CS 5121 GROUP PROJECT

SYSTEM DESIGN FOR AN ONLINE TEA STORE IN UML

GROUP No. : 8

MEMBERS : GIANNI BAMBINO - ID 15184943

THOMAS KIRBY - ID 8365407

SUBMITTED : NOVEMBER 23rd 2015

CONTENTS **PAGE**

1.1 INTRODUCTION

The selected domain for this system design project is an online shop for quality unblended teas.
Other more familiar domains, e.g. film DVDs, classical/jazz CDs and niches within the book trade were considered. But each of these was eventually rejected due to :-
1) The high level of competition in these market sectors, especially in film DVD sales where effectively free distribution via 'torrent' groups is destroying existing video distribution businesses every day.
2) The difficulty in gauging actual, i.e. enduring, consumer demand for what are for most of us whimsical purchases whose consommation may be easily diverted by a skilful shop assistant or soon forgotten if frustrated in easily obtaining it.
3) The tendency of people setting up businesses in these domains to let their decisions be guided by their own tastes and preferences rather than by cold commercial criteria – a principal cause of the failure of many of them with resulting debt and distress to their owners..
4) The current depressed state of consumer confidence – which would view such expenditure as unnecessary and even wasteful given the need of householders to plan sensibly for their future.

In pondering what to base this project on, a team member absentmindedly drifted into his usual evening time ritual of preparing a mug of jasmine flower tea . . . and there the idea was born.
Make no mistake about it but tea is addictive, both physically and psychologically
Its consumption is ubiquitous throughout the world. And it is fair to say that Irish people can hold their own with any nation for tea-drinking. It is something our parents and grandparents grew up with; and something our own generation – despite the surge in fashionable coffee drinking from the late '60s on – has by and large maintained. For our own generation, who witnessed the first vegetarian and wholefood restaurants spring up in Ireland, tea drinking patterns have become more enriched by the slow but steady growth of non-African/Indian teas for both their flavour and health benefits. On the negative side it has to be said that tea – like other foods – has been a victim of the drive towards convenient preparation : over 80% of teas on display in supermarkets today are in the tea-bag form and almost all leaf teas are back tea blends.

So the base market for quality tea is there in this country. Native Irish consumers need to be educated a bit at first. They need to be made question the sense of buying those black tea blends that hold premium position in the tea segment for decades – blends that are themselves increasingly being 'cut' with cheap grades. For non-natives, no such education is needed. There is an estimated 60,000 – 80,000 Chinese people living in Ireland. As well as this there has been an increasing influx of people from Southeast Asia

into Ireland over the last 15 years. These people are much more selective in their choice of tea and frequently echew the supermarket brands to get their stock from those unblended teas that are available in Asian food shops here. Unfortunately even these sources do not always provide the best quality Chinese teas as they are usually prisoners of their suppliers as far as choice goes.
The past 5-10 years has seen a small number of online tea shops set up in Ireland. But their selection of offerings is quite poor – and the prices, e.g. €5 per 100g of jasmine tea, are outrageous. An online retailer of quality unblended Chinese teas offering fair prices would have no serious competition for either quality or price in Ireland today.

1.1.2 Business Model

So the business model would essentially consist of sourcing quality unblended Chinese leaf teas from their regions of origin. Initially only a small, i.e. no more than 20 varieties in both 125g and 250g packets, range would be sold from a website heavily contented with information and videos on each. A users' blog – possibly a forum – would be included to enable members to exchange knowledge and ideas on tea varieties, as well as to gauge market demand for them. Initially no trade selling would take place off the website. But large buyers online could arrange for both discount and even credit would be possible subject to the usual trade criteria. Given the oriental bias of present Chinese tea consumers in this country, it would be essential to have the home and business webpages of the website at least to be available in Mandarin to expedite sales to these people as they are the likeliest first customers.

The vast bulk of sales would be expected between Thursday and Sunday of each week – when most do their shopping. Thus the balance of advantages in relation to online payments seems to go to Stripe with their 'Weekly(Wednesday) – 2 business days rolling' payment option. Secure retention of customers' card data is also managed directly by Stripe, relieving the online shop of this obligation.

Realistically, this sort of enterprise would begin as a spare-time job for perhaps a second generation Chinese-Irish business graduate reasonably fluent in spoken Chinese, familiar with Chinese teas and with a friend to help with web engineering aspects. The 'tea warehouse' would be a box room in the owner's home. Delivery of orders would be by An Post or other national courier.

1.2 CONTRIBUTIONS OF TEAM MEMBERS

For the initial system requirements definition both team members worked together to define them.
Likewise with the use case diagram and its subsequent modifications as the project progressed.
Otherwise task allocation amongst the team members was generally based on past experience or express interest. Given his past experience with databases, Mr Bambino was most suitable for the database system design.
The drawing up of class and high-level architecture diagrams was allocated to Mr Kirby who had more experience with object-oriented languages.
CRC cards, communication, sequence and state diagrams was undertaken by Mr Bambino.
Both team members contributed to the project life-cycle discussion.
Mr Bambino proofed and edited as required the final report which was assembled by Mr Kirby.

2. PROJECT LIFE-CYCLE & MANAGEMENT

We assumed the full system would be implemented within six months. As it is for a retailer it is desirable to have basic functionality by Christmas. Therefore, we decided that the project should follow one of the **Agile methodologies**. These programming-centric methodologies have few rules and practices, all of which are fairly easy to follow. We chose **Extreme Programming (XP) methodology** for this project, because it is good for:

- Very small projects with highly motivated, cohesive, stable, and experienced team;
- Projects with enough analysis and design documentation;
- Confident users and developers with web technology;
- Conceptually simple projects;
- Projects with a short time schedule.

Extreme programming (XP) is founded on four core values: communication, simplicity, feedback, and courage. XP uses the key principles of continuous testing, simple coding performed by developers, and close interactions with end users to build systems very quickly. After a superficial planning process, projects perform analysis, design, and implementation phases iteratively.

