Appendix A

Case Study of Trading House Automation System

In this appendix, we present the design specifications of an application named “Trading House Automation System (TAS)” [78]. The TAS is a moderate sized application system which deals with the various book keeping activities of a trading house. It maintains a database for its regular customers. When a customer places an order, before processing the order, the credit-worthiness of the customer is first checked by analyzing the history of his payments to different bills sent to him in the past. If the customer is credit-worthy, the items that he has ordered are checked against the list of items that the trading house deals with. The items in the order which the trading house does not deal with, are not processed any further. The items in the customer’s order that the trading house deals are checked for availability in the inventory. If the items are available in the inventory in the desired quantity, a bill with the forwarding address of the customer is printed. The material issue slip is printed and subsequently the inventory data is adjusted to reflect the sale. For out-of-stock items, a pending order file is created for further processing, once the stocks are available. The purchase department periodically generate indents for purchasing out-of stock items. For generating indent the purchase department also examine the pending order file. It also maintain the details of vendors who supply the items that the TAS deals. Further, TAS also answers some managerial queries.

A.1 Requirement Specification of TAS

The objective of the TAS project is to develop a computerized system that would automate various book keeping activities associated with the business of a trading house. The following are the functional requirements of TAS.
A. Case Study of Trading House Automation System

**R1:** The TAS has a set of regular customers. However, a new customer can also register himself into the system. The TAS maintains the names, address etc. of its customers. Each of the customers should be assigned a unique *Customer Identification Number (CIN)* by the TAS. The customers quote their CIN in their transactions.

**R2:** All users of the TAS should pass through an authentication check based on user *login* and *password*.

**R3:** A customer place order with the TAS for various kind of commodities that the TAS deals. A customer also can place an order in advance, delete an order already placed, update an existing order, and keep holding an ongoing order for a certain period of time.

**R4:** Once an order is placed, the accounts department of the trading house first checks the credit-worthiness of the customer. The credit-worthiness of the customer is determined by analyzing the history of its payments to different bills sent to him in the past. If a customer is not credit-worthy, his orders are not processed any further and an appropriate order message regarding rejection of the order is issued to customer with a copy to the manager of the trading house.

**R5:** The items in the customer’s order are checked for availability in the inventory.

- If the items are available in the desired quantity, then
  - a bill (every bill should have its unique bill number and bill date) with the forwarding address of the customer is printed.
  - a material slip (with a unique identity number and a reference of order and bill numbers) is printed.
  - inventory data is adjusted to reflect the sale to the customer.

- If any of the ordered items are not available in the inventory in sufficient quantity to satisfy the order, then these out-of-stock items along with the quantity ordered by the customer and its CIN are stored in a “pending-order” queue for further processing to be carried out when the purchase-department issues the “Generate-indent” command.

**R6:** The accounts department of the trading house would notify against all pending bills periodically. It also would receive payment against a bill. Further, against a bill submitted by a vendor, it would make payment.
A.2. Actors and Use Cases in TAS

R7: Once an order is accepted, the customer can pay online immediately or pay latter by cash or cheque. Account department of the trading house takes care about the late payment.

R8: The storekeeper of the trading house would process an issue slip (which are in the queue) and then update the stock accordingly. It also notifies the customer concerned. The storekeeper, in addition to these, receives items from vendors and then updates the inventory for newly purchased items.

R9: The purchase department should periodically issue commands to generate in- 
dents. When a command to generate indent is issued, the TAS should examine the pending order queue to determine the orders that are pending and determine the total quantity required for each of the items. It then creates a tender and invite quotation from a list of registered vendors. From all the list of submitted quotations, it selects the lowest bid and submit purchase order to the lowest bidder.

R10: A registered vendor against a floated tender can submit a quotation. Further, against a previous supply, a vendor can submit a bill, and receive payment.

R11: The TAS should also support a query handler to answer customer queries, managerial queries, vendor queries and also guest user queries.

