**ACKNOWLEDGEMENTS**

We would like to express our respectful thanks to Dr.May Phyo Oo, Principal of University of Computer Studies (Banmaw), for allowing us to develop this System and encouraging us with wise counseling for the task ahead.

Our special thanks to Daw Aye Aye Thein, Associate Professor, Head of University of Computer Studies , Banmaw, for her valuable suggestions and editing on our project.

We would also like to express our deepest thanks to our supervisor, Daw Yi Mon Aung, Lectuter, Faculty of Information Science, University of Computer Studies, Banmaw, for her guidance and support. Her precious superior suggestions and supervisions are supportive on accomplishment of our project.

Last but not least, we extend out thanks to all our teachers and colleagues for their motivating encouragement and stimulating discussions about the project.

**ABSTRACT**

* Our project is “Restaurant Management System” which includes orders of the customers, customer information management, waiter and chief information management and menu information management and so on.
* The system wake to provide service facility to restaurant and also to the customers.
* This system is implemented to reduce the manual work and enhance the accuracy of work in the restaurant.
* And, the system manages and maintains the record of the customers and their order.
* Through the place ordering menu, the customer can simply order the food.
* Manager can manage employees, make salary payrole, manage annoucement and so on.
* Waitesr can record orders and service to the customers.
* Cashier can calculate the total amount of the orders and accept the bill from the customers.
* Chiefs make the meals, create new menu and accept the order menu to cook.

**Contents**

Title Page

1. Introduction 4
2. Objectives 4
3. Class Diagram 5
4. Use Case Diagram 8
5. Sequence Diagram 10
6. Collaboration Diagram 13
7. Statechart Diagram 14
8. Component Diagram 15
9. Code for Restaurant Management System
10. Conclusion 16
11. References

**INTRODUCTION**

* “Restaurant Management System” has been proposed to be implemented to replace the manual system.
* The main aim of this project is computerization of all processes which happens in the restaurant.
* It is the analysis and design of a software system.
* This system is design with class, use case ,sequence, collaboration, statechart and component diagrams.
* This system will be helpful for customers , waiters , chiefs and manger in the restaurant.

**OBJECTIVES**

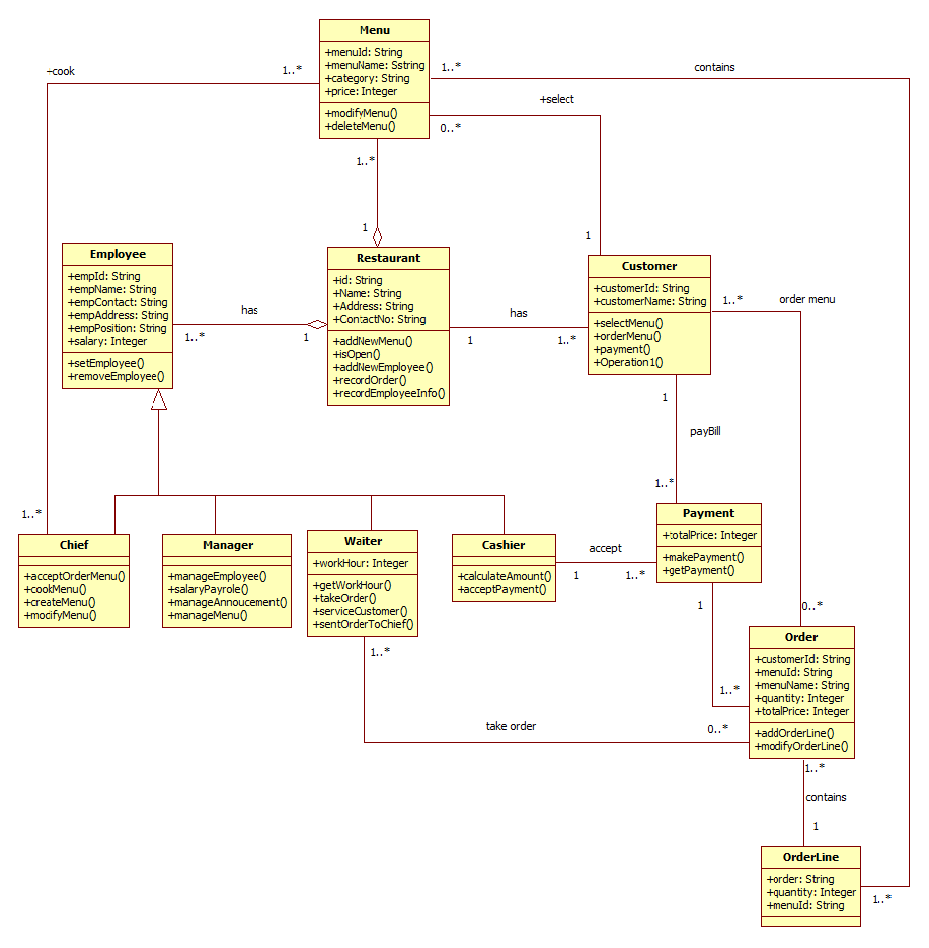
* To help restaurant administrator manage the restaurant business and help customer for ordering and reserve table.
* To describe how the system will perform and under which it must operate.
* To maintain food orders day to day record in the system.
* To create a more convenient design and focused for healthy.
* To develop ordering and reservation system in restaurant.