Testing and efficient coding practices are core to XP. In fact, each day code is tested and placed into an integrative testing environment. If bugs exist, the code is backed out until it is completely free of errors. XP development relies heavily on refactoring. Development proceeds in very short iterations, typically 1-2 weeks in duration. Prior to each iteration features are broken down into small use cases. Those are estimated by developers and then chosen by a client based on their cost and business value. This methodology guarantees that the core of the system, i.e. ordering and payment features is delivered quickly and the rest added later as revenue allows.

We estimate that 1 developer, proficient in database and web applications, is needed to deliver the system. This person will report to a single project manager. With a tight deadline, regular meetings with the project manager would be needed. Developer incentives would also assist here.

A CASE tool combining design, development and testing capabilities would expedite the work and facilitate supervision by the project manager. As with all projects, the project deliverables list should include the system request, feasibility analysis, initial workplan, staffing plan, project charter, standards list and risk assessment.

3. SYSTEM REQUIREMENTS

3.1 FUNCTIONAL REQUIREMENTS

3.1.1 Maintaining System Information

1.1 The system will employ a database that will be mounted on a web server and which holds:

(1) Stock data; (2) Sales data; (3) Customer data; (4) Website content.

1.2 The system will allow remote connection to the server-mounted database from the IT system at the company premises.

1.3 The system will enable updates between the company's database and the server database for: (1) Stock additions; (2) Price changes; (3) Stock item deletion; (4) Customer data extraction; (5)Maintaining website content.

3.1.2 Online Sales Process

2.1 The system will allow the website (both the tea shop products and promotional information web pages) to be viewed by all internet users.

2.2 The system will allow users of the tea shop website to submit information that, after validation, allows them to become registered users of the online tea shop.

2.3 The system will allow registered users to submit order information for tea items they wish to buy. When an order is dispatched the status of the Registered User is raised to that of Customer.

2.4 The system will provide means to validate order information and confirming payment acceptance before confirmation of the goods order.

2.5 The system will notify each registered user/customer when an order has been confirmed and payment successfully made. The notification will be both online and by email.

2.6 The system will automatically update both the stock and the customers data on the server database.

2.7 The system will allow buyers too cancel their orders within 30 minutes of its confirmation.

3.1.3 Website Content

3.1 The system will have an appropriately organized content management system (CMS) for the various information sheets, videos, blog entries, customer queries,etc. that the website will need to engage with its online customers. The shop manager will be able to add or remove items of this.

3.2 NON-FUNCTIONAL REQUIREMENTS

4.1 The system will make provision for integrity, persistence, concurrency and redundancy to the extent expected for a database used for an online shop handling ~ 1,000 simultaneous users.

4.2 The maximum response time of the system will be 7 seconds.

4.3 The system will be available 24/7. Maintenance of the system will be done using a copy on a backup server.

USE CASE DIAGRAM

4. HIGH-LEVEL ARCHITECTURE OVERVIEW

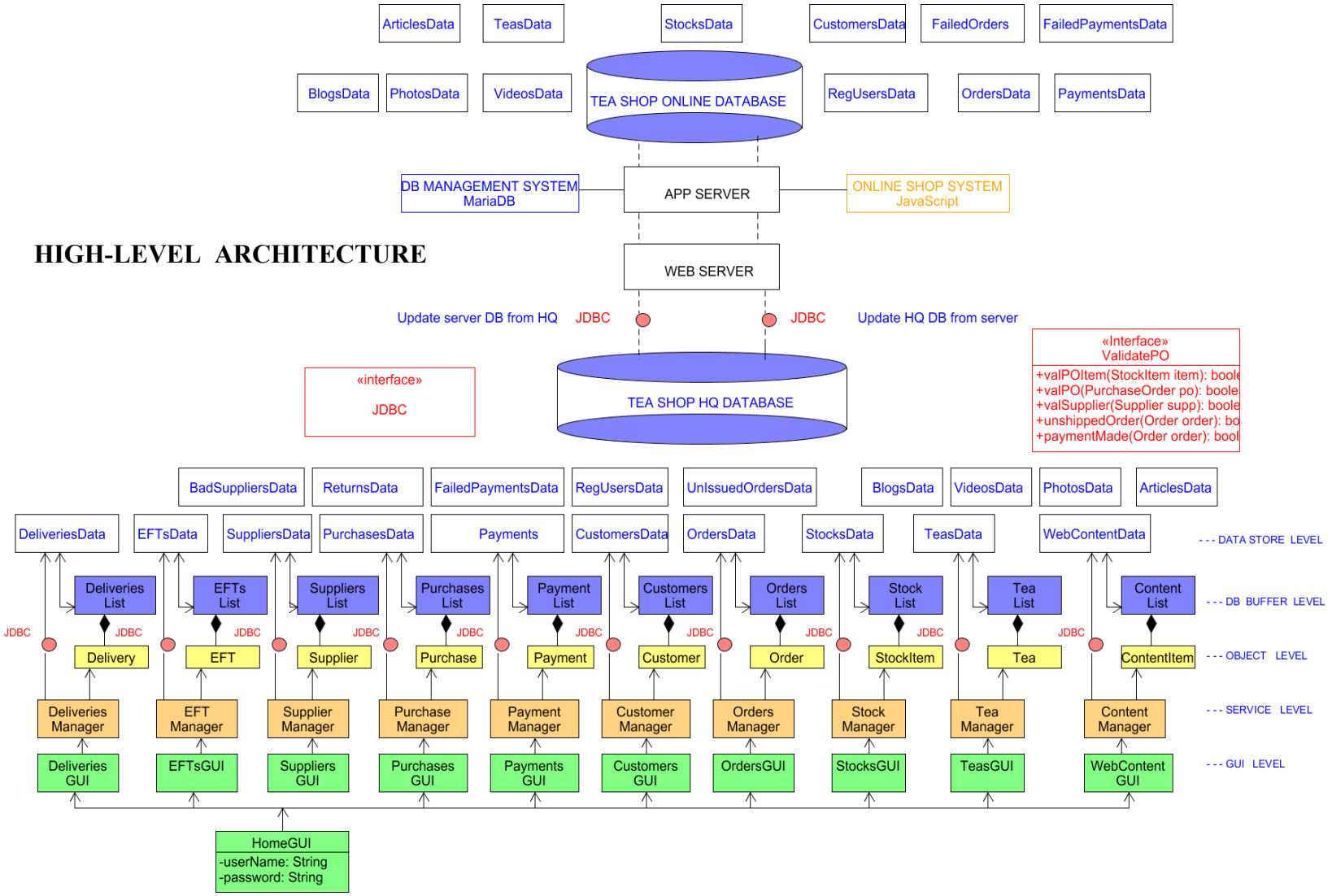
The retail-related parts of the database are mounted on the web server that supports the online shop's website and web application. This minimises both the risk of disconnections and the delay in reading/writing between the online shop's system and the database. A HQ database holds ALL business data including that which is on the server. We feel this arrangement ensures better privacy of the business data. A datalink via JDBC can be readily made to allow browsing of the data on the web server. Automation of updates between common data on both databases can be implemented.