A.2  Actors and Use Cases in TAS

All the requirement specifications mentioned in the previous section are depicted in a use case diagram as shown in Fig. A.1. Figure A.1 shows all actors, all use cases, and relationships among different use cases and their sub use cases (that is, include, extend use cases). In Table A.2, Table A.2 and Table A.2 actor-wise various functionalities and the corresponding use cases are presented.
A. Case Study of Trading House Automation System

Figure A.1: Use case Diagram of TAS
## A.2. Actors and Use Cases in TAS

Table A.1: Actors in the TAS and Their Roles

<table>
<thead>
<tr>
<th>Actor</th>
<th>Functionality</th>
<th>Use case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer</td>
<td>1. Registration to the TAS (as first time).</td>
<td>Registration</td>
</tr>
<tr>
<td></td>
<td>2. Authentication checking</td>
<td>Check In</td>
</tr>
<tr>
<td></td>
<td>3. Placing an order</td>
<td>Create Order</td>
</tr>
<tr>
<td></td>
<td>– as an advanced order</td>
<td></td>
</tr>
<tr>
<td></td>
<td>– urgent order</td>
<td></td>
</tr>
<tr>
<td></td>
<td>– regular order</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4. Update (an existing) order</td>
<td>Edit Order</td>
</tr>
<tr>
<td></td>
<td>– delete Order</td>
<td></td>
</tr>
<tr>
<td></td>
<td>– modify order</td>
<td></td>
</tr>
<tr>
<td></td>
<td>– hold order (for a certain period)</td>
<td></td>
</tr>
<tr>
<td>Accounts Manager</td>
<td>1. Check credit-worthiness</td>
<td>Check Customer</td>
</tr>
<tr>
<td></td>
<td>– order is rejected if customer is not a credit-worthy</td>
<td></td>
</tr>
<tr>
<td></td>
<td>– otherwise, order is passed to the department of inventory for further processing</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2. Notify to customers against their pending bills</td>
<td>Notify Customer</td>
</tr>
<tr>
<td></td>
<td>– issue notification to a customer against a bill if the bill is due more than 60 days.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3. Make payment to a vendor against a submitted bill</td>
<td>Make Payment</td>
</tr>
<tr>
<td></td>
<td>– online payment through credit card</td>
<td></td>
</tr>
<tr>
<td></td>
<td>– online payment through banking</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&gt; payment through cash</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&gt; payment through DD or cheque</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4. Receive payment from a customer</td>
<td>Receive Payment</td>
</tr>
<tr>
<td></td>
<td>– online payment through credit card</td>
<td></td>
</tr>
<tr>
<td></td>
<td>– online payment through banking</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&gt; payment through cash</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&gt; payment through DD or cheque</td>
<td></td>
</tr>
</tbody>
</table>

"Continued in the next page"
"Table A.1-continued"

<table>
<thead>
<tr>
<th>Actor</th>
<th>Functionality</th>
<th>Use case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inventory Manager</td>
<td>1. Check items in an order, so that if</td>
<td>Process Order</td>
</tr>
<tr>
<td></td>
<td>– any item is out-of-stock</td>
<td></td>
</tr>
<tr>
<td></td>
<td>– any item is with held</td>
<td></td>
</tr>
<tr>
<td></td>
<td>– any item is under purchase</td>
<td></td>
</tr>
<tr>
<td></td>
<td>– All items are available</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2. Make bill against an order</td>
<td>Make Bill</td>
</tr>
<tr>
<td></td>
<td>3. Issue slip for an order</td>
<td>Issue Slip</td>
</tr>
<tr>
<td>Store Keeper</td>
<td>1. Process the slip from the queue of issue slips</td>
<td>Process Supply</td>
</tr>
<tr>
<td></td>
<td>– update stock</td>
<td></td>
</tr>
<tr>
<td></td>
<td>– notify the supply to customer</td>
<td></td>
</tr>
<tr>
<td></td>
<td>– notify for the supply to the delivery house</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2. Receive purchased item supplied by a vendor</td>
<td>Receive Items</td>
</tr>
<tr>
<td></td>
<td>– update inventory for newly arrived items</td>
<td></td>
</tr>
<tr>
<td></td>
<td>– notify supply to the accounts department</td>
<td></td>
</tr>
<tr>
<td></td>
<td>– issue receipt to the supplier</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3. Process pending orders when items are available</td>
<td>Process Pending</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Order</td>
</tr>
<tr>
<td>Purchase Manager</td>
<td>1. Get items supplied by a vendor</td>
<td>Receive Item</td>
</tr>
<tr>
<td></td>
<td>2. Create and notify tender</td>
<td>Create Tender</td>
</tr>
<tr>
<td></td>
<td>3. Generate indent against all items on demand</td>
<td>Generate Indent</td>
</tr>
<tr>
<td>Vendor</td>
<td>1. Submit quotation against a tender</td>
<td>Submit Quotation</td>
</tr>
<tr>
<td></td>
<td>2. Submit bill against a supply</td>
<td>Submit Bill</td>
</tr>
</tbody>
</table>
A.2. Actors and Use Cases in TAS