**Class Diagram**

In software engineering, a class diagram in the Unified Modeling Language(UML) is a type of static structure diagram that describes the structure of a system by showing the system’s classes, their attributes, operations (or methods), and the relationships among objects.

The class diagram is the main building block of object-oriented modeling. It is used for general conceptual modeling of the structure of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main elements, interactions in the application, and the classes to be programmed.

**Class Diagram of the Restaurant Management System**



**Description of Class Diagram**

The “Restaurant Management System” has eleven classes.The menu class consists of attributes such as menuId , menuName, category, price and two operation such as modifyMenu, deleteMenu. The Employee class inciude attributes such as empId, empName, empAddress, empPosition, empContact, salary and two operations such as setEmployee, removeEmployee .

The Restaurant class include attributes such as id,name,address,contactNo and five operations such addnewMenu, isOpen, addNewEmployee, recordOrder, recordEmployee. The Customerclass consists of attributes such as customerId,customerName and operations such as select Menu,orderMenu,payment. The Chief Class has four operations such as acceptorderMenu ,CookMenu,createMenu, modifyMenu.

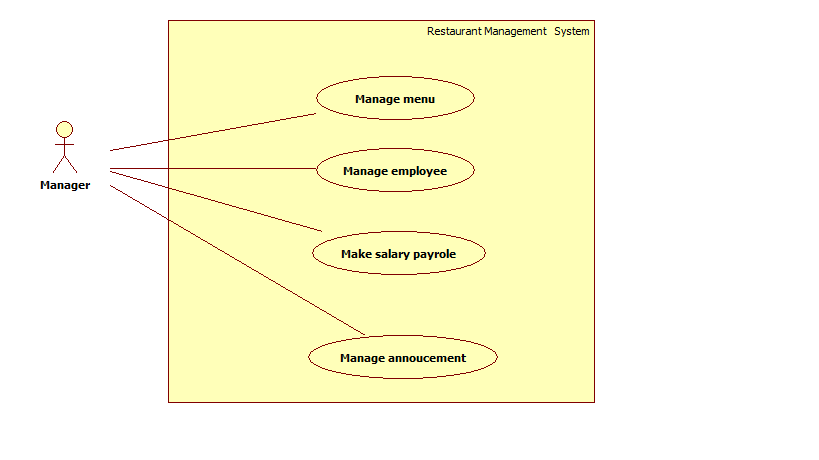
The manager Class has four operations such manage employee, salarypayrole, manageAnnoucement , manage menu.The Chief Class has four operations such as acceptOrderMenu ,cookMenu , createMenu , modifyMenu .The Waiter Class has only one attribute that workhour, and four operations such as getWorkHour , takeOrder , serviceCustomer , sentOrderToChief. The Cashier Class has two operations such as calculateAmount , acceptPayment .

