los productos disponibles y se encarga de acceder a los datos del sistema con los productos disponibles, filtrar estos datos según el criterio del cliente y realizar la acción de mostrar estos datos.

Agente Venta Productos

Se encarga de gestionar todo el proceso de compra una vez el usuario selecciona los productos que desea comprar de la lista de productos que le ha mostrado el Agente Mostrar Productos.

Recibe la percepción del cliente para realizar la compra de uno o más productos mostrados anteriormente, y procesa esta compra mostrando al cliente el contenido de la cesta (como confirmación de la realización de la compra), actualizando la información del historial de compras y mandando un mensaje de solicitud de envío al agente logístico con la información de los productos a enviar.

Una vez recibe el mensaje de confirmación de envío, que puede llegar del Agente Logístico o del Agente Almacén, realiza la acción de cobrar el pedido a la entidad bancaria y también realiza la acción de enviar la factura al cliente con la información de compra y los detalles del envío.

Por último recibe la percepción del cliente para pedir una devolución de un producto comprado anteriormente, el agente trata esta petición consultando la información del historial de compras para asegurarse que el cliente compró ese pedido y luego decide según sus criterios si acepta o no la petición de devolución de producto. Una vez ha decidido realiza la acción de informar al usuario sobre el resultado de la petición, si esta se confirma se le adjunta la información de cómo realizar la devolución.

Agente Productos Externos

Se encarga de la puesta a la venta en nuestro sistema de los productos que los vendedores externos quieren vender. Recibe la percepción de estos vendedores externos con toda la información del producto que el vendedor quiere que aparezca en nuestra tienda y actualiza la información de los productos disponibles con el nuevo producto a vender.

Agente Logístico

Se encarga de gestionar el envío de un producto comprado a su destinatario. Recibe el mensaje del Agente Cliente que le informa de los productos a enviar y el destino dónde tienen que ser enviados, entonces según el tipo de producto decide si se debe realizar el envío desde un almacén o si el envío se realiza de forma externa a la tienda.

En el caso que el envío se realice mediante un almacén de nuestra tienda, el agente consulta la información de los almacenes, selecciona el almacén que tenga los productos a enviar y que esté más cercano al destino y manda un mensaje al Agente Almacén que representa a ese almacén en concreto, dónde le informa de que debe realizar un envío de los productos seleccionados al destino recibido.

En el caso que el pedido sea realizado de forma externa, el agente realiza la acción de comunicar al vendedor externo el producto que tiene que enviar y su destinación y luego procede mandando un mensaje al Agente Cliente informando de que el envío ha sido realizado por el vendedor externo y que puede proceder con el pago del pedido.

Agente Almacén

Este agente tiene la peculiaridad de que se encuentra distribuido en todos los almacenes de la tienda, de forma que en cada almacén hay un agente que gestiona únicamente a ese almacén.

Se encarga de gestionar los envíos de los productos que están en su almacén negociando con los transportistas de la zona. El agente recibe un mensaje del Agente Logístico en el que se le informa de un nuevo envío a realizar, los productos que conforman el envío y el lugar de destinación. El agente va recibiendo mensajes del agente logístico y va actualizando la información de los envíos a realizar y intenta agrupar los productos en lotes para enviarlos todos juntos.

Antes de enviar los productos primero realiza la acción de pedir precio del envío, en la que el agente se comunica con todos los transportistas de los que tiene conocimiento y les pide el precio para realizar los envíos a sus destinaciones pertinentes. Una vez el agente conoce todos los precios de los transportistas, selecciona el transportista más económico y realiza la acción de avisar al transportista de los nuevos envíos que va a tener que realizar, dónde el agente le informa de la destinación de los productos que conforman el lote.

Para finalizar, cuando el transportista confirma que realizará el envío, el agente manda un mensaje al Agente Cliente informando de que el envío va a ser realizado y que puede proceder a cobrar el pedido.

Agente Recomendador

Este agente se encarga de gestionar las reseñas y valoraciones de los usuarios que quieren valorar un producto que han comprado y de mandar recomendaciones a los usuarios en función de sus compras realizadas anteriormente y las valoraciones de los productos.

Después de cierto tiempo de que el cliente haya recibido sus productos, el agente recomendador solicita al cliente unas valoraciones de esos productos, cuando el agente recibe la información de la reseña del cliente y su valoración y actualiza la información del feedback del producto.

El agente también recibe una percepción auto-generada por el sistema que le recuerda que hay que enviar recomendaciones a los clientes, entonces el agente consulta la información de los clientes, su historial de compras y el feedback de los productos y realiza la acción de enviar recomendación al cliente, le manda un correo con información de productos en los que podría estar interesados y que tienen una buena valoración por parte de sus clientes.

3. Diseño detallado

Agente Recomendador

El agente Recomendador está compuesto por un plan independiente y una capacidad. El plan *guardar reseñas* es disparado por la acción *Usuario quiere escribir una reseña* y pretende almacenar la reseña creada por el usuario comprador para su uso posterior. Por eso mismo, accede a los datos de *Feedback de productos* para escribir la reseña.

La capacidad de *Enviar Recomendación* es más compleja. Contiene dos planes: por una parte, obtener los intereses de un usuario y, por otra, enviar la recomendación. El primer plan es disparado por un evento de *Toca enviar recomendación* que será periódica definida por nosotros. Este plan accederá al *Historial de compras* y *Feedback de Productos* para saber qué productos pueden ser de interés para el usuario. Una vez haya calculado los intereses, se transmitirá el interés al segundo plan, quien se encargará de enviar la recomendación anteriormente calculada al usuario comprador.

Agente Venta productos

Este agente se dedica a la gestión íntegra del proceso de compra de un producto. Está formado por un plan de *Crear pedido* y una capacidad de *Cobrar pedido*.

El primer plan se dedica a la interacción con el usuario y al procesamiento del pedido que ha hecho el usuario. Será disparado por una petición externa al sistema de *Petición de compra de productos* y como respuesta a esta petición, lanzará una respuesta al usuario con la confirmación del pedido. Asimismo, enviará un mensaje al agente *Ag Logístico* para que éste pueda procesar el envío de este pedido. Este plan accedera a la información de productos, a la información de clientes y al historial de compras/pedidos.

Por otra parte, la capacidad *Cp Cobrar pedido* se dedicará a ejecutar el proceso íntegro del pago del pedido. Contiene tres planes diferentes. El *Plan Cobrar pedido* es disparado por un mensaje del agente *Ag Almacén* con la confirmación del envío del producto. Una vez recibe esta confirmación, de procederá a cobrar el pedido del comprador donde en función del tipo de producto comprado se ejecutará el *Plan emitir factura* (si el producto es propio) o el *Plan pagar vendedor externo* (si el producto es de terceros. El *Plan emitir factura* tan solo enviará la

factura al cliente una vez pagado y el *Plan pagar vendedor externo* ingresará el dinero al vendedor externo.