A clear and user-friendly GUI is essential for this type of enterprise as it will most likely be a one person show, if not a spare time job and all daily tasks must be done as quickly as possible. A customised GUI system would enable easy data access and analysis. In our architecture's schema, the 'Manager' classes control all data processing calls coming from the GUIs. Basically the 'Manager' class methods convert each graphical request event, e.g. a click on a "Show Current Stock", into a corresponding SQL query within Java code sent via JDBC to the database's StocksData. On retrieval of the requested data it is then returned to the StocksGUI where a graphics method then displays it in the centre panel of the screen for the user to study. Each GUI has all the essential database queries needed as buttons or items on a drop-down list. Collection classes buffer entry/retrieval between the database and objects used by the GUIs.

A small sub-system at the HQ computer streamlines purchasing, delivery, outgoing payments, etc. Its main attention is devoted to monitoring stock-levels and re-purchasing. A class diagram for this process is sketched out below the System Architecture Diagram. The HQ part of the system is coded in Java.

The major part of the software system is that devoted to managing the online sales. Here anyone may view the site but only registered users and customers (reg. users with one or more order) can buy teas. The buying process is essentially similar to that seen on many online retail sites. It is covered in detail in later sections. Provision is made for cancellation of an order within 30 minutes after the sale. Volume customers buying online may qualify for discounts and trade customers may qualify for credit according to volume and credit rating. Failed – or interrupted – orders are held in the system database and retrieved to that user's order until either processed or removed. New orders, payments, customers and reg users are buffered in collection class objects before being written to the database. As this system must interact with webpage data it is best written in JavaScript. AJAX enables adding/extracting data from the server database, which must also interact via JDBC API to the HQ 's IT system.

HIGH-LEVEL ARCHITECTURE



5. DATABASE

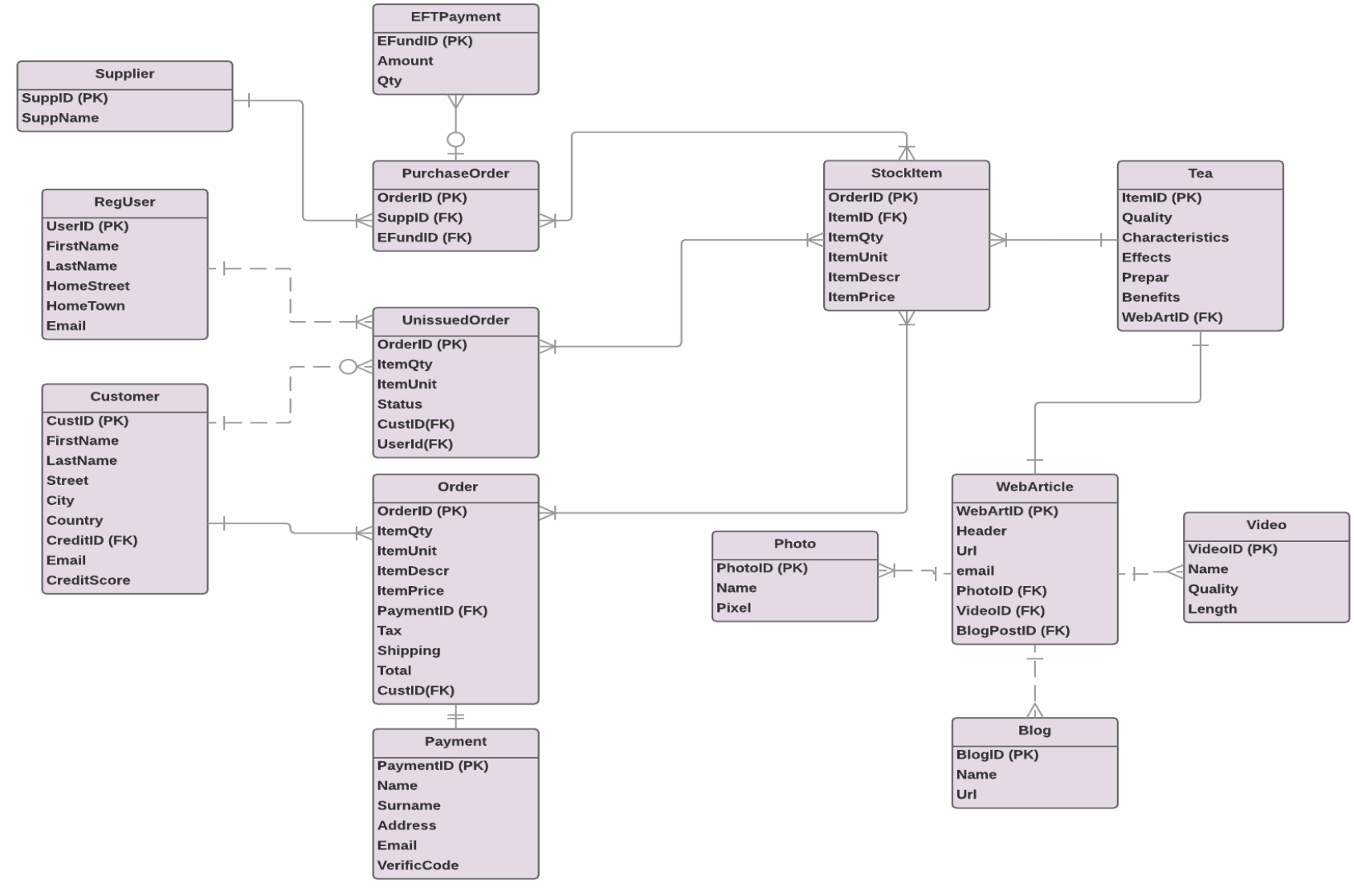
The tea shop system needs to present tea content effectively to users and to capture order data. We recognized that these goals were dependent upon a good design of the data management layer for the new application. We approached the design of the data management layer in two steps: by selecting the object-persistence format and mapping the problem domain classes to the selected format.