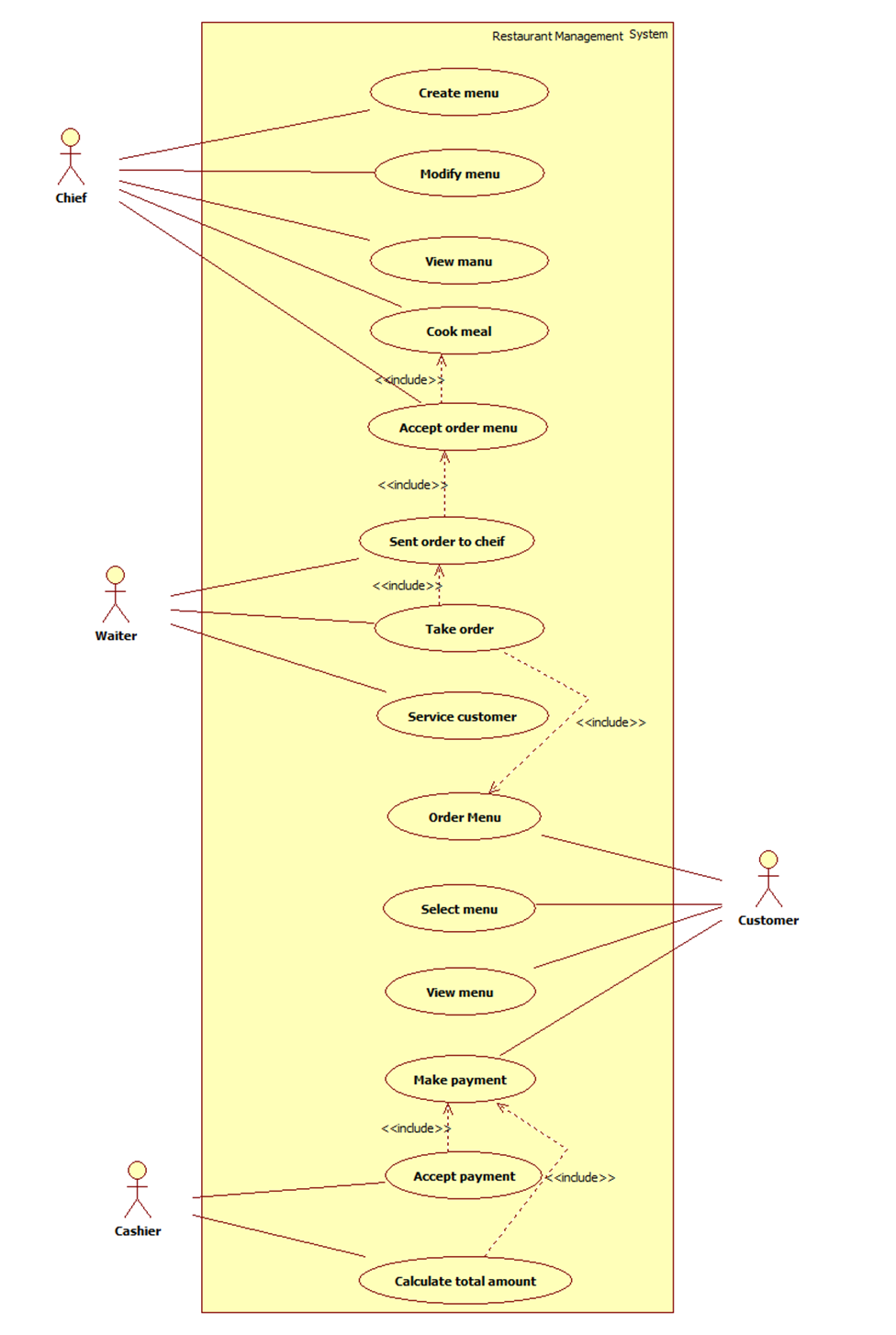
The Chief Class has four operations such as acceptOrderMenu ,cookMenu , createMenu , modifyMenu.The Payment Class has single attribute that totalPriceand consists of two operations such as makePayment, getPayment.The Order class include attributes such as customerId, menuId, menuName, quantity, totalprice and two operations such addOrderLine, modifyOrderLine. The Orderline class include attributes such as order, quantity, menuId.