"Table A.1-continued"

<table>
<thead>
<tr>
<th>Actor</th>
<th>Functionality</th>
<th>Use case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer</td>
<td>1. Retrieving information regarding an item</td>
<td>Customer Query</td>
</tr>
<tr>
<td></td>
<td>2. Status of an order</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3. Forget CIN and getting it</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4. FAQs</td>
<td></td>
</tr>
<tr>
<td>Manager</td>
<td>1. Statistics of different items sold over any given period of time</td>
<td>Manager Query</td>
</tr>
<tr>
<td></td>
<td>2. Customer profile</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3. Customer purchase history</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4. Status of the inventory</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5. Statements of accounts(day, week, month, year)</td>
<td></td>
</tr>
<tr>
<td>Vendor</td>
<td>1. Query about the specification of an item</td>
<td>Vendor Query</td>
</tr>
<tr>
<td></td>
<td>2. Status of the quotation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3. Status of the bill</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4. FAQs</td>
<td></td>
</tr>
<tr>
<td>Guest</td>
<td>1. General information (queries posed by a borrower)</td>
<td>General Query</td>
</tr>
<tr>
<td></td>
<td>2. FAQs</td>
<td></td>
</tr>
<tr>
<td>Administrator</td>
<td>1. Provide access privilege to different users</td>
<td>Set User</td>
</tr>
<tr>
<td></td>
<td>– Vendor, Accounts Manager, Inventory Manager</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1. Prohibit access rights to a customer or other users</td>
<td>Withheld User</td>
</tr>
<tr>
<td></td>
<td>1. Some system setting on different operations</td>
<td>System Setting</td>
</tr>
</tbody>
</table>
A.3 Use Case Descriptions

In this section, we present the details use case descriptions of the use cases mentioned in Table A.2, Table A.2 and Table A.2. The use case descriptions are presented following the UML 2.0 syntax of use case template [89].

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Registration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>This use case allows a user to register to the TAS</td>
</tr>
<tr>
<td>Actors</td>
<td>Manager, Customer, Vendor</td>
</tr>
</tbody>
</table>
| Pre condition | 1. User fills a form with certain desired information to be used for the registration  
                  2. A database of all existing users |
| Main scenario | 1. User submit the form  
                  2. Search the database for the user  
                  3. Message to the user |
| Alternative scenarios | 2.1 User information does exist  
                         – user cannot register |
| Post condition | Update the database with the new entry and a CIN is assigned and notified to the user |
| Dependency    | The use case Registration includes use case “Search” to search a database of all existing registered users |
A.3. Use Case Descriptions

Table A.3: Check In Use Case

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Check In</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>Authentication checking based on login and password</td>
</tr>
<tr>
<td>Actors</td>
<td>Manager, Customer, Vendor</td>
</tr>
<tr>
<td>Pre condition</td>
<td>A database of all existing users.</td>
</tr>
</tbody>
</table>
| Main scenario | 1. User submit UserId and Password  
2. Search the database of all registered users  
3. Message to the user |
| Alternative scenarios | 2.1 UserId and password do not match |
| Post condition | If authentication is failed, then display welcome screen, else display the screen for next transaction depending on the user type and log the time of check-in. |
| Dependency    | This use case includes the Search use case |

Table A.4: Check Customer Use Case

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Check Customer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>To check credit-worthiness for a customer</td>
</tr>
<tr>
<td>Actors</td>
<td>Accounts Manager</td>
</tr>
<tr>
<td>Pre condition</td>
<td>Payment history record of all customers is available.</td>
</tr>
<tr>
<td>Main scenario</td>
<td>1. Get Customer Identification Number (CIN)</td>
</tr>
</tbody>
</table>
| Alternative scenario | 2.1. Customer is not credit-worthy  
– reject the order  
– move the order from waiting queue  
2.2. Customer is credit-worthy  
– move order from waiting queue to processing queue |
| Post condition | If order number is not NULL, remove it from the queue |
| Dependency    | None |
### Table A.5: Create Order Use Case