Select Object-Persistence Format

The main step is to identify the type of objects that would be in the system and how they would be used. We agreed that the bulk of the data in the system would be the text and numbers that are exchanged with web users regarding customers and orders.The main concern is that relational technology was not optimized to handle complex data, such as the photos and videos that the web content ultimately will require. It can be convenient to store web content using a random file. In this way, the system can be delivered as envisioned while keeping the technology requirements reasonable.The online shop will regularly download order information to the HQ system using a transaction file containing all the required information for that system. Also, it is needed to design the file that stores temporary order information on the web server as customers shop through the website. The file would contain the fields that ultimately would be transferred to an order object.

Map Problem Domain Objects

Based on the decision to use a random file to store the problem domain objects, we created an object-persistence design. It is advised to map all the problem domain classes to the tables in the database. Based on this rule, we include the following table in the database: EFT Payment Table, Customer Table, Reg User Table, Purchase Order Table, Order Table, Unissued Order Table, Tea Table, Stock Item Table, Supplier Table, Web Article Table. We proceeded to create a set of tentative primary keys for each of the tables and the file. While reviewing the current set of attributes for each of the tables, we thought that we had left out the idea of a web article containing a set of photos and video. As such, we added other three tables: Photo Table, Video Table and Blog Table. For the one-to-many relationships, the multi-valued side should be mapped to a column in its table that can store a foreign key back to the single-valued side. This rule is applicable to the associations and at the following tables: Stock Item Table, Photo Table, Video Table and Blog Table, Order Table, Purchase Order Table and Unissued Order Table.



DATABASE SYSTEM DIAGRAM

6. CLASS MODEL / UML FOR ONLINE TEA SHOP

The class model is shown overleaf.

Essentially it implements the actors and actions shown in the Use Case Diagram.

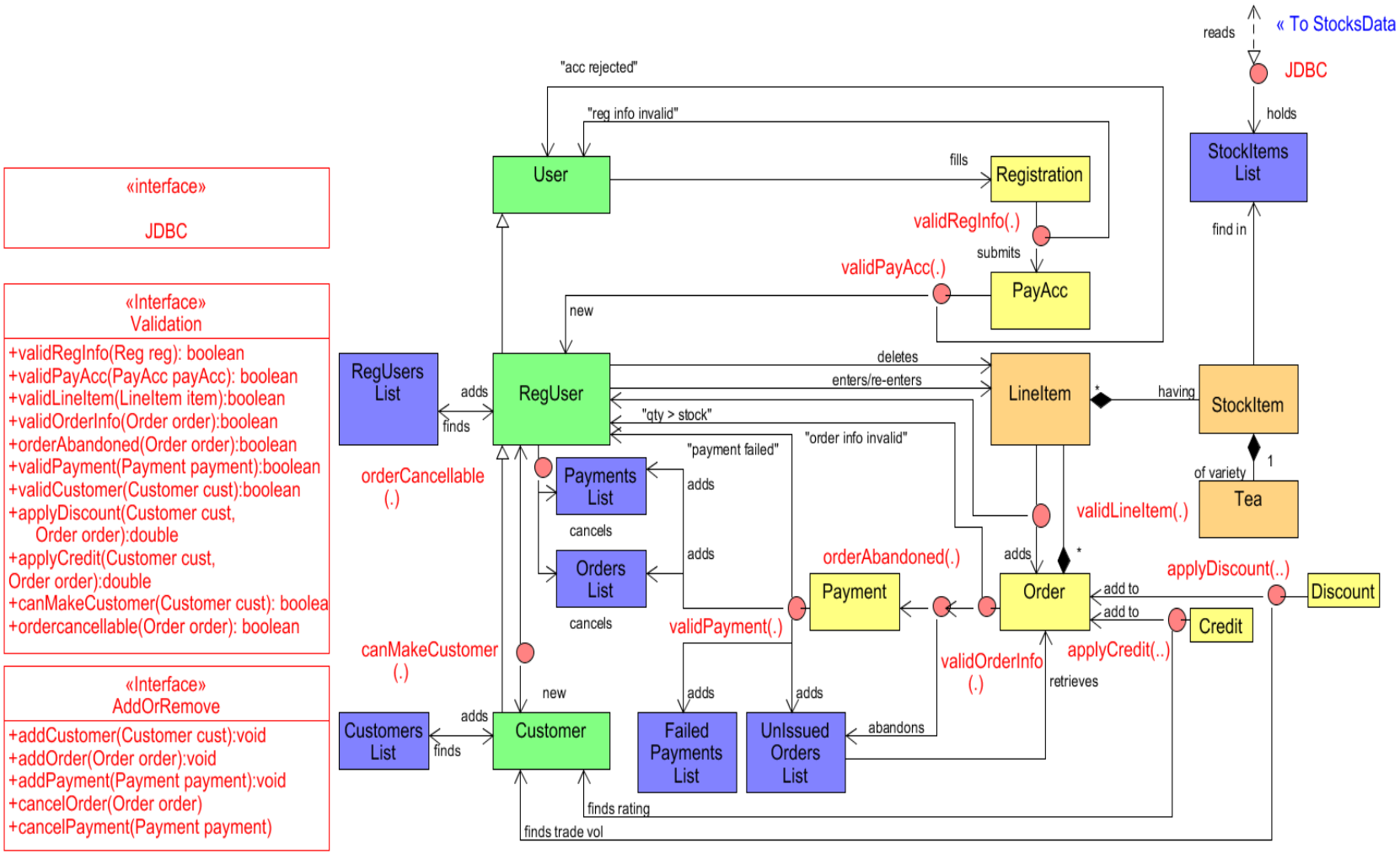
CRC cards for two important classes are shown below.