**Use Case Diagram**

A use case diagram at its simplest is a representation of a user’s interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses.

**Use Case Diagram of the Restaurant Management System**





**Description of Use Case Diagram**

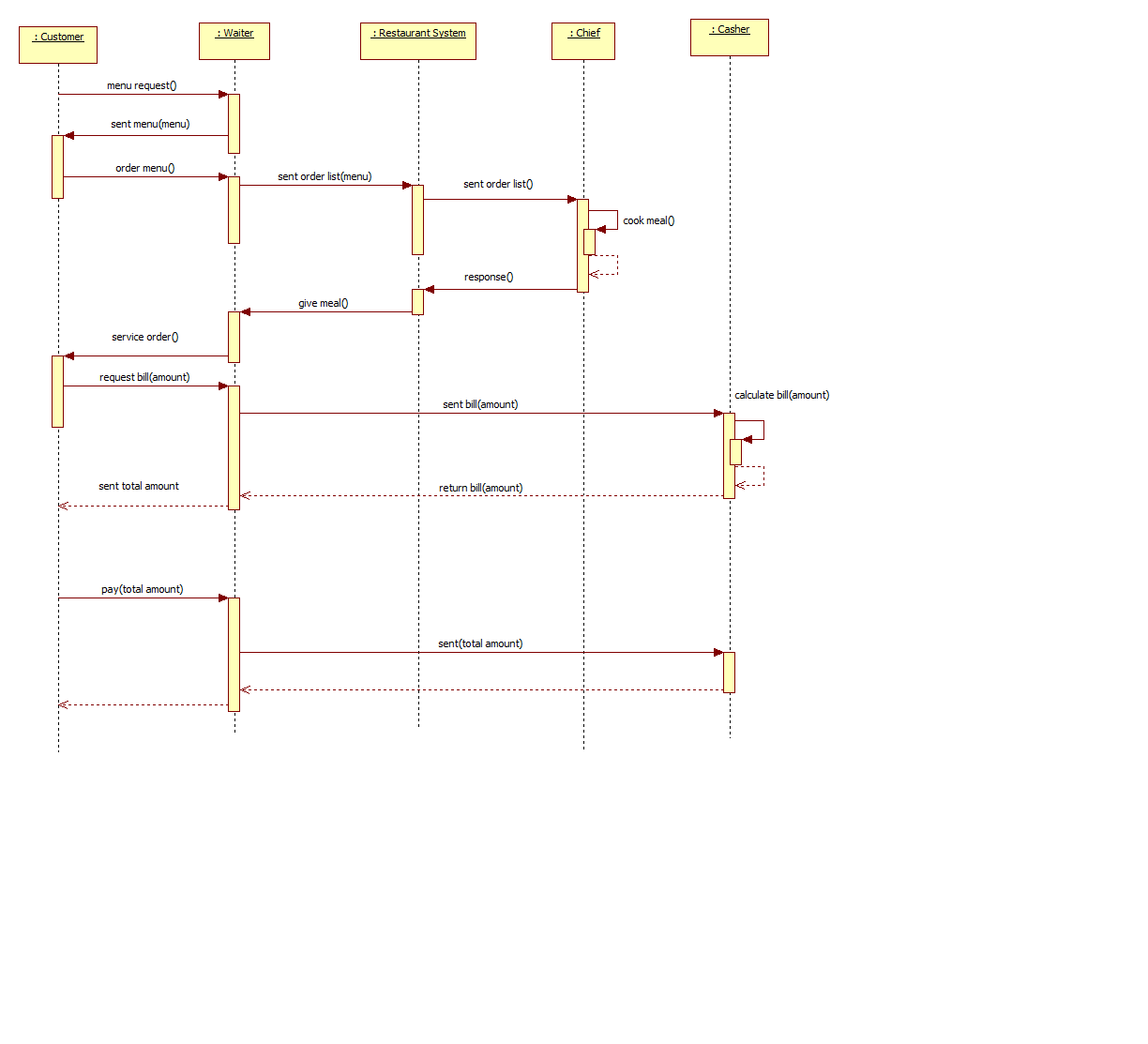
The Restaurant Management System has Five actors and eighteen usecases. Chief actor create menu, modify menu, view menu, cook meal, and accept order menu . Waiter actor send order to chief, take order, service customer. Customer actor order menu, select menu, view menu. Cashier actor accept payment, calculate total amount. Manager actor manage menu, manage employee, make salary, payrole, manage annoucement.