Agente Almacén

Este agente estará formado por una capacidad y un plan. Por una parte tendrá que negociar con los transportistas, y por otra, comunicar al transportista que ha sido elegido del nuevo envío. La capacidad *Negociar con transportistas* es disparado por un mensaje recibido del agente Logístico en el que se informa que hay un nuevo envío que realizar. Dentro de esta capacidad, se separa en dos planes, uno es la creación del lote y otro el negociar el lote con los transportistas. El *Plan Crear Lote* pretende agrupar los diferentes productos para que el envío sea en menor cantidad. El segundo plan *Negociar Lote con transportistas* se dedica a negociar el precio del envío del lote creado anteriormente con los diferentes transportistas. Enviará peticiones de precios a los diferentes transportistas externos y éstos responderán con el precio con el que trabajan. Deberá guardar en memoria estos precios para poder compararlos posteriormente.

Cuando finaliza, elegirá a un transportista según unos criterios y enviará un mensaje al segundo plan *Comunicar transportista*. Este plan avisará al transportista de que hay un nuevo envío y después comunicará al *Agente Venta Productos* para que actualice su stock.

Agente Logístico

El agente logístico contiene dos planes. El primero es disparado por un mensaje del agente *Ag Venta Productos* en el que se informa de que ha habido una nueva compra efectuada. Se accede a los *datos del almacén*, se informa al agente *Ag Almacen* de que hay un nuevo envío que realizar y en caso de que sea un producto externo y es el vendedor quien lo tiene que realizar, se lanza el segundo plan. Éste comunica al vendedor externo del producto pedido y comunica un mensaje al agente *Ag Venta productos* para actualizar el *stock* y proceder al cobro del pedido.

Agente Productos externos

Este agente encargado de gestionar los productos externos sólo contiene un plan sencillo. Es disparado por la petición de un vendedor externo que quiere vender un producto a partir de nuestra plataforma. El plan debe guardar en memoria información del producto que va a

poner a la venta y información del vendedor. Cuando el proceso ha finalizado, se envía una confirmación al vendedor de que el producto ha sido publicado.

Agente Devoluciones

Este agente contiene una capacidad formada por tres planes de trabajo. El primer plan *Decidir resultado devolución* se dedica a procesar la petición externa de nueva devolución y a decidir si la devolución será aceptada según los motivos del cliente. Una vez definida la decisión final, se disparará el segundo plan *Comunicar datos de la devolución*. Éste informará al comprador del resultado de la petición y información asociada a cómo realizar la devolución.

El segundo plan se disparará cuando haya un producto devuelto en nuestros almacenes. Se devolverá el dinero al comprador.

Agente Mostrar productos

Este agente contiene un solo plan ya que se dedicará a procesar todas las peticiones de usuarios que quieren acceder a la plataforma para ver los productos puestos a la venta. El plan *Mostrar Productos* se disparará por una petición de *Petición Productos disponibles*. Como respuesta, enviará la información de los productos disponibles del sistema.

4. Descripción de ontologías

Para la realización de nuestra versión de Amazon utilizaremos parte de una ontología externa ya existente que contiene conceptos que nos resultan útiles junto con la ontología que hemos diseñado para poder ampliar y representar correctamente el dominio de nuestro proyecto.

4.1. Conceptos ontología externa: *http://schema.org*

- **Person**: Este concepto representa una persona cliente en nuestro sistema. Nos interesan concretamente las propiedades: *address, email, name* y *telephone*.
- **GeoCoordinates**: Este concepto representa una ubicación o localización real. Nos interesan concretamente las propiedades: *address*, *addressCountry*, *latitude*, *longitude*, *postalCode*.
- **Rating**: Este concepto representa aquello que en nuestro sistema llamamos valoración o feedback del usuario. Nos interesan concretamente las propiedades: *author*, *ratingValue*, *description*.
- **Organization**: Este concepto representa tanto a los vendedores externos como a los transportistas de nuestro sistema. Nos interesan concretamente las propiedades: *address*, *name*, *telephone*, *email y disambiguatingDescription* (que utilizaremos para indicar si se trata de un vendedor o de un transportista)

4.2. Ontología diseñada

Mensajes:

Conceptos que representan las percepciones que recibe el sistema del entorno exterior, las acciones que realiza y los mensajes de protocolo de nuestro sistema. Serán recibidos y procesadas por los agentes del sistema. Hemos definido los siguientes mensajes:

Añadir_producto_externo: Representa la percepción del sistema que recibe cuando un vendedor tercero quiere vender un producto a partir de nuestro sistema. Contiene una propiedad *productoExternoContiene* que la relaciona con el concepto Vendedor y el Producto_externo asociado.

Nueva_compra_producto: Representa la percepción en la que un comprador quiere comprar uno o más productos del sistema. Contiene la propiedad *nuevaCompraContiene* que la relaciona con un pedido del usuario. Éste pedido contiene los productos que quiere comprar y la localización a la que debe ser enviado el pedido.

Nueva_valoracion: Representa la percepción externa de la escritura de una nueva reseña de un producto del sistema. Contiene la propiedad *nuevaReseñaContiene* que relaciona esta percepción con el concepto Feedback que representa una reseña.

Petición_devolución: Representa la percepción de una petición de devolución de un producto por un usuario que ha comprado a partir de nuestra plataforma. Contiene una propiedad *peticionDevolucionContiene* que la relaciona con el concepto Devolución. Este está relacionado con el producto asociado.

Petición_productos_disponibles: Representa la percepción de un cliente que quiere ver los productos disponibles de nuestro sistema. Contiene una propiedad *peticionProductosDisponibles* que la relaciona con el concepto Restricciones_cliente. Éste representa las restricciones de compra que tiene el cliente como la marca o el rango del precio que busca.

Precios_envio: Representa la percepción que envía el agente Almacén para conocer los precios de envío que proporcionan los diferentes transportistas. Tiene una propiedad *preciosEnvioPercep* que contiene la relación con el pedido asociado.

Producto_devuelto: Representa la percepción de un producto que ha sido físicamente devuelto al almacén. Contiene una propiedad *devolucionProductoConfirmado* que la relaciona con el producto que ha sido devuelto.

Toca_enviar_recomendación: Representa el evento periódico que recibe el sistema para enviar las recomendaciones al cliente.

Comunicación_resultado_devolución: Representa la acción de comunicar al cliente que ha solicitado una devolución del resultado de dicha solicitud. Tiene una propiedad *resultadoDevolución* que la relaciona con el concepto Devolución.

Comunicar_pedido_enviar: Representa la acción de comunicar a un vendedor externo que debe encargarse del envío de un pedido. Tiene una propiedad *pedidoExterno* que la relaciona con el concepto Pedido.