Order	
Responsibilities	Collaborators
Maintain order details Calculate order additions (e.g. tax, delivery)	Customer / RegUser Validation interface AddOrRemove interface

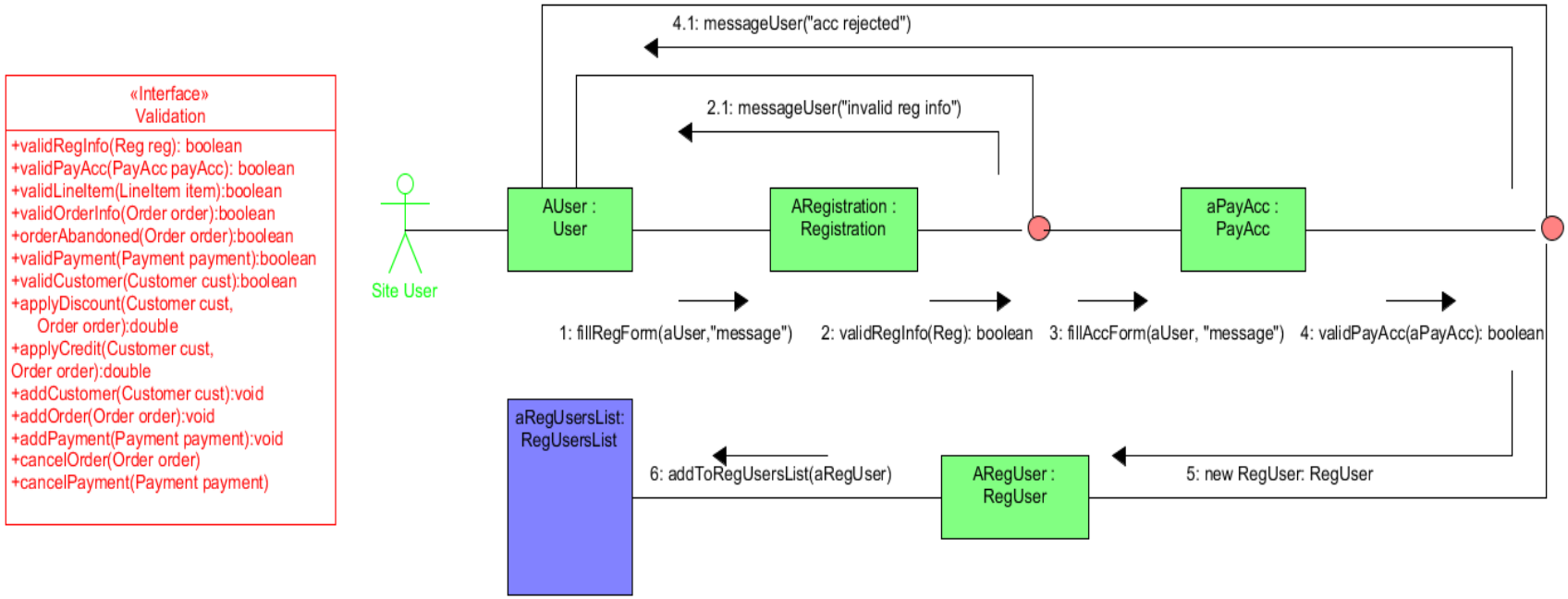
Customer	
Responsibilities	Collaborators
Maintain customer details Meet request to order and cancel Meet requests for customer data	Validation interface AddOrRemove interface

There are 3 kinds of users – Users, Registered Users and Customers. Users are essentially web browsers interested in the product information but not yet interested in buying. Registration involves filling in personal and payment details. On the first delivered order a Registered User becomes a Customer.
High volume customers may qualify for a discount. If they also are trade buyers they may qualify for credit.