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Create Order</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Summary</strong></td>
<td>This is a use case through which a customer can place an order of a list of items that TAS deals with.</td>
</tr>
<tr>
<td><strong>Actors</strong></td>
<td>Customer</td>
</tr>
</tbody>
</table>
| **Pre condition** | 1. Customer is checked in successfully  
                | 2. System displays the menu screen of the customer |
| **Main scenario** | 1. Customer selects the option *create order* from items and specify the quantity of each item selected  
                | 3. Check the selection (if all entries are filled in)  
                | 4. Submit the order  
                | 5. Notification message with order number to the customer |
| **Alternative scenarios 1** | 3.1 Selection is incomplete  
                | – prompt the incorrect entry(ies)  
                | – goto 2 |
| **Alternative scenarios 2** | 4.1. User quit the transaction create order  
                | – return to the menu screen of the customer |
| **Post condition** | 1. Order is created with an order number  
                | 2. Order is placed into a queue of orders |
| **Dependency** | None         |
### Table A.6: Process Order Use Case

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Process Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>This use case checks an order in a queue of orders, generate indents for items not in inventory, prepare bill when all items are available and issue a delivery order slip</td>
</tr>
<tr>
<td>Actors</td>
<td>Manager (Inventory Manager)</td>
</tr>
<tr>
<td>Pre condition</td>
<td>None</td>
</tr>
</tbody>
</table>
| Main scenario | 1. Check the status of the customer  
2. Check the items in the order  
3. For all items in the order, check stock of items and process the order |
| Alternative scenarios 1 | 1.1 Customer is not credit-worthy  
– reject order  
– send regret message to the customer |
| Alternative scenarios 2 | 2.1 Not a single items is valid in the order  
– reject the order  
– send regret message to the customer |
| Alternative scenarios 3 | 2.2 Some items are valid items  
2.2.1 get the option from the customer |
| Alternative scenarios 4 | 2.2.1.1 Quit the process order task  
– reject the order |
| Alternative scenarios 5 | 2.2.1.2 Modify the order list  
– get modification in the order |
| Alternative scenarios 6 | 3.1 Items are currently out of stock  
– place the order in the waiting list  
– message to the customer |
| Alternative scenarios 7 | 3.2 Items are currently suspended  
– reject the order  
– regret message to the customer |

"Continued in the next page"
A. Case Study of Trading House Automation System

"Table A.6-continued"

| Alternative scenario 8 | 3.3 Items are available  
|                        | 3.3.1 check exemplary status of the customer  
|                        | 3.3.2 get payment for the order  
|                        | 3.3.3 update customer history of payment  
|                        | 3.3.4 update stock of inventory  
|                        | 3.3.5 successful order processing to the customer |

| Alternative scenario 9 | 3.3.1.1 Customer is of “exemplary” status 
|                        | – allow bonus |

| Post condition         | None |
| Dependency             | Generate Indent, Issue Slip and Generate Bill use cases |

Table A.7: Delete Order Use Case

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Delete Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>Deleting an order from the list of all orders under process</td>
</tr>
<tr>
<td>Actors</td>
<td>Customer</td>
</tr>
<tr>
<td>Pre condition</td>
<td>Order number is validated</td>
</tr>
</tbody>
</table>
| Main scenario     | 1. Get confirmation  
|                   | 2. Charge for order deletion  
|                   | 3. Display message for the status of operation |
| Alternative scenarios | 1.1. Order is processed and in the process of delivery  
|                     | – deletion operation is failed  
|                     | – exit to the main screen of the customer  
|                     | 1.2. Reject (Quit) the operation  
|                     | – exit to the main menu screen of the customer |
| Post condition     | 1. Deleted order is removed from the queue  
|                    | 2. Update record of orders       |
| Dependency         | Include the use case for calculating order deletion charge |
A.3. Use Case Descriptions

Table A.8: Modify Order Use Case

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Modify Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>Facilitates adding, deleting item(s) and updating quantity.</td>
</tr>
<tr>
<td>Actors</td>
<td>Customer</td>
</tr>
<tr>
<td>Pre condition</td>
<td>1. Order number is validated</td>
</tr>
</tbody>
</table>
| Main scenario | 1. Check the status of the order  
2. Get the modified order form fill in  
3. Check the validity of modification sought  
4. A message to confirm the modification to the customer  
5. Place the order in the queue of orders under process |
| Alternative scenarios 1 | 1.1. Order is processed for delivery  
– exit the modification operation |
| Alternative scenarios 2 | 2.1. Invalid modification entries  
– exit the modification operation |
| Post condition | None                           |
| Dependency    | None                           |
### Table A.9: Edit Order Use Case