Avisar_transportista_envio: Representa la acción de avisar al transportista elegido, que debe encargarse del envío de un lote. Tiene una propiedad *enviaLote* que lo relaciona con Lote.

Cobrar_pedido: Representa la acción de cobrar el pedido al cliente una vez se ha gestionado el envío de éste. Tiene una propiedad *cobrarCliente* que lo relaciona con el concepto Person(schema.org)

Confirmacion_producto_externo_añadido: Representa la confirmación de que un producto de un vendedor externo ha sido añadido al stock del sistema.

Confirmacion_cesta: Representa la acción de confirmar los productos que formarán el pedido del cliente.

Devolver_dinero_cliente: Representa la acción de devolver el dinero al cliente cuando se ha aceptado la devolución de un producto y se ha recibido dicho producto.

Emitir_factura: Representa la acción de generar una factura con los detalles del pedido (desglose) y los datos de la entrega. Tiene una propiedad *tieneFactura* que lo relaciona con el concepto Factura_de_compra.

Enviar_recomendacion: Representa al acción de enviar una recomendación al cliente cuando el sistema decide que toca enviarla. Tiene una propiedad *enviaRecomendacion* que la relaciona con el concepto Recomendación.

Informacion_productos_disponibles: Representa la acción de informar al cliente sobre los productos disponibles que cumplen su restricción. Tiene una propiedad *productosFiltrados* que la relaciona con el concepto Products(schema.org).

Ingresar_dinero_vendedor_externo: Representa la acción de devolución de dinero al vendedor externo cuando se ha devuelto un producto.

Pedir_precio_envio: Representa la acción de pedir precios a los transportistas con el fin de seleccionar el más adecuado para enviar el lote.

Solicitud_envio: mensaje que avisa al centro logístico que debe procesar el envío de un pedido.

Realiza_envio: mensaje que avisa al almacén que debe encargarse de enviar un pedido.Tiene una propiedad *realizaEnvioPedido* que lo relaciona con Pedido.

Confirmacion_envio: mensaje que confirma que el almacén ya ha avisado al transportista que debe realizar el envío del lote que contiene ese pedido.

Confirmacion_envio_externo: mensaje que confirma que el pedidos ya se está procesando por el vendedor externo, por lo tanto ya se puede proceder al cobro y pago al vendedor.

Producto

Concepto que representa un Producto en nuestra tienda. Tiene las siguientes "data properties":

- Nombre: Un string que representa al nombre que describe a ese producto
- Marca: Un string que representa la marca del producto
- TipoProducto: Un string que representa la categoría en la que se engloba el producto (Ropa, electrónica, limpieza…)
- Modelo: Un string que representa el modelo de ese producto concreto (Televisor Samsung **UHK2550**)
- Precio: Un double que representa el precio del producto
- TipoEnvio: Un string que identifica si el envío de ese producto es externo o interno
- Valoracion: Un double que almacena la media de las valoraciones de los compradores para ese producto

También tiene las "object properties" siguientes: *almacena* que lo relaciona con Almacén y representa el almacén que contiene ese producto concreto, *cumpleRestriccion* que lo relaciona con Restricciones_Cliente y representa un producto que cumple esas restricciones,

devolucionAsociada que lo relaciona con Devolución y representa a un producto que es devuelto en esa devolución, *devolucionProductoConfirmado* que lo relaciona con Producto_devuelto y representa un producto de una devolución que ha llegado a la tienda, *estaFormadoPor* que lo relaciona con Envio y representa un producto que forma parte de un envío, *pedidoContiene* que lo relaciona con Pedido y representa un producto que forma parte de un pedido/compra, *recomienda* que lo relaciona con Recomendacion y representa un producto recomendado por la tienda.

Pedido

Concepto que representa un pedido de un cliente; es decir, los productos que ha confirmado que va a comprar. Tiene las propiedades: *pedidoContiene* que lo relaciona con Productos, *pedidoDestinadoA* que lo relaciona con GeoCoordinates (schema.org), *pedidoPerteneceA* que lo relaciona con Person (schema.org), *fechaPedido* y *precioTotal*.

Recomendación

Concepto que representa una recomendación del sistema a un cliente sobre los productos en los que puede estar interesado.

Almacén

Concepto que representa un almacén físico. Tiene como propiedades *almacena* que lo relaciona con Producto, *almacenAgrupa* que lo relaciona con Lote, *almacenTransportistas* que lo relaciona con los Organitzation(schema.org) que sean transportistas de ese almacén y *esLocalizadoEn* que lo relaciona con GeoCoordinates(schema.org).

Razón_devolución

Concepto que representa el motivo por el cual el cliente decide devolver un producto previamente comprado en la tienda. Hay 3 tipos de razones de devolución en nuestro sistema:

- Defectuoso: El producto no ha llegado en las condiciones esperadas y presenta defectos.
- Equivocado: El cliente ha recibido un producto que no había pedido.
- No satisfactorio: El producto no ha superado las expectativas del cliente.

Tiene como propiedad *tiene_razon,* que lo relaciona con Devolución y indica que esa devolución tiene esta instancia como razón de devolución.

Restricciones_cliente

Concepto que representa las restricciones que introduce el cliente en el momento de realizar una búsqueda de productos y encontrar solamente productos que cumplan con todas sus restricciones. Tiene como propiedades *cumpleRestriccion*, que lo relaciona con los elementos de Producto que cumplen con la restricción que representa el concepto, *peticionProductosDisponibles,* que lo relaciona con la percepción Peticion_productos_disponibles, y *tieneRestriccion,* que lo relaciona con Cliente y representa al cliente que genera la restricción.

Categorías

Concepto que representa a las distintas categorías de productos de nuestro sistema. Tiene una propiedad llamada *perteneceA* que lo relaciona con el concepto Products (schema.org) y representa a los productos que pertenecen a esa categoría.

Centro_logístico

Concepto que representa al centro logístico de nuestro sistema. Tiene como propiedades *tienePedidos,* que lo relaciona con Pedido y representa el pedido que tiene que ser enviado, y *creaEnvios*, que lo relaciona con Envío y representa el envio que genera el centro logístico.

Devolución

Concepto que representa a una devolución de un producto en nuestro sistema. Tiene las siguientes propiedades:

- *devolucionAsociada:* Lo relaciona con Product (schema.org), representa el producto que tiene que ser devuelto.
- *devolucionContiene:* Lo relaciona con el concepto Factura_de_compra y representa la factura de compra asociada a esta devolución.
- *peticionDevoluciónContiene:* Lo relaciona con la percepción Petición_devolución y representa la percepción que activa ese proceso de devolución.
- *resultadoRevolución:* Lo relaciona con la acción Comunicacion_resultado_devolucion.
- *tieneRazon:* Lo relaciona con el concepto Razon_Devolución y representa el motivo de la devolución.