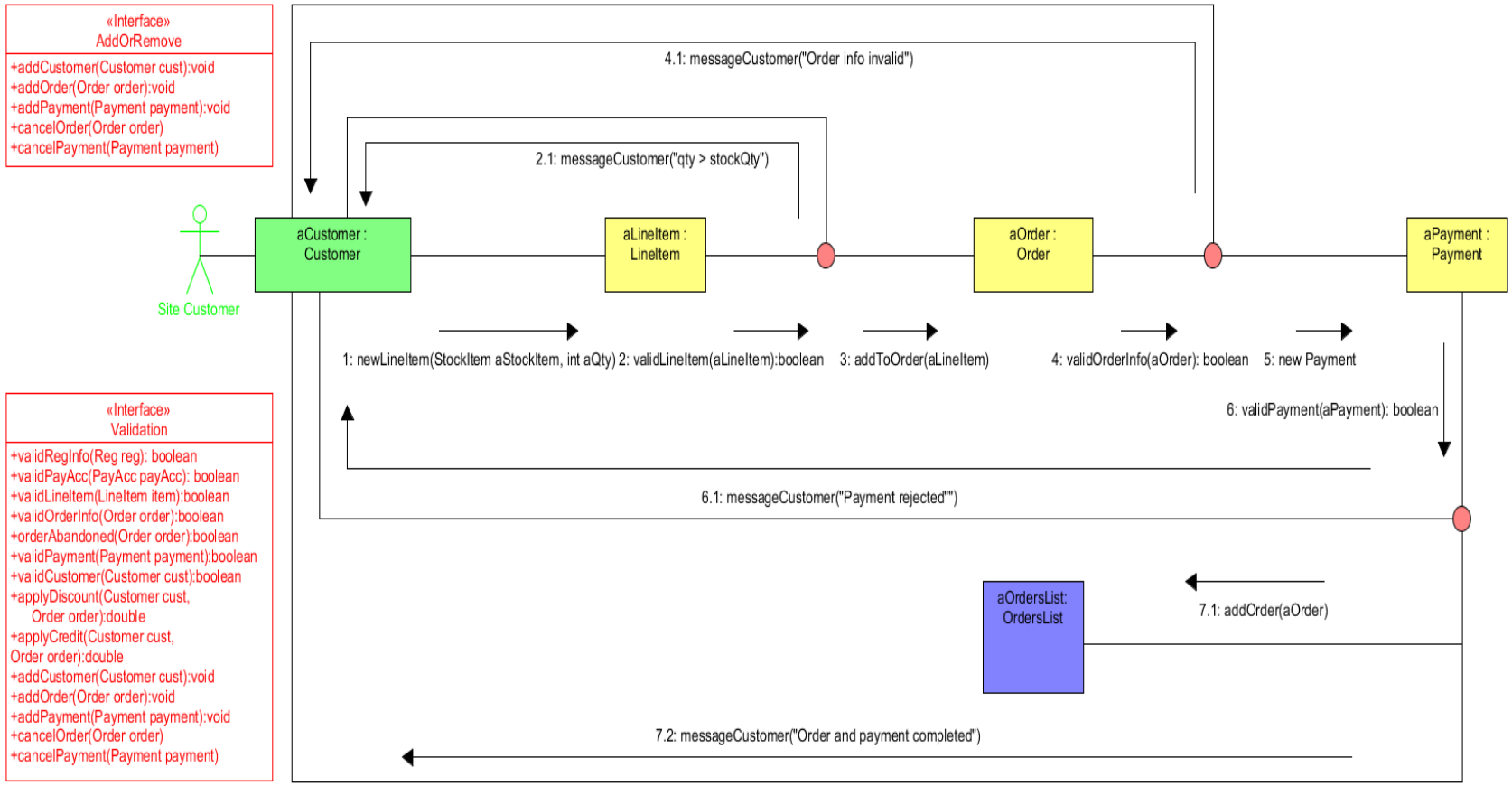
The buying process is essentially the same as on Amazon – a Reg User or Customer just adds a quantity of all desired tea items to an order, then proceeds to the ‘checkout’ where the order and payment are processed. Along the way there are validations for each line item, the whole order and payment acceptance. If payment is accepted, a notification is sent and the order is held pending cancellation. If cancelled or if the order is abandoned before payment, it is buffered to Unissued Orders collection and later added to the corresponding database location. Otherwise the order is actioned for shipping to the buyer and added to the Orders colelction for later insertion in the OrdersData part of the server database.