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Edit Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>This use case facilitates a user to delete an order already placed, update an order or withheld an order for a certain period of time.</td>
</tr>
<tr>
<td>Actors</td>
<td>Customer</td>
</tr>
<tr>
<td>Pre condition</td>
<td>Same as the Create Order use case.</td>
</tr>
</tbody>
</table>
| Main scenario  | 1. Select order number  
2. Search the list of all orders for the given order number  
3. System display the screen for the order number  
4. Select option  
5. Display message regarding the status of the operation. |
| Alternative scenarios 1 | 2.1. Order number does not exist  
– display the “Edit Order” screen for customer |
| Alternative scenarios 2 | 4.1. The option is “delete”  
– call the “Delete Order” use case  
4.2. The option is “modify”  
– call the “Modify Order” use case  
4.2. The option is to “hold” an order  
– call the “Withheld Order” use case  
4.2. Quit |
| Post condition | None                                                                       |
| Dependency     | None                                                                       |
### Table A.10: Pay Bill Use Case

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Pay Bill</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>Allow a customer to submit payment against a pending bill corresponding to an order. The payment made is through ATM card or Internet banking.</td>
</tr>
<tr>
<td>Actors</td>
<td>Customer</td>
</tr>
</tbody>
</table>
| Pre condition       | 1. Customer is an authorized user to execute the operation  
                       2. ATM card service provider and Internet banking service are accessible from the TAS. |
| Main scenario       | 1. Check the order status  
                       2. Select mode of transaction  
                       3. Update order  
                       4. Status message to the user |
| Alternative scenarios 1 | 1.1. Order does not exist i.e. invalid order number  
                       – failed message to the customer  
                       – go to “Pay Bill” screen with “Retry” suggestion or Quit the system |
| Alternative scenarios 2 | 1.2. Payment against the order no. has already been made.  
                       – appropriate message to the user  
                       – goto the main screen of the system or quit. |
| Alternative scenarios 3 | 1.3. Order exist with payment due  
                       – appropriate message to the user |
| Alternative scenarios 4 | 2.1. Mode of transaction is through ATM Card  
                       2.1.1 select the ATM card service from a list of ATM Card Provider  
                       2.1.2 call the selected ATM card service provider |
| Alternative scenarios 5 | 2.2. Mode of transaction is through Internet banking  
                       2.2.1 select the Internet baning service provider from a list of services  
                       2.2.2. call the selected Internet banking service provider |
| Alternative scenarios 6 | 2.1.2.1. ATM Card service provider fail to authenticate ATM card holder  
                       – appropriate message to the customer  
                       – goto “Pay Bill” screen with try option or quit the operation. |

"Continued in the next page"
A. Case Study of Trading House Automation System

"Table A.10-continued"

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Pay Bill contd...</th>
</tr>
</thead>
</table>
| Alternative scenarios 7 | 2.1.2.2. Full payment is not made  
– appropriate message to the customer  
– update payment history of the customer  
– update order with current payment and record transaction number, date etc. |
| Alternative scenarios 8 | 2.2.2.1. Internet bank service provider fails to authenticate customer’s account in the Internet bank service  
– appropriate message to the customer  
– got the “Pay Bill” screen with retry option or quit the operation |
| Alternative scenarios 9 | 2.2.2.2. Full payment is not made  
– appropriate message to the customer  
– update the payment history of the customer  
– update order with current payment and transaction number, date, amount etc. |
| Alternative scenarios 10 | 2.1.2.3. Full payment is made (ATM)  
– appropriate message to the customer  
– update the payment history of the customer  
– update the customer status as credit worthy  
– update order with current payment information i.e. transaction, number, date, amount etc. |
| Alternative scenarios 11 | 2.2.2.3. Full payment is made (Internet)  
– appropriate message to the customer  
– update the payment history of the customer  
– update the customer status as credit worthy  
– update order with current payment information i.e. transaction, number, date, amount etc. |
| Post condition      | 1. Customer status updation upon successful full payment  
2. Payment made recorded against the order no. |
| Dependency          | Search, ATM credit card service, Internet bank service |
## A.3. Use Case Descriptions

### Table A.11: Generate Indent Use Case

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Generate Indent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>It will generate an indent for all items in an order, which are not or partly available in stock</td>
</tr>
<tr>
<td>Actors</td>
<td>Inventory Manager</td>
</tr>
<tr>
<td>Pre condition</td>
<td>Inventory database is available</td>
</tr>
</tbody>
</table>
| Main scenario | 1. Create an indent with a unique ID and Order number as reference  
2. For all out-of-stock items get items numbers, names, makes and quantities intended  
3. Prepare indent form |
| Alternative scenario | None |
| Post condition | An indent is created |
| Dependency    | None |