Envío

Concepto que representa el envío de un pedido en nuestro sistema. Tiene como propiedades *creaEnvios,* que lo relaciona con Centro_logístico y representa el centro logístico que ha generado el envio concreto, y *estáFormadoPor,* que lo relaciona con Product (schema.org) y representa los productos que contiene el envío y que serán enviados al cliente.

Factura_de_compra

Concepto que representa la factura de un pedido en nuestro sistema. Tiene como propiedades *devolucionContiene,* que lo relaciona con Devolución y representa la factura de la compra del pedido que contenía el producto a devolver, y *tieneFactura,* que lo relaciona con la acción Emitir_factura.

Lote

Concepto que representa a un lote, que es un conjunto de productos a enviar, en nuestro sistema. Tiene las siguientes propiedades:

- *almacenAgrupa:* Lo relaciona con el concepto Almacen y representa al almacén que crea ese lote.
- *enviaLote*: Lo relaciona con la acción Avisar_transportista_envio.
- *esEnviadoPor:* Lo relaciona con Organitzation (schema.org) y representa al transportista encargado de enviar ese lote.
- *loteTiene:* Lo relaciona con Product (schema.org) y representa a los productos que forman parte de ese lote.

Oferta_transportista

Concepto que representa la oferta que hace un transportista para enviar un lote. Tiene la propiedad *recibeOferta* que lo relaciona con Almacén y representa al almacén que recibe esa oferta para enviar un lote.

Recomendación

Concepto que representa las recomendaciones que reciben los usuarios sobre productos que podrían interesarles. Tiene como propiedades *recomienda,* que lo relaciona con Product

(schema.org) y representa los productos que se recomiendan, y *enviaRecomendación,* que lo relaciona con la acción Enviar_recomendación.

4.3 Relación de agentes con la ontología

En este apartado explicaremos los diferentes agentes que tenemos en mente implementar con Python. Cada agente utilizará ciertos conceptos definidos anteriormente con tal de que el sistema tenga unas ontologías que puedan entender todos.

Ag Recomendador

Como hemos explicado en el diseño detallado, este agente se dedica a enviar recomendaciones al usuario de manera periódica. Conocerá los conceptos de Recomendación, Producto y Rating para poder construir esta recomendación.

El proceso lo iniciará el evento *Toca_enviar_recomendación*. A partir de este evento calculará a partir de los diferentes usuarios, Productos, Rating y Recomendaciones de otros usuarios la publicidad para el usuario. Una vez haya construido esta publicidad, enviará la acción *Enviar_Recomendación*.

Ag Venta productos

Este agente se dedica a gestionar todo el proceso íntegro de compra de un producto. Conocerá los conceptos de Producto, Pedido, Vendedor y Factura_de_compra.

El proceso se lanzará a partir de una petición externa de Nueva_Compra_producto. El sistema procesará el pedido y en caso de que esté bien definido enviará una acción de Confirmación_Cesta. Al mismo tiempo, enviará un mensaje interno al agente logístico de Solicitud_envio con tal de procesar la compra efectuada.

Entonces, cuando el agente reciba la Confirmación_envio o Confirmación_envio_externo, enviará una acción de Cobrar_pedido. Si el producto enviado es del sistema, emitirá una factura con la acción Emitir_factura. En caso contrario, hará la acción Ingresar_dinero_vendedor_externo.

Ag Almacén

Este agente se dedica a negociar con los transportistas el mejor precio posible de envío.

Conoce los conceptos de Almacén, Lote, Pedidos, Productos y Transportista. El concepto lote le permitirá agrupar diferentes pedidos geolocalizados a fin de realizar menos envíos.

Su plan de trabajo se lanzará cuando reciba un mensaje interno de Realiza_envio con su información asociada. Entonces comenzará el proceso de negocio de precios con los transportistas externos enviando acciones de Pedir_precios_envio y recibiendo en respuesta Precios_envio. Esta respuesta de Precios_envio tendrá asociada una Oferta_transportista. Se decidirá entonces, qué transportista tiene mejor precio y se lanzará el segundo plan gracias al Avisa_transportista_envío.

Al final del proceso se enviará un Confirmación_envío al agente Venta Productos para actualizar el stock.

Ag Logístico

Este agente se dedica a gestionar la logística del sistema.

Conocerá los conceptos de Pedido, Producto, Vendedor, Centro_Logístico y GeoCoordinates (schema.org)

Se pondrá en marcha gracias a un mensaje interno de Solicitud_envio que llegará del agente Venta Productos. Entonces en función del tipo de producto realizará dos acciones diferentes. Si el producto es de un vendedor externo, se ejecutará el segundo plan de Comunicar envío externo. Sino, se enviará la acción un mensaje Realiza_envio para que el agente Almacén pueda realizar su tarea.

En el plan de comunicar envío externo, por una parte enviará un mensaje interno de Confirmacion_envio_externo al Agente Venta productos para actualizar el stock y por otra, realizará la acción de Comunicar_pedido_enviar.

Ag Productos externos

Este agente se dedica a guardar en memoria los productos que los vendedores externos quieren poner en nuestro sistema.

Conocerá los conceptos de Producto_Externo y Vendedor.

El agente se pondrá en marcha cuando reciba una nueva percepción de Add_producto_externo. Entonces confirmará los campos requeridos y actualizará el stock del sistema. Finalmente enviará una acción al vendedor de Confirmación_producto_externo_added para confirmar que su producto ha sido puesto en la tienda con éxito.

Ag Devoluciones

Este agente gestiona todas las devoluciones del sistema.

El Agente debe conocer los conceptos de Devolución, Pedido, Razón_devolución y Producto.

Se lanzará a partir de una percepción de Peticion_devolución. Entonces analizará la razón que lleva asociada y en función del resultado de ésta se efectuará la acción de Comunicación_resultado_devolucion. Cuando el usuario envíe el paquete y el sistema la reciba, obtendrá la acción de Producto_devuelto. Entonces, desarrollará la acción de Devolver_dinero_cliente.

Ag Mostrar Productos

Este agente se limita a hacer queries de productos que puedan interesar a los clientes.

Debe conocer los conceptos de Producto, Categoría, Persona (que representa el cliente) y las Restricciones_cliente, que le servirán al sistema para filtrar los productos.

Recibirá la percepción de Peticion_productos_disponibles con las Restricciones_cliente asociadas y realizará un filtrado de los productos según estas restricciones. Entonces, enviará la percepción de Informacion_productos_disponibles.

5. Implementación del sistema

En los siguientes apartados explicaremos con detalle la implementación explicitado en los apartados anteriores.

5.1 Detalle de la implementación: Directorios

En nuestro sistema hemos implementado un directorio llamado *SimpleDirectoryService*. Este directorio tiene como función principal conocer todos los agentes que están conectados al sistema y que están en funcionamiento. Desde el punto de vista de un agente, sirve para conocer la dirección de otro agente para comunicarse con él. Es por eso, que todos los agentes, en su inicialización, se conectan a este directorio.