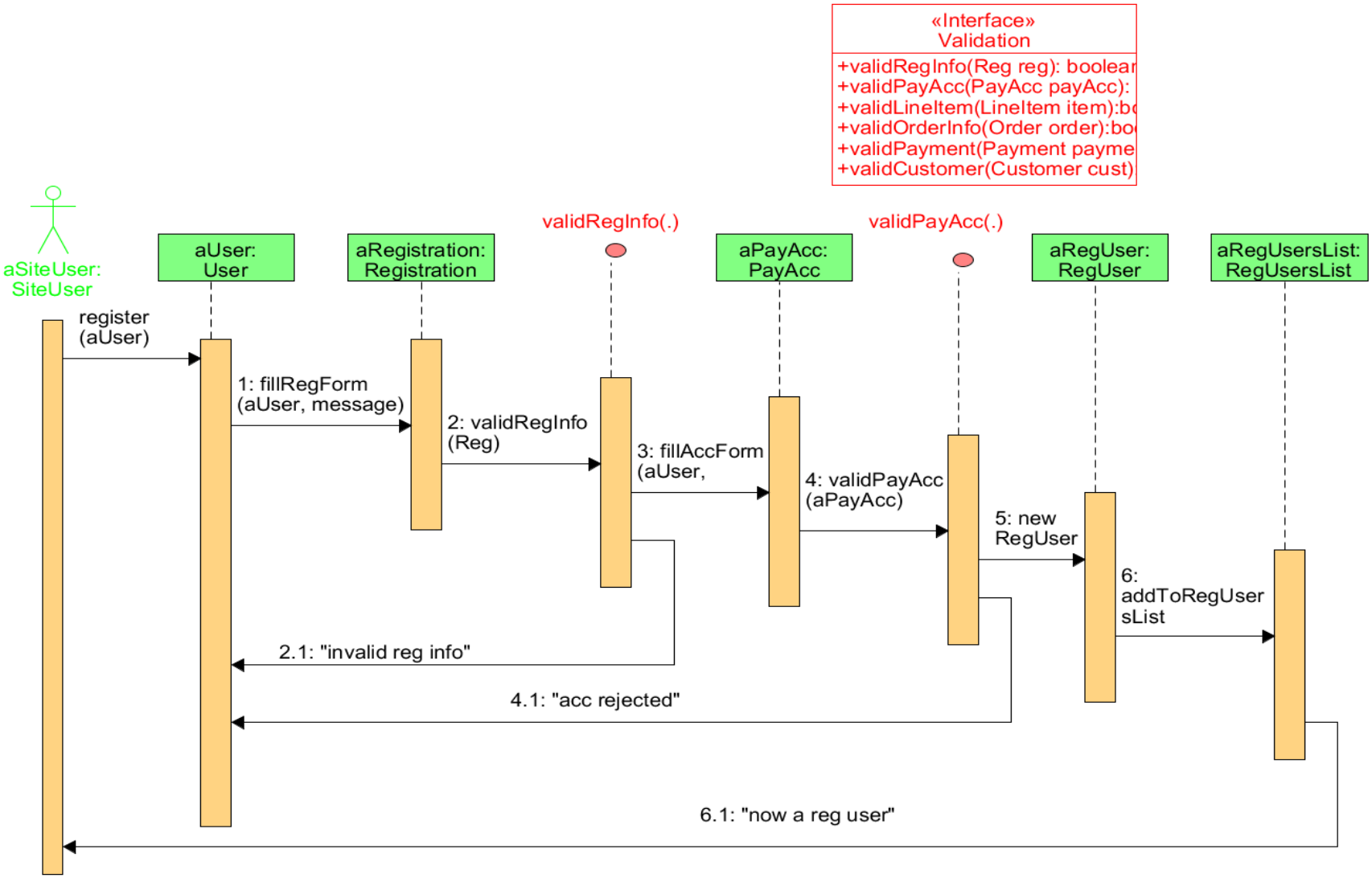
CLASS DIAGRAM FOR ONLINE TEA SHOP



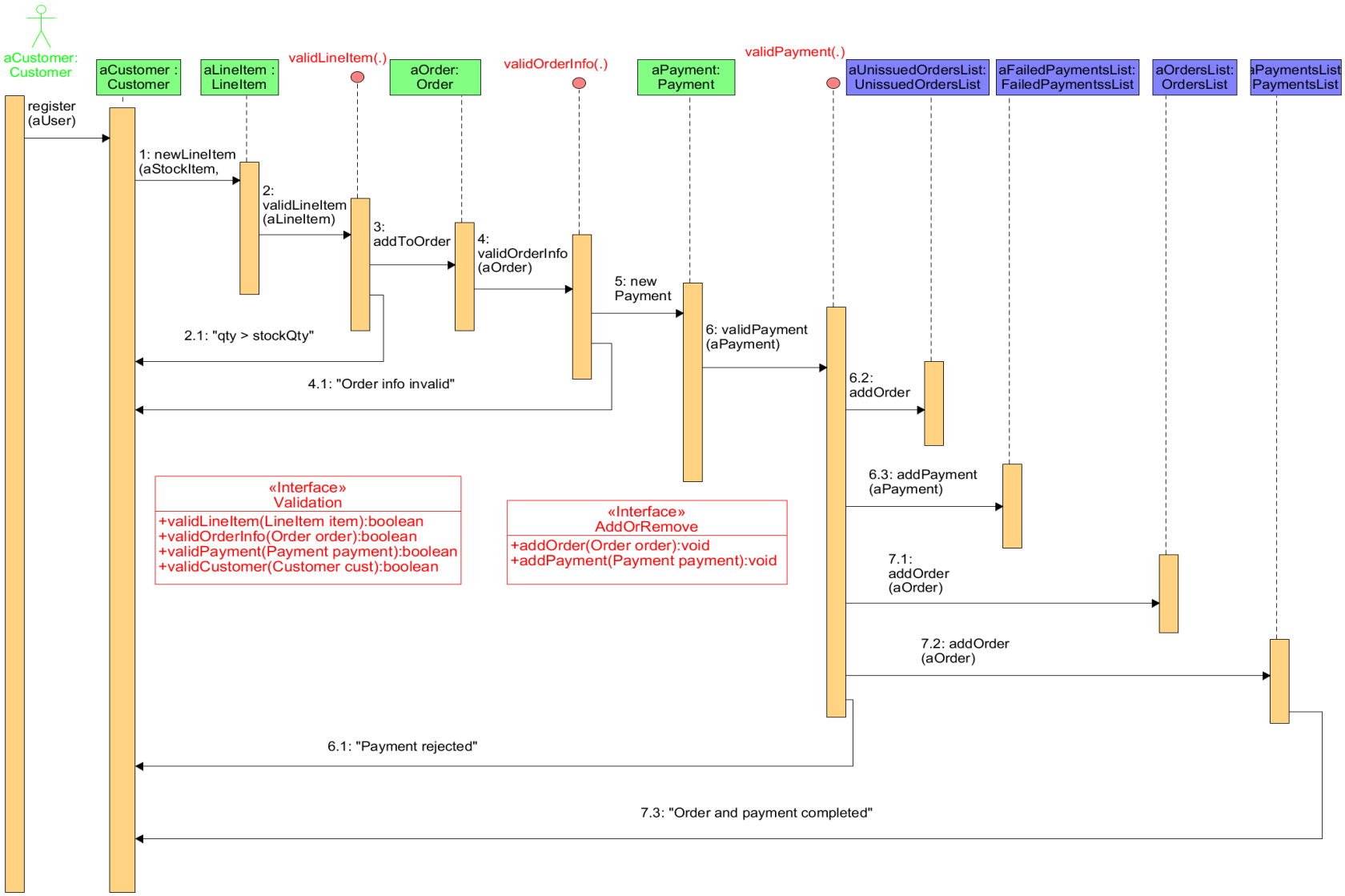
COMMUNICATION DIAGRAM 1 **Use Case :** User Becomes Registered User