### Table A.12: Issue Slip Use Case

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Issue Slip</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>This use case issues a slip for all available item in an order.</td>
</tr>
<tr>
<td>Actors</td>
<td>Inventory Manager</td>
</tr>
<tr>
<td>Pre condition</td>
<td>Inventory database is available</td>
</tr>
</tbody>
</table>
| Main scenario | 1. Create an issue slip with a unique ID and a reference number  
2. For all not available items in the order, get items numbers, name, makes and quantities intended  
3. Prepare delivery slip |
<p>| Alternative scenario | None |
| Post condition | An Issue Slip is prepared |
| Dependency    | None |</p>
<table>
<thead>
<tr>
<th>Use case name</th>
<th>Notify Pending Bills</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>Notify a customer about all it pending bills</td>
</tr>
<tr>
<td>Actors</td>
<td>Accounts Manager</td>
</tr>
<tr>
<td>Pre condition</td>
<td></td>
</tr>
<tr>
<td>Main scenario</td>
<td>1. Get customer identity number</td>
</tr>
<tr>
<td></td>
<td>2. Search the list of pending bills for all pending bills of the customer</td>
</tr>
<tr>
<td></td>
<td>3. For all pending bills notify to customer or print notification</td>
</tr>
<tr>
<td>Alternative scenario 1</td>
<td>1.1. CIN does not exist</td>
</tr>
<tr>
<td></td>
<td>–return to the main menu</td>
</tr>
<tr>
<td>Alternative scenario 2</td>
<td>2.1. No match found</td>
</tr>
<tr>
<td></td>
<td>–message for non-existence of pending bill.</td>
</tr>
<tr>
<td></td>
<td>–return to the main menu</td>
</tr>
<tr>
<td>Alternative scenario 3</td>
<td>3.1. Bill date does not exceeds 60 days.</td>
</tr>
<tr>
<td></td>
<td>–ignore the bill(s)</td>
</tr>
<tr>
<td>Post condition</td>
<td>None</td>
</tr>
<tr>
<td>Dependency</td>
<td>None</td>
</tr>
</tbody>
</table>
### Table A.14: Make Payment Use Case

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Make Payment (to vendor’s bill)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Summary</strong></td>
<td>Making payment against a bill submitted by a vendor</td>
</tr>
<tr>
<td><strong>Actors</strong></td>
<td>Accounts Manager</td>
</tr>
<tr>
<td><strong>Pre condition</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Main scenario</strong></td>
<td>1. Get bill number  2. Verify the existence of the bill  3. Confirm payment  4. Update account database  5. Print payment order and cheque</td>
</tr>
<tr>
<td><strong>Alternative scenario 1</strong></td>
<td>2.1. Bill number does not exist  – return to the main menu</td>
</tr>
<tr>
<td><strong>Alternative scenario 2</strong></td>
<td>2.2. Payment already made against the bill number.  – return to the main menu</td>
</tr>
<tr>
<td><strong>Alternative scenario 3</strong></td>
<td>4.1. Out of fund  – return to the main menu</td>
</tr>
<tr>
<td><strong>Post condition</strong></td>
<td>Balance is updated by amount paid if the payment is successful</td>
</tr>
<tr>
<td><strong>Dependency</strong></td>
<td>None</td>
</tr>
</tbody>
</table>
### Table A.15: Process Pending Order Use Case

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Process Pending Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>Process pending orders, which are in the queue and items are available</td>
</tr>
<tr>
<td>Actors</td>
<td>Store Keeper</td>
</tr>
<tr>
<td>Pre condition</td>
<td>None</td>
</tr>
</tbody>
</table>
| Main scenario    | 1. Get an order from the queue of pending orders.  
|                  | 2. Check validity of the pending order.  
|                  | 3. For each item in the order update inventory  
|                  | 4. Notify supply to customer  
|                  | 5. Notify for supply to the delivery house                                             |
| Alternative      | 1.1. Pending order queue is empty  
| scenario 1       | – return to the main screen                                                           |
| Alternative      | 2.1. Order is cancelled  
| scenario 2       | – return to the main screen                                                           |
| Alternative      | 2.2. Order is withheld  
| scenario 3       | – return to the main screen                                                           |
| Post condition   | Inventory is updated if order is supplied                                               |
| Dependency       | None                                                                                  |
A.3. Use Case Descriptions