Hemos tenido que cambiar un poco la implementación de búsqueda de la versión inicial para que pueda cumplir con las funcionalidades que necesitábamos. El cambio ha sido que en vez de limitarse a buscar un Agente, busca un conjunto de éstos. Esto permitía que cuando un agente pedía un conjunto de agentes, el directorio le devolvía todos los que cumplía con esas características.

5.2 Detalle de la implementación: Agentes

El sistema que hemos implementado en Python implementa los mismos agentes que hemos definido en la fase de diseño detallado, mas 3 agentes que no aparecen en esta fase y que representan a los actores externos de nuestro sistema: el cliente, el vendedor externo y los transportistas. Estos tres actores externos son representados por el Agente Cliente, el Agente Vendedor Externo y el Agente Transportista. Esto es debido a que estos agentes son necesarios para que los actores externos puedan interactuar con nuestro sistema. Además, cada uno de ellos tendrá ciertos parámetros que ayudarán a parametrizar las características de cada actor externo.

Agente Mostrar Productos

Este agente es el que gestiona todo el proceso de consultas de productos en nuestra base de datos, se encarga de proporcionar al Agente Cliente de todos los productos que cumplan las restricciones que especifique.

La implementación de este agente sigue muy de cerca el diseño detallado y la ontología definida con anterioridad. Tal y como se definió en la ontología, recibe mensajes de tipo *Peticion_productos_disponibles* dónde el contenido de ese mensaje es de tipo Restricciones_cliente y manda mensajes de tipo *Informacion_productos_disponibles* dónde el contenido de este mensaje es de tipo Producto. La única diferencia respecto a la ontología inicial es que no conoce el concepto Categoría ya que este concepto no existe en nuestra implementación, ya que es representado por un atributo del concepto Producto y que los mensajes de tipo *Informacion_productos_disponibles* no son explícitamente de ese tipo.
El funcionamiento del agente es el siguiente:

- El agente recibe mensajes FIPA-ACL de tipo *Peticion_productos_disponibles,* el contenido del cual es de tipo *Restricciones_cliente* y que representa el filtro que quiere el cliente para buscar productos tal y como está definido en la ontología.
- El agente trata ese mensaje y obtiene las restricciones del cliente, luego obtiene todos los productos de la tienda almacenados en el fichero **/datos/productos** y los guarda en forma de grafo RDF.
- En el caso de que exista una o más restricciones se procederá a filtrar sobre los productos de este grafo. Para realizar este filtrado se implementa una consulta en SPARQL donde se buscan los productos cuyos atributos coincidan con las restricciones obtenidas. Una vez realizada la consulta se obtendrá el resultado y se generará un grafo RDF que contendrá esos productos.
- En el caso de que no se obtenga ninguna restricción en el mensaje, el grafo a enviar coincidirá con el grafo RDF de productos que se ha obtenido del fichero **/datos/productos**, de forma que se enviarán todos los productos de la tienda sin aplicar ningún filtro (ya que no hemos recibido ninguna restricción)
- Una vez finalizada correctamente la ejecución se devuelve el grafo de productos en formato RDF al AgenteCliente como respuesta a su petición de productos disponibles.

Agente Venta Productos

Este Agente se dedica a procesar los pedidos de los usuarios. La implementación de este agente es muy similar a lo especificado en el diseño detallado. Los cambios mas destacados són que la petición que se llamaba *Nueva_Compra_Producto* se llama en la implementación *Petición_compra,* la no realización de la acción *Cobra_pedido* debido a que no se ha llegado a implementar el cobro de pedidos y tampoco se efectua la accion *Ingresar_dinero_vendedor_externo* por el mismo motivo.

El agente recibe mensajes FIPA-ACL de tipo *Petición_compra* con los productos que el usuario quiere comprar y trata la petición.

Por una parte, el agente guarda el pedido en el històrico de compras (archivo **/datos/compras**) y responde al Agente que le ha enviado la petición con otro mensaje FIPA-ACL de *Confirmación_cesta*.

Por otra parte, de manera **asíncrona**, trabaja con una cola que se va llenando cuando recibe mensajes de tipo *Petición_compra*. Esta cola trata dos tipos de mensaje.

- Tipo *Solicitud_envio*. Cuando la cola recibe un mensaje de este tipo, el agente envía un mensaje FIPA-ACL con el tipo *Solicitud_envio* al Agente Logístico para que trate la logística del pedido. En la respuesta de este mensaje, se sabe la confirmación del envío y si este envío es externo, interno o de los dos tipos a la vez. Cuando recibe la confirmación del envío, el agente envía un mensaje FIPA-ACL de tipo *Emitir_factura* al Agente Cliente con la confirmación del envío.
- Tipo *Nueva_compra*. Cuando la cola recibe un mensaje de este tipo, el agente envía un mensaje al Agente Recomendador para que éste sepa que se ha realizado una nueva compra y pueda tratar las recomendaciones pertinentes.

Agente Almacén

Este agente gestiona el proceso de envío de productos internos, siendo el encargado de encontrar al mejor transportista para realizar el envío de los productos a los clientes.

Difiere de la especificación que le fue dada en la ontología en diversos aspectos, primero de todo el concepto Lote no existe en nuestro sistema, por lo cual es un concepto que el agente no conocerá. En nuestra implementación hemos añadido un mensaje llamado *Contraoferta_envio* que no aparece en la ontología del diseño detallado, este mensaje lo manda el almacén a los Agentes Transportistas y pide una oferta con un menor precio del recibido en *Pedir_precio_envio,* otra diferencia con la ontología del diseño detallado es que el mensaje *Pedir_precio_envio* no tiene contenido de tipo *Oferta_transportista* sino que el contenido simplemente es el precio del envío.

Para ello, una vez recibido un mensaje FIPA-ACL de tipo *Realiza_envio* por parte del Agente Logístico con todos los productos que se deben enviar desde el almacén al cliente, empieza un proceso de descubrimiento de transportistas y una negociación con contraofertas.

El proceso de descubrimiento de los Agentes Transportistas se hace a través de un servicio de directorio que contiene los transportistas que se han registrado en el sistema hasta ese momento. Después de obtener a los transportistas disponibles, se inicia el proceso de negociación en el que el Agente Almacén solicita los precios de envío emitiendo un mensaje FIPA-ACL de tipo *Pedir_precio_envio* y se queda a la espera de las respuestas, para elegir el mejor transportista (en este caso consideramos mejor transportista simplemente aquel que nos ofrece un precio de envío menor). Después de esta primera negociación, el agente Almacén intenta rebajar el precio, enviando un mensaje FIPA-ACL *Contraoferta_envio* indicando el mejor precio que ha encontrado con una ligera reducción.

