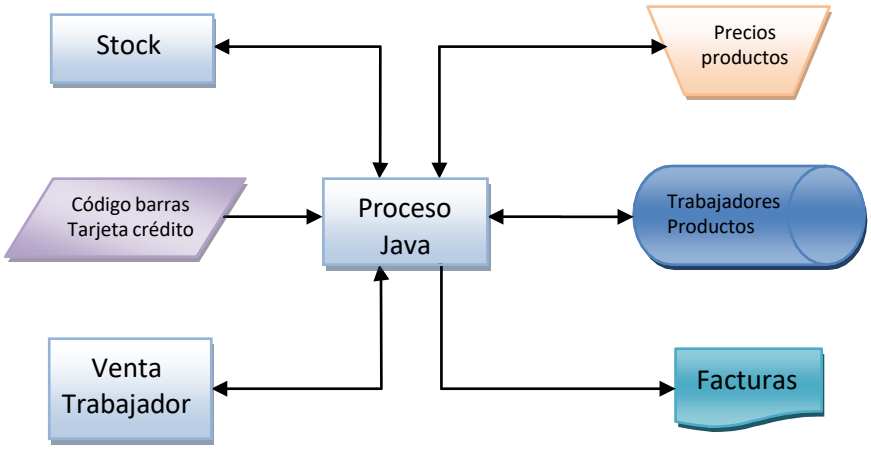
La empresa BK desea una aplicación que no requiere de demasiados cambios y que posea una estructura lo suficientemente rígida, y con unos requisitos lo suficientemente claros como para optar por un modelo de ciclo de vida de los denominados ***en cascada con retroalimentación***.

Como una de las premisas es la utilización de software libre, decidimos desarrollar el proyecto con lenguaje de programación Java, el cual, al ser un lenguaje orientado a objetos, nos dará la suficiente libertad para poder crear una aplicación lo bastante amplia para abarcar cualquier requerimiento solicitado por el cliente.

Así mismo, para poder realizar dicho proyecto, nos apoyaremos en una herramienta muy eficaz y que nos permite trabajar con lenguaje Java, y que, aunque es software propietario, igualmente es gratuito y libre, como JDeveloper de Oracle.

Tanto el lenguaje Java como el paquete JDeveloper lo podemos descargar desde la web principal de Oracle, y tras su instalación, ponernos a trabajar desarrollando la aplicación solicitada.

Para comenzar con el proyecto nos creamos un pequeño esquema indicativo de los distintos requerimientos solicitados:



Como **requisitos funcionales** tenemos los siguientes:

- Almacenar los datos del trabajador:
  - DNI
  - Nombre
  - Apellidos
  - Nº Seguridad Social
  - Fecha de nacimiento
  - Teléfono
  - Localidad
- Almacenar información de los productos:
  - Código
  - Marca
  - Nombre comercial
  - Precio
  - Cantidad
- Proporcionar facturas de las ventas.
- Llevar la cuenta de lo que vende cada trabajador.

- Controlar el stock de productos en almacén.
- Operar con lector de código de barras y tarjetas de crédito.
- Controlar los precios de los productos y ofrecer la posibilidad de operar con ellos.

Y como **requisitos no funcionales**:

- Reducir el tiempo de respuesta de la aplicación
- Imposibilitar el proceso simultáneo de peticiones aunque haya varios equipos funcionando al mismo tiempo

Tras estas premisas iniciales, planificaremos una serie de reuniones con el cliente para poder perfilar con más detenimiento cada uno de los requerimientos de la aplicación a diseñar, como por ejemplo:

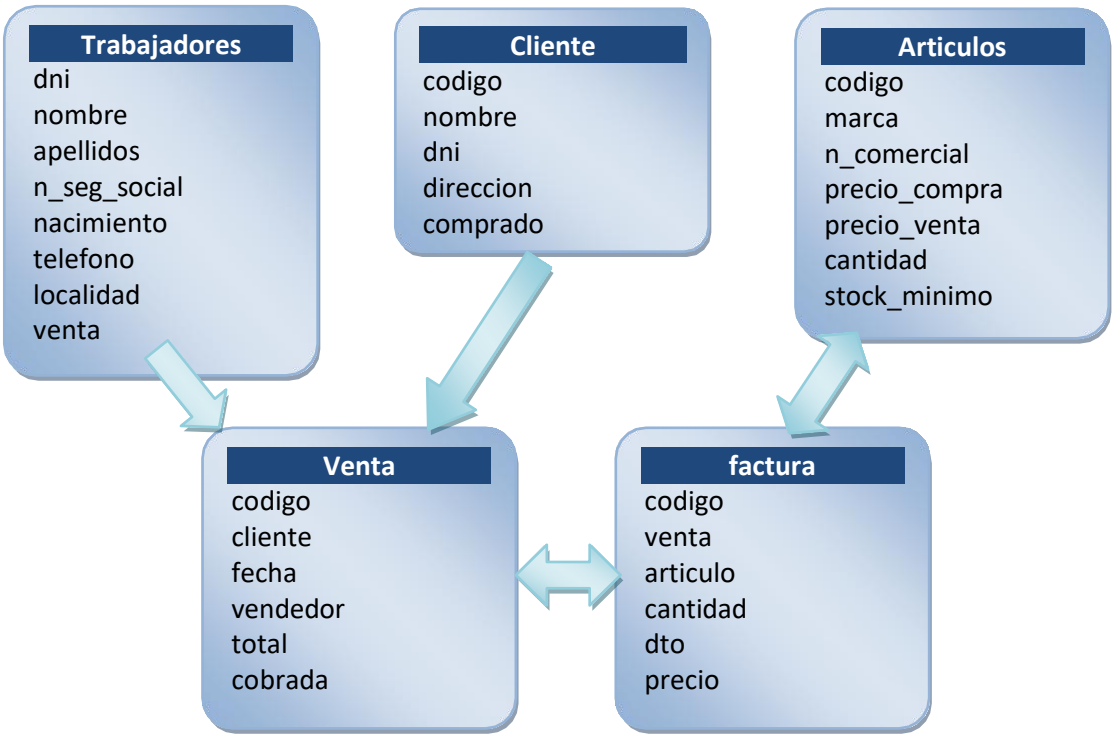
- Si necesitamos controlar el stock de almacén, deberemos añadir un campo más para incluir el stock mínimo y conseguir que el programa nos avise cuando se vaya a rebasar (***cantidad*** menor o igual que ***stock mínimo***).
- Al necesitar emitir facturas, necesitamos igualmente incluir los datos de los clientes a los que irán dirigidas, y el código del vendedor que realizará la venta para poder controlar cuántas transacciones ha realizado dicho empleado.
- Para poder admitir tarjetas de crédito como pago, tendremos que poner un campo nuevo en la factura que permita indicar la forma de entrega económica y opte por simplemente indicar qué cantidad se abona en metálico o que se va a hacer uso del mencionado dispositivo magnético para realizar el pago.
- Los artículos que se vayan a vender se obtendrán de la tabla de artículos pero se controlarán en una nueva tabla que recogerá en cada registro la factura a la que pertenece, la cantidad comprada, el código del producto, el precio e incluso un campo que denominemos descuento por si en cualquier momento tuviéramos que realizar una rebaja en un producto determinado dentro de una operación estipulada.

Cuando finalicemos con el refinamiento de todas estas premisas gracias a las reuniones con los encargados de la empresa y empleados que vayan a manejarla, sólo nos quedará comprobar cuáles son los equipos que se van a utilizar y si será necesario ampliar o modificar el parque informático de dicha empresa.

Con todo ello terminamos con la fase del **Análisis de Requisitos** y tendremos que enfocar el problema en base a un diseño eficaz, para lo cual podemos basarnos en el esquema realizado anteriormente con el objetivo de dividir dicho sistema en partes, determinando la función que realizará cada una de ellas de forma inequívoca.

Crearemos una base de datos con SQL para poderla utilizar con Java mediante JDBC.

