Domain Model comments

The domain model that was provided in the project description was needed to be modified and added more classes and fields.

Initially a new class, SalesOrderLine was provided. It was a crucial implementation, because the process of the order works as follows:

The selected product is added to the SalesOrderLine where a quantity field is implemented to set the amount of each selected product. Exactly one product can be added to a SalesOrderLine and zero or many SalesOrderLine can contain the same product.

Afterwards, one or many SalesOrderLine can be added to the SalesOrder and we have exactly one SalesOrder per order.

Next, some of the multiplicities given were needed to be changed. The relation between SalesOrder and Invoice looks as follows: A SalesOrder can have only one invoice and an invoice should be connected only to one SalesOrder.

The relation between Supplier and Product also needed to be modified slightly: A Supplier can supply more products and one product can have many suppliers.

Some additional fields were provided to the classes:

* In the Customer class, an email field was provided, because customers have the possibility to make an order by sending an e-mail. An isCompany field was also provided in order to register if a customer is a private person or company (in this case, mostly clubs).
* In SalesOrder a discount and a deliveryFee attribute was added to store the value of additional fees and discounts for the price calculation methods.
* In the Product class the subclasses of item types (Clothing, Equipment and GunReplicas) were kept and an additional field – inStock – was implemented, which stores the amount of products that are available in the store. Before that point, there was no record of the current amount, only the level of minimum stock (minStock), which is responsible to store a value in order to compare it with the inStock value. If inStock is below minStock, an adequate method calls for an automatic refill for the given product from the given supplier.

Transformation into relational model

The Domain Model needed to be transformed into the relational model in order to have a better understanding of how to implement the information into the SQL-script and queries. The transformation was made “the nice way”, meaning that we put the closely related attributes into the same table and the less related ones into a separate table. The principle should be that all the attributes depend on the primary key and the primary key only. Other close relations should be put into a separate table (normalization). Typical example is the zip code, city, country relation. The customer class contained the field zipC and City, which are closely related. Therefore a new table, Country was created to improve the quality of the relational model.

Between the Supplier and Product tables we have many to many relations, therefore a new table ProductSupplier was created with dummy IDs in order to make the connection between the tables.