Finalmente cuando recibe las respuestas a la contraoferta, elige al transportista que ofrece el precio de envio mas bajo, y comunica al Agente Logístico una *Confirmación_envío* a través de un mensaje FIPA-ACL.

Agente Cliente

Este agente representa al actor externo Cliente. Este agente no aparecía en el diseño detallado, pero es necesario en el sistema ya que proporciona la forma para que los actores externos de tipo Cliente puedan interactuar con nuestro sistema.

Tiene implementado cierta interfaz gráfica para que el usuario pueda interactuar con este agente que le representa. Las interfaces gráficas son las siguientes:

- **/busca** sirve para buscar productos y comprar productos de nuestro sistema
- **/devolver** sirve para devolver un producto que ha comprado el usuario. En esta interfaz, muestra todos los pedidos del sistema, esto permite al usuario seleccionar un pedido que no ha efectuado él. Lo que queremos demostrar con este caso es que más tarde, el sistema no le dejará efectuar la devolución si el pedido no es suyo.

A la hora de ejecución hemos añadido ciertos parámetros que puede hacer que éste interactúe de una forma diferente. Son los siguientes:

- **--username <Nombre del usuario>** : permite personalizar el agente y poner un nombre del usuario al agente. Este *username* identificará a partir de su inicialización al cliente y por lo tanto solo podrá efectuar devoluciones de pedidos que ha efectuado este usuario. Si no se especifica el username, el sistema utiliza *generic* como usuario.

- **--recomendado <Acción que quiere efectuar al recibir una recomendación>** : Esta opción permite al agente elegir que acción desea efectuar cuando recibe una recomendación de productos del sistema. Hemos implementado 3 opciones posibles:
    - **0** : No compra el producto cuando recibe la recomendación
    - **1** : Compra todo lo que recibe de la recomendación
    - **2** : Decisión aleatoria sobre la compra de los productos recomendados

El funcionamiento de compra de este agente es la siguiente:

- Cuando el usuario se conecta a **/busca** puede introducir las restricciones que quiere para el filtrado de productos del sistema. Cuando hace *submit*, el agente envía un mensaje FIPA-ACL al Agente Mostrar Productos para que éste pueda devolver los productos con las características que ha puesto el usuario. El contenido de este mensaje son las restricciones del cliente que tienen el nombre de *Restricciones_cliente* en nuestra ontología.
- Una vez, ha enviado el mensaje, el Agente Mostrar Productos le devuelve todos los productos que cumplen esas características. Y con estos productos, el agente muestra otra interfaz con los productos que puede comprar.
- El usuario debe seleccionar los productos que quiere comprar y enviar una segunda petición para hacer la compra. En este caso, el Agente construye un mensaje con los productos que el usuario ha seleccionado y envía una petición al Agente Venta Productos que procesa esta compra.

- En la respuesta de la compra, el agente recibe la cesta de la compra con el precio final. En este punto se ha finalizado la compra.

Por otra parte este Agente puede enviar una petición de devolución de pedidos. Hemos tenido que simplificar el sistema a fin de cumplir con todas las funcionalidades más importantes que tiene nuestro sistema, por eso, en nuestro caso, el sistema **sólo** puede devolver pedidos enteros y no productos parciales de un pedido. El funcionamiento es el siguiente:

- El usuario accede a **/devolver**. El sistema le devuelve las compras que ha hecho este usuario. **Pero en nuestro caso**, solo para demostrar que funciona, nos muestra todas las compras del sistema.

- El usuario selecciona el pedido que quiere devolver. Al hacer submit, el Agente envía una petición de devolución al Agente Devoluciones para procesar ésta. Como respuesta, se le muestra al usuario si la devolución ha sido aceptada o denegada. (por

ahora, se le deniega la compra si el usuario no es propietario de la compra y se acepta si sí que lo es).

Además de estas funcionalidades proactivas, tiene funciones secundarias. El Agente Cliente puede recibir recomendaciones del Agente Recomendador. Cuando las recibe trata el mensaje para saber si es un mensaje FIPA-ACL y si todo es correcto, en función del parámetro que se ha puesto al inicializar el Agente, envía una petición de compra o no.

También es capaz de responder a las peticiones de valoraciones que le envía el Agente Recomendador a través de un mensaje FIPA-ACL *Peticion_valoración* cierto tiempo después de haber realizado una compra. El comportamiento es sencillo, responde con una valoración aleatoria de 0 a 10 para cada uno de los productos que el Agente Recomendador le pide valorar (todos los de la compra realizada).

Después de una compra también es capaz de recibir mensajes FIPA-ACL de tipo *Emitir_factura* del Agente Venta Productos. Este mensaje es recibido de manera asíncrona y se muestra por terminal la factura.

Agente Devoluciones

Este Agente gestiona todas las devoluciones de las compras efectuadas por los clientes.

La implementación de este agente difiere bastante a la de la especificación en el diseño detallado. Las diferencias principales son que razon_devolución es un concepto que no existe en nuestra implementación, por lo tanto no puede ser conocido por el agente, también tenemos que la *Petición_devolución* no tiene asociada ninguna razón de devolución, ya que no existe, y por lo tanto es aceptada o denegada en función de si la compra ha sido realizada por él o no. Toda la parte de devolver el producto a la tienda y devolver el dinero al cliente tampoco está implementada por falta de tiempo, por lo tanto aquí tampoco se sigue la ontología especificada por el agente.

El agente funciona de la siguiente manera, cuando recibe una petición de devolución con la compra que quiere devolver el cliente, verifica y valida que sea un mensaje FIPA-ACL y después decide si la devolución es aceptada o denegada.

La decisión, por ahora, se basa en la verificación de si el cliente es propietario de esa compra, ya que en el mensaje FIPA-ACL, el cliente le envía el nombre del usuario de su agente. Si el

usuario es el propietario de esa compra, el Agente Devolución le responde con otro mensaje FIPA-ACL con el *ResultadoDevolución*.

Si la devolución es aceptada, el agente escribe la devolución en la el fichero **/datos/devoluciones** donde se guarda un historial completo de todas las devoluciones.

Agente Logístico

Este agente gestiona la logística del sistema. Está implementado de forma que difiere un poco de la ontología usada para especificar al agente.

En primer lugar el concepto GeoCoordinates no existe en nuestra implementación, de forma que el agente no puede conocer el concepto, y en segundo lugar solo existe un almacén, de forma que siempre le mandará el mensaje al mismo Agente Almacén sin consultar la proximidad del envío. Los mensajes que recibe y manda el agente si que contienen la misma acción que los mensajes especificados en la ontología.

Tiene una cola de trabajo para procesar los productos que no pertenecen al sistema (productos externos). Recibe peticiones con FIPA-ACL con acción *Solicitud_envio* del Agente Venta Productos. Cuando recibe esta petición valida el mensaje FIPA-ACL y procesa los productos que van incluidos.