Tenemos dos entidades ya definidas con claridad como son ***Trabajadores*** y ***Artículos*** aunque, como vimos anteriormente, tendríamos que crear otras como son ***Ventas*** con la que podríamos obtener los artículos vendidos, las facturas a emitir o ya emitidas, los trabajadores que han realizado la venta, etc. ***factura*** en la que se grabará toda la información individual de cada artículo vendido, y ***Cliente*** para poder identificar a quién se le realiza la venta e ir controlando las adquisiciones realizadas.



Con ello terminamos la fase del **Diseño** y comenzamos con la **Codificación**, y dado que el cliente ha decidido utilizar software libre, procederemos a la creación de la aplicación utilizando el lenguaje Java y apoyándonos en el entorno de desarrollo JDeveloper, el cual usaremos para el diseño de la base de datos, sus tablas, relaciones, y toda la aplicación en general, obteniendo los ficheros objeto en esta fase de codificación, el cual podremos utilizar para realizar las ***pruebas unitarias*** gracias a JUnit que es el entorno de pruebas para Java.
Una vez realizada con éxito este conjunto de pruebas unitarias, comprobaremos el funcionamiento de todo el sistema con todas sus partes interrelacionadas, para lo cual contactaremos con el cliente con el fin de poder implementar en sus equipos una versión *Beta* que nos permita probar su desarrollo en el entorno donde se implantará definitivamente.

Cuando el conjunto de todas estas pruebas haya finalizado con éxito, podremos comenzar a crear la ***documentación*** donde especificaremos todas y cada una de las etapas realizadas en el desarrollo del software, es decir, tendremos que entregar tres guías al cliente:

- ***Guía técnica:*** Dirigida a analistas y programadores, y que nos permitirá realizar correcciones y mantener la aplicación en un futuro. Incluiremos el diseño de la aplicación, la codificación de los programas y las pruebas que se le han realizado.
- ***Guía de uso:*** Dirigida a los usuarios finales o clientes. Describiremos la funcionalidad de la aplicación, cómo comenzar a ejecutarla, ejemplos de uso, requerimientos software y solución a los posibles problemas que se puedan presentar, es decir, intentaremos dar a los usuarios finales toda la información necesaria para utilizar la aplicación.
- ***Guía de instalación:*** Incluiremos toda la información necesaria para la puesta en marcha y la explotación, así como la seguridad del sistema. Esta información la dirigiremos al personal informático responsable de la instalación, en colaboración con los usuarios finales (clientes). Con ella intentaremos dar toda la información para garantizar que la implantación de la aplicación se realice de forma segura, confiable y precisa.

Comprobado que el software es fiable, carece de errores y terminado el proceso de documentación, procederemos a dar a conocer la aplicación a los usuarios finales para que comiencen a utilizarla, o lo que se denomina ***fase de explotación***, para lo cual haremos que los futuros clientes estén presentes en el proceso de instalación y le iremos comentando el proceso, realizando las Beta Test en los equipos

del cliente y con carga de trabajo normales (como el software es “a medida”, nosotros realizaremos la instalación).

Para finalizar con el ciclo de vida del software que hemos creado procederemos a pactar con el cliente un servicio de ***mantenimiento*** de la aplicación que pueda conllevar una futura mejora de la funcionalidad del software, nuevas necesidades, expansiones o incluso modificaciones y actualizaciones para adaptarse a las nuevas tendencias del mercado, o al nuevo hardware que pueda adquirir el cliente.

Se le ofrecerá al cliente un listado con los distintos precios de dicho mantenimiento, ya que no cuesta lo mismo añadir una tabla más para controlar los proveedores, que modificar la aplicación para que admita entrada de datos a través de una red externa que incluiría un extra de seguridad.

Le ofreceremos un precio por un mantenimiento ***perfectivo*** (que mejorará su funcionalidad), ***evolutivo*** (modificaciones o expansiones para futuras nuevas necesidades), ***adaptativo*** (modificaciones o actualizaciones que se adapten a las nuevas tendencias del mercado o nuevos componentes hardware)y ***correctivos*** (para errores que pueda tener la aplicación en el futuro).