Table A.16: Process Supply Use Case

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Process Supply</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Summary</strong></td>
<td>For an issue slip in the queue of pending supplies, it processes for supply</td>
</tr>
<tr>
<td><strong>Actors</strong></td>
<td>Store Keeper</td>
</tr>
<tr>
<td><strong>Pre condition</strong></td>
<td>None</td>
</tr>
</tbody>
</table>
| **Main scenario** | 1. Get an issue slip from the queue of issue slips  
                        2. For each item in the issue slip update the stock inventory  
                        3. Notify the supply to customer  
                        4. Notify for the supply to the delivery house |
| **Alternative scenario 1** | 1.1. Queue is empty  
                                      – return to the main screen |
| **Alternative scenario 2** | 2.1. Some items are out-of stock or not sufficient in stock  
                                      – create a due slip for item(s) unable to supply  
                                      – put the due slip in the queue of pending order  
                                      – create demand for items with insufficient stock |
| **Post condition** | Stocks are updated for items supplied and demands are created for items not supplied |
| **Dependency** | None            |
### Table A.17: Receive Payment Use Case

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Receive Payment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>Receive payment against a bill from a customer</td>
</tr>
<tr>
<td>Actors</td>
<td>Accounts Manager</td>
</tr>
<tr>
<td>Pre condition</td>
<td>None</td>
</tr>
</tbody>
</table>
| Main scenario | 1. Get bill number  
2. Verify bill  
3. Get mode of payment  
4. Confirm receiving payment  
5. Update accounts database  
6. Print payment receipt |
| Alternative scenario 1 | 2.1. Bill number does not exist  
– return to the main screen |
| Alternative scenario 2 | 2.2. Payment already received  
– return to the main menu |
| Alternative scenario 3 | 3.1. Mode of payment is cash  
– update cash balance  
– update accounts database |
| Alternative scenario 4 | 4.1. Mode of payment is cheque  
4.1.1. enter cheque details  
4.1.2. issue request for cheque reconciliation to drawee bank  
4.1.3. update account database |
| Alternative scenario 5 | 5.1. Mode of payment is credit card  
5.1.1. get credit card details  
5.1.2. send card details for money transfer to card issuer  
5.1.3. update account database |
| Alternative scenario 6 | 4.1.2.1. Bank fails transaction  
– return to the main menu |
| Alternative scenario 7 | 5.1.2.1. Fail to authenticate card or balance is insufficient  
– return to the main menu |
| Post condition | account database is updated if receive payment is success |
| Dependency    | Cheque authentication, card authentication |
### A.3. Use Case Descriptions

#### Table A.18: Query Handler Use Case

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Query Handler</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>Answer to users’ query</td>
</tr>
<tr>
<td>Actors</td>
<td>Manager, Customer, Vendor</td>
</tr>
<tr>
<td>Pre condition</td>
<td>None</td>
</tr>
<tr>
<td>Main scenario</td>
<td>1. Get query&lt;br&gt;2. Check eligibility of the user&lt;br&gt;3. Return response to query</td>
</tr>
<tr>
<td>Alternative scenario 1</td>
<td>2.1. User is not eligible for the query&lt;br&gt;– return to the main screen</td>
</tr>
<tr>
<td>Post condition</td>
<td>None</td>
</tr>
<tr>
<td>Dependency</td>
<td>Q_Manager, Q_Customer, Q_Vendor</td>
</tr>
</tbody>
</table>

#### Table A.19: Edit User Use Case

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Edit User</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>To edit the profile of a user</td>
</tr>
<tr>
<td>Actors</td>
<td>Manager, Customer, Vendor</td>
</tr>
<tr>
<td>Pre condition</td>
<td>None</td>
</tr>
<tr>
<td>Main scenario</td>
<td>1. Check authorization&lt;br&gt;2. Update profile&lt;br&gt;3. Notify the change</td>
</tr>
<tr>
<td>Alternative scenario 1</td>
<td>1.1. Authorization fails&lt;br&gt;– return to the main screen</td>
</tr>
<tr>
<td>Post condition</td>
<td>Updated user’s profile in the database</td>
</tr>
<tr>
<td>Dependency</td>
<td>None</td>
</tr>
</tbody>
</table>
Table A.20: Receive Item Use Case

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Receive Item</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>Receiving items supplied by a vendor</td>
</tr>
<tr>
<td>Actors</td>
<td>Store Keeper</td>
</tr>
<tr>
<td>Pre condition</td>
<td>None</td>
</tr>
</tbody>
</table>
| Main scenario | 1. Update the inventory for the newly arrived items  
                   2. Issue receipt to the supplier  
                   3. Notify supply to the accounts department |
| Alternative scenario 1 | None |
| Post condition | None         |
| Dependency    | None         |