Si en el mensaje solo hay productos de nuestra plataforma, el Agente construye un mensaje FIPA-ACL con los productos para enviarselo al Agente Almacén. Cuando recibe la respuesta de este agente, responde al Agente Venta Productos con la confirmación del envío.

En el caso de que en el mensaje tenga incluidos productos externos, encola un mensaje en la cola de trabajo para que se envíe de manera asíncrona. En esta cola, cuando recibe mensajes, se envía un mensaje al Agente Vendedor Externo con el producto incluido para que sea éste agente el que trate este envío. Este mensaje es de tipo *Avisar_vendedor_externo_envio*.

En el tercer caso que hay productos de dos tipos (internos y externos), el Agente envía un mensaje al Agente almacén y encola un mensaje para que sea enviado de manera asíncrona al Agente Vendedor Externo.

Agente Productos externos

Este agente se dedica a guardar los productos que los vendedores externos quieren poner en nuestro sistema. La forma en la que hemos implementado este agente coincide exactamente

con la descripción de la ontología. El agente recibe mensajes de tipo *Add_producto_externo* donde el contenido del mensaje es de tipo *Producto*, obtiene ese producto del mensaje y lo guarda en nuestra base de datos. Una vez ha realizado la acción devuelve el mensaje *Confirmacion_producto_externo_added*.

El funcionamiento del agente es el siguiente:

- El agente trata mensajes FIPA-ACL de tipo *Add_producto_externo* que le manda el Agente Vendedor Externo. Cuando recibe una petición de este tipo obtiene la información del producto que contiene el mensaje y lo guarda en un grafo RDF.
- Obtiene del fichero **/datos/productos** todos los productos disponibles en forma de grafo y hace la unión de los dos grafos de forma que se añade el nuevo producto.
- Guarda este grafo nuevo en el fichero **/datos/productos** de nuestro sistema para actualizar la información de los productos con el nuevo producto para que los otros agentes que consulten esta fuente de datos puedan obtener la información del nuevo producto.
- En caso que el proceso se ejecute de forma correcta, genera un mensaje FIPA-ACL de tipo *Confirmación_producto_externo_added* y lo envía al Agente Cliente. Este mensaje pero no es tratado por el Agente Cliente en esta implementación del sistema así que no tiene utilidad, pero para una futura versión del sistema podría ser útil.

Agente Recomendador

Este agente gestiona los procesos de obtención de valoraciones de productos por parte de los clientes que los han adquirido y el proceso de recomendación proactiva de productos que pueden resultar interesantes a los clientes.

La implementación de este agente difiere bastante de la ontología ya que hemos añadido la funcionalidad de pedir Valoraciones además de enviar recomendaciones. El mensaje *Toca_enviar_recomendación* no existe en la ontología, sino que es un timer que tiene el agente que hace recomendaciones cada X tiempo. El concepto rating se llama valoración en nuestro sistema y la forma de elegir las recomendaciones no se hace teniendo en cuenta las valoraciones, se hace buscando productos de categorías que coincidan con el historial de búsqueda del cliente.

La obtención de valoraciones se hace también de forma proactiva. Cuando un cliente realiza una compra, el AgenteVentaProductos le envía un mensaje FIPA-ACL indicando *Nueva_compra*. Cierto tiempo después, el agente recomendador decide que es hora de

obtener las valoraciones de los productos de esa compra, y envía un FIPA-ACL al agente cliente indicando que se trata de una *Petición_valoración* junto con los productos que quiere que el cliente valore. El cliente responde con una *Nueva_valoración* que contiene sus valoraciones de los productos que había comprado.

La recomendación se hace eligiendo los productos candidatos a ser comprados por el cliente, esta decisión se toma consiguiendo los tipos de productos que ha comprado hasta el momento el cliente, y seleccionando los productos de esos tipos que aún no han sido adquiridos por el usuario. Para no colapsar al cliente con muchos productos, se elige uno solo de forma aleatoria y se le envía a través de un mensaje FIPA-ACL indicando que és una *Recomendación*.

Agente Transportista

Este agente representa a los actores externos Transportista. En nuestro diseño detallado no aparece este agente pero es necesario para representar a cada actor externo de tipo transportista que quiera registrarse en nuestro sistema para recibir ofertas de envíos. Hemos implementado de manera que no estén prefijados, sino que se registran en un servicio de directorio desde el cual serán descubiertos por los agentes que lo necesiten, en nuestro caso únicamente el Agente Almacén.

Se trata de un agente personalizable a partir de los parámetros de entrada que hemos implementado. Los parámetros posibles son los siguientes:

- **--ntp <Número>**: es el número de transportista. Sirve para identificar al transportista.
- **--precio <precio>**: es el precio que tiene el transportista cuando hace un envío.
- **--contra <contraoferta>**: es el valor en el que reducirá la contraoferta para poder hacerse cargo del envío. Si no se pone ninguna, se pone por defecto a 0, es decir, que la contraoferta es el mismo precio definido anteriormente.

Al crear un nuevo transportista, se le debe asignar un número de transportista utilizando el parámetro **--ntp**. Cuando recibe una petición de precios de envio, responde con el precio indicado a través del parámetro **--precio**. Es posible que deba tratar con una negociación más compleja con contraofertas. Por esa razón tiene el parámetro de **--contra**.

Este agente puede tratar dos tipos de mensajes FIPA-ACL: *Pedir_precio_envio* y *Contraoferta_envio*.

En el primer caso, el agente responde con los precios que se le ha fijado en la inicialización. El contenido con el precio tiene el nombre de la ontología que hemos definido de *Precios_envio*.

En el segundo caso, donde el agente debe responder con una contraoferta, el agente responde un mensaje FIPA-ACL con la contraoferta calculada a partir de los parámetros que hemos configurado en su inicialización.

Agente Vendedor Externo

Este agente representa al actor externo del sistema “Vendedor Externo”. Este agente no estaba definido en nuestro diseño detallado pero es necesario ya que sino el vendedor externo no tendría ninguna forma de interactuar con nuestro sistema. Cuenta con una interfaz

gráfica para introducir la información de los productos que quieren poner a vender y para seleccionar el tipo de envío que quieren para ese producto. A esta interfaz se accede con la ruta **/vende.**

El funcionamiento del agente es el siguiente:

- Cuando el usuario que representa al vendedor externo se conecta a **/vende** le aparece un formulario donde puede introducir los atributos del producto que quiere vender. Concretamente el vendedor introduce el nombre del producto, su categoría/tipo de producto, su precio, su modelo y su marca.

- Con los atributos siguientes definimos a un producto en nuestro sistema, el vendedor también puede decidir qué tipo de envío quiere para sus productos: Interno, donde será la tienda la que mande el producto desde su almacén, o Externo, donde se le comunicara al vendedor de que tiene que realizar un envío de ese producto y la tienda ejercerá sólo de intermediaria.