**Sequence Diagram**

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical view of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

A sequence diagram has two dimensions: as parallel vertical lines(lifelines), different processes or objects that live simultaneously , and as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner. Sequence diagram are time focus and they show the order of interaction visually by using the vertical axis of the diagram to represent time what messages are sent and when.

**Sequence Diagram of the Restaurant Management System**



**Description of Sequence Diagram**

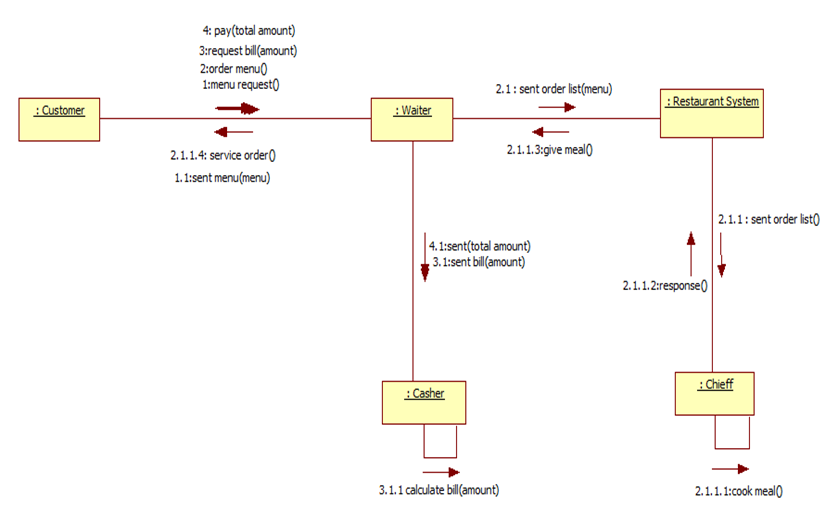
A UML Sequence diagram showing Restaurant Management System. Customer can see our restaurant management system overview. They can order via waiter from menu list. In our system admin can update,delete and insert menu and administrator can also change the restaurant of staff and manage the restaurant business. If customer delete or change their order they tell restaurant manager. Manager can manage their cancel problem. Manager can manage all other information. The customer can order one or more at that time. The restaurant of casher will calculate the total amount of their order. And then, the customer pay the total amount to the waiter. Finally, the casher receive payment from the waiter. Sequence diagram is collecting the major objects.

**Collaboration Diagram**

A collaboration Diagram, also known as a communication diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language(UML).Collaboration diagram is the dynamic aspect of the system and oftem comes in the form of a visual chart.Unlike the sequence diagram, collaboration sequencing cannot be shown graphically and messages are numbered to indicate order in which they are sent.

Messages can be numbered sequentially but more commonly a hierarchical numbering scheme is used. The rational for hierarchical numbering of messages to reflect the structure of nested activations that is made explict on sequence diagram. Collaboration diagrams do not include return arrows at the end of activations. Date that is returned from a message is prefixed to message name and separated from it by the assignment symbol ‘:=’.