Table A.21: Withheld User Use Case

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Withheld User</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>Withheld a user if more than three transactions are not paid or cancelled or withheld</td>
</tr>
<tr>
<td>Actors</td>
<td>Manager</td>
</tr>
<tr>
<td>Pre condition</td>
<td>None</td>
</tr>
</tbody>
</table>
| Main scenario | 1. Get customer id.  
                   2. Get previous transaction.  
                   3. Check last three transaction  
                   4. Notify user |
| Alternative scenario 1 | 3.1. Last three transactions are yet to be paid  
                                                – withheld the customer by setting a tag |
| Post condition | None          |
| Dependency    | None          |
A.4 Sequence Diagrams of Use Cases in TAS

In this section, sequence diagrams (also called interaction diagrams) of all the use cases described in the previous section is presented. The sequence diagrams are drawn according to the syntax of UML 2.0 [89].

Figure A.2: Sequence Diagram of Registration Use Case

Figure A.3: Sequence Diagram of Check In Use Case
A. Case Study of Trading House Automation System

Figure A.4: Sequence Diagram of *Edit Order* Use Case
A.4. Sequence Diagrams of Use Cases in TAS

Figure A.5: Sequence Diagram of Delete Order Use Case

Figure A.6: Sequence Diagram of Modify Order Use Case
A. Case Study of Trading House Automation System

Figure A.7: Sequence Diagram of Create Order Use Case

Figure A.8: Sequence Diagram of Notify Pending Bills Use Case
A.4. Sequence Diagrams of Use Cases in TAS

Figure A.9: Sequence Diagram of Process Order Use Case
Figure A.10: Sequence Diagram of Pay Bill Use Case
A.4. Sequence Diagrams of Use Cases in TAS

Figure A.11: Sequence Diagram of Issue Slip Use Case

Figure A.12: Sequence Diagram of Generate Indent Use Case
A. Case Study of Trading House Automation System

begin
getBillNumber(x)
v = verifyBill()
if (v == NULL) then break
else displayMainScreen() // Bill does not exist
end if
payPayment()
u = updateBalance()
displayMainScreen() // Out of fund
end if
message(Payment fails)
removeBill()
notifyPayment()
message(Payment successful)
printCheque()
displayMainScreen()

Figure A.13: Sequence Diagram of Make Payment Use Case
A.4. Sequence Diagrams of Use Cases in TAS

Figure A.14: Sequence Diagram of Receive Payment Use Case
A. Case Study of Trading House Automation System

Figure A.15: Sequence Diagram of Process Supply Use Case
A.4. Sequence Diagrams of Use Cases in TAS

![Sequence Diagram of Process Pending Order Use Case](image)

**Figure A.16: Sequence Diagram of Process Pending Order Use Case**

break

[p = NULL]

message(No pending order)

displayMainScreen() // Return to the main screen

s = checkOrderValidity(

alt

s = Insufficient

createDueSlip(i)

placeDemand(i)

[Else]

v = checkOrderValidity()

opt

v = Cancelled

placeOrderQueue()

// Push the order into the queue of pending orders

message(Order is cancelled)

opt

v = Withheld

displayMainScreen()

// Return to the main screen

message(Order is placed)

Figure A.16: Sequence Diagram of Process Pending Order Use Case
A.5 Statechart Diagrams of Objects in TAS

Six modal classes are involved with the TAS system. These modal classes are Customer, Order, Item, Account, Supplier, and Bill. Here, we present the statechart diagrams of objects of each of these modal classes.

Figure A.17: Statechart Diagram of an Order Object

Figure A.18: Statechart Diagram of a Customer Object
A.5. Statechart Diagrams of Objects in TAS

Figure A.19: Statechart Diagram of a *Item* Object

Figure A.20: Statechart Diagram of a *Bill* Object

Figure A.21: Statechart Diagram of an *Account* Object
Figure A.22: Statechart Diagram of a Supplier Object
A.6 Class Diagrams of TAS

The TAS design has several controller, boundary and entity classes. In this section, we present the class diagrams depicting the relationships among these classes. In Fig. A.23, first-level view of the class diagrams showing the relationships among controller and boundary classes are given. In Fig. A.24, second-level view of the class diagrams showing the relations among controller classes and entity classes is shown. The third level-view showing the links among entity classes is shown in Fig. A.25.
A. Case Study of Trading House Automation System

Figure A.23: Class Diagram of TAS (First-level View)
Figure A.24: Class Diagram of TAS (Second-level View)
A. Case Study of Trading House Automation System

Figure A.25: Class Diagram of TAS (Third-level View)