- Una vez el vendedor ha introducido todos los campos del formulario le da a la opción “Poner a la venta”. El agente comprobará que todos los campos del formulario tengan valor antes de comunicarse con el Agente Productos Externos, en caso de que algún campo no tuviera valor nos mostrará un mensaje de error avisando de un error en la introducción. En el caso de que todo esté correcto, se comunicará con el Agente Productos Externos.

- La comunicación con el Agente Productos Externos se realizará enviando un mensaje FIPA-ACL a este agente con la acción *Add_productos_externos* asociada. En este mensaje también se mandará el producto a enviar en forma de grafo RDF.

- Una vez mandado el mensaje, en nuestro sistema se muestra directamente el mensaje de confirmación de la puesta de producto a la venta y no se espera a una confirmación del Agente Productos Externos.

6. Nivel avanzado

En lo referente a la implementación avanzada, hemos conseguido cumplir con los tres objetivos que nos resultaron más interesantes:

- **Agentes transportistas dinámicos:** modificamos las operaciones de búsqueda de agentes para poder obtener todos los agentes que pertenecen a un tipo en concreto. Esto nos ha servido tanto para poder encontrar todos los clientes conectados en el sistema como para poder descubrir los transportistas registrados. Esto permite que se puedan ir añadiendo nuevos agentes sin tener que modificar el código, simplemente ejecutando un nuevo transportista con los parámetros que deseemos, podremos utilizarlo sin necesidad siquiera de un reinicio de los demás agentes. Nuestra implementación nos permite tener un único directorio donde estén todos los agentes, de cualquier tipo. Por otra parte, dado que hacer un directorio únicamente para transportistas és muy sencillo y teniendo en cuenta que no tendremos excesivos transportistas, hemos obviado la implementación de ésta.

- **Negociación compleja con transportistas:** hemos realizado un sistema de negociación en tres pasos, en el que primero se reciben los precios de cada transportista, luego se les intenta rebajar el precio con una contraoferta y finalmente elegimos al transportista más barato. Hemos añadido un parámetro a los transportistas para determinar su comportamiento ante las contraofertas y otro para determinar su precio inicial, por lo que se puede tener que un transportista sea el más caro en una negociación simple pero al realizar la contraoferta se vuelva la mejor opción.

- **Petición de valoraciones y recomendación proactiva:** implementamos el Agente Recomendador, que nos ha permitido obtener las valoraciones de los productos que ha comprado un cliente después de cierto tiempo de haberlos adquirido. Habíamos pensado en hacer que el cliente no conteste siempre a las valoraciones pero al final decidimos que queríamos obtener siempre las valoraciones de las compras.

También recomendamos productos a todos los clientes registrados en el sistema cada cierto intervalo de tiempo, para ello hemos utilizado un planificador de tareas. Hemos intentado hacer una aproximación simple a la recomendación de productos de interés para cada cliente, haciendo que se les recomienden productos que aún no tienen de los tipos/categoria que ya ha comprado.

7. Repartición de tareas

La repartición de la carga de trabajo que ha supuesto la realización del proyecto se ha hecho de forma equitativa desde el principio del proyecto.

En la primera fase de diseño, cada miembro ha podido hacer una parte diferente, permitiendo obtener un sistema con ideas de cada uno de los integrantes del proyecto. Para poner ideas en común hemos utilizado las clases de laboratorio donde el intercambio de ideas se ha hecho de manera recurrente.

En la segunda fase de implementación del sistema, el objetivo ha sido paralelizar las diferentes tareas con tal de avanzar el proyecto de la manera más rápida y efectiva posible. La definición final de la ontología así como una comunicación fluida entre los compañeros ha permitido que el progreso del proyecto no haya sido estancado en ningún momento.

Aún así también hemos buscado la eficiencia y la comodidad teniendo en cuenta que fragmentar y dividir el trabajo en tareas demasiado pequeñas podría afectar negativamente a la velocidad del desarrollo del proyecto.

Hemos empezado implementando el proceso de compra y el envío de productos, y ha sido muy útil tener bien definida la ontología que íbamos a utilizar, ya que a pesar de que ha ido variando con nuevos conceptos o eliminando algunos conceptos, nos ha servido para poder desarrollar de forma paralela sin conflictos ni problemas incluso con agentes con un gran componente comunicativo, como podrían ser el AgenteVentaProductos con el AgenteCliente o AgenteLogístico.

No podríamos decir exactamente qué miembro ha implementado qué agente ya que, aunque hemos realizado una repartición de funcionalidades bien definida, siempre hemos aceptado las mejoras propuestas por nuestros compañeros de proyecto. Creemos que las reuniones han sido clave para poder realizar un proyecto de esta envergadura y complejidad.

8. Evaluación crítica

Una vez realizada la práctica de principio a fin, procederemos a evaluar los resultados obtenidos de nuestra implementación. Consideramos que hemos implementado una solución válida pero que todavía tiene puntos débiles que podrían mejorar.

En primer lugar, consideramos que las funcionalidades principales descritas en el enunciado de la práctica han sido implementadas correctamente. No obstante, hemos tenido que asumir ciertas hipótesis con el fin de que la complejidad del proyecto no creciera de manera exponencial.

En segundo lugar, el hecho de ofrecer ciertas interfaces gráficas de usuario permiten tener una interacción que se aproxima un poco más a la realidad. Sin embargo, han sido interfaces simples sin estilos ya que el proyecto no se centraba en éstas.

En tercer lugar, cabe destacar que la implementación de tres funcionalidades avanzadas en nuestro sistema añade un valor y complejidad a nuestra plataforma. Por otra parte, la implementación del directorio permite a todos los agentes conocer los tipos de agentes con los que se comunica de forma dinámica.

Por otra parte, también consideramos que nuestro sistema no es perfecto. Una de las debilidades que encontramos es el hecho de que el Agente Logístico no organice paquetes antes de comunicarlos al Agente Almacén. También ciertas hipótesis como que un usuario solo puede devolver una compra efectuada entera y no sólo ciertos productos de éstos. En este último caso, hemos estado pensando la implementación pero finalmente concluimos que complicaba mucho la complejidad del sistema. También hemos obviado ciertos conceptos de la Ontología planeada para simplificar todo el sistema como es la geolocalización y el envío de productos desde el Almacén más cercano. Por estos detalles consideramos que el sistema podría considerarse limitado.

Finalmente, después de haber trabajado con un lenguaje nuevo, una metodología diferente y un nuevo protocolo de comunicación, consideramos que la experiencia ha sido muy positiva. Hemos podido entender la importancia de la definición estática de los conceptos con los que íbamos a tratar así como la definición y repartición de tareas en módulos diferentes. Creemos que ésta metodología de trabajo permite conceptualizar de manera clara y concisa un sistema distribuido muy escalable.