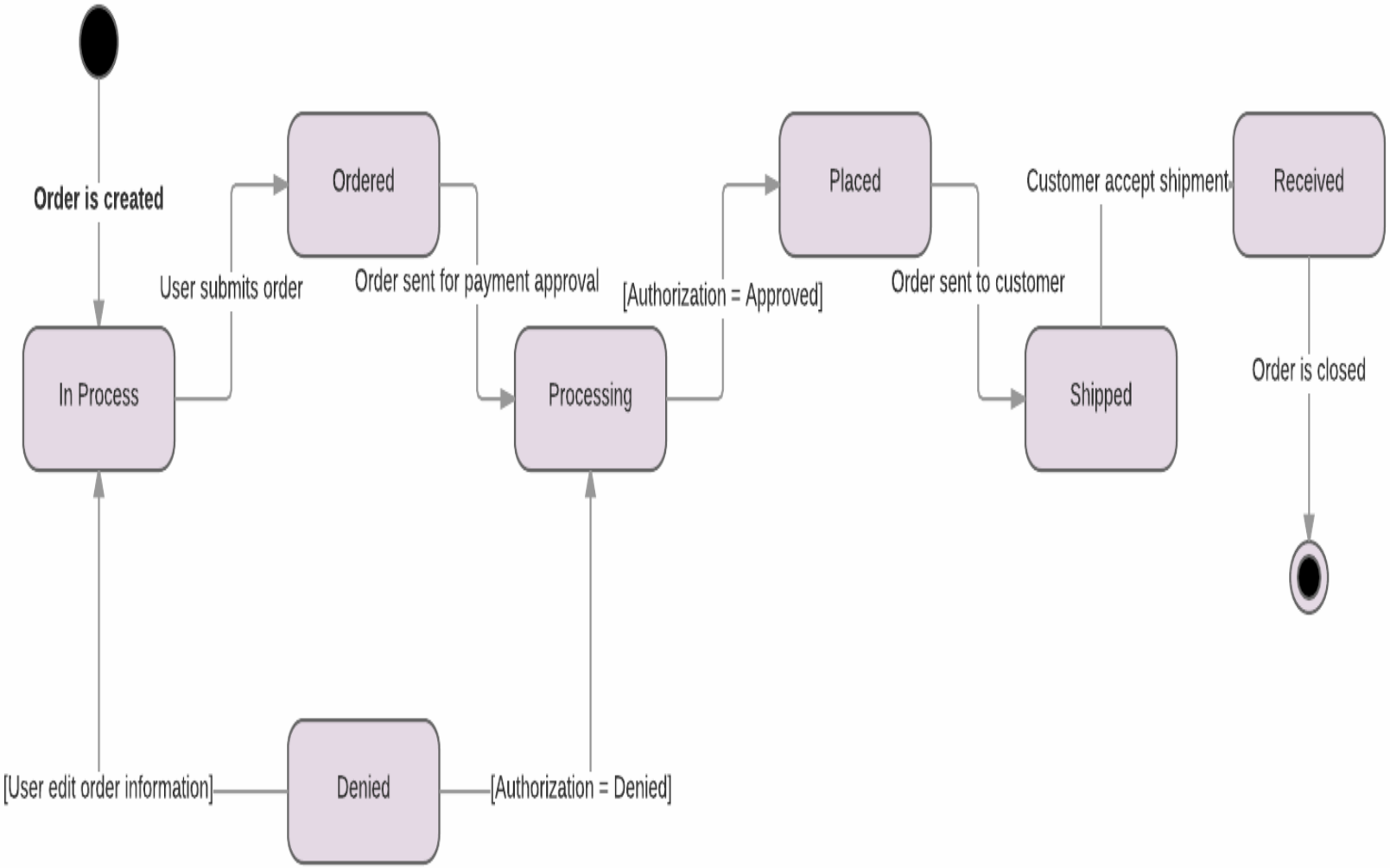
COMMUNICATION DIAGRAM 2 **Use Case :** Customer Submits Order



SEQUENCE DIAGRAM 1 - Use Case : Registering a User



SEQUENCE DIAGRAM 2 : Customer Submits Order



STATE DIAGRAM Use Case : Customer Submits Order

7. CONCLUSIONS

7.1 UML

(1) Firstly and lastly, it must be acknowledged that it is much harder to design an online shop system than to use it. This is true whether or not UML modelling is first done.
(2) It is hard to correctly design the more complex use cases of a class diagram without first doing collaboration diagrams for them. Class diagrams can also be quite a challenge topologically as far as producing a logically sequenced diagram with no line-crossings is concerned – especially with UMLet.
(3) Seeing the 'big picture' view of a project is difficult at the outset of a project.
(4) UML design is really only learned from doing a good number of projects from the outset with it.
(5) Using UML for small or medium sized projects may be prohibitive because of the time spent on the learning curve and the lack of visible gain from it. In the case of this project, it is doubtful if developers would spend time developing class diagram via UML. But for projects that envision many changes in both content, capacity and procedure it is likely that a flexible system architecture might only be achieved through modelling it first with something like UML.
(6) Where UML is adopted for system design, it is vital from both the point of view of effectiveness and productivity that a professional UML tool set is acquired.

7.2 A System Design Project & Its Management

(1) Task and time allocation is very important to achieve project completion within allowed timeframe.
(2) Coordination between members doing different tasks, e.g. different diagrams for the same system, is something that ought to be frequent so as to avoid rework.
(3) The system requirements would be better specified if the system designers were able to interview likely users of the final system. The foregoing project is predicated on users preferring the common online purchase process – but this is a dangerous supposition as they may want something very different.
(4) The system designed is constrained by the limited knowledge and/or experience that the team members have with the technologies involved, i.e. database linking, web app development and software engineering in general. With greater technological proficiency the team could produce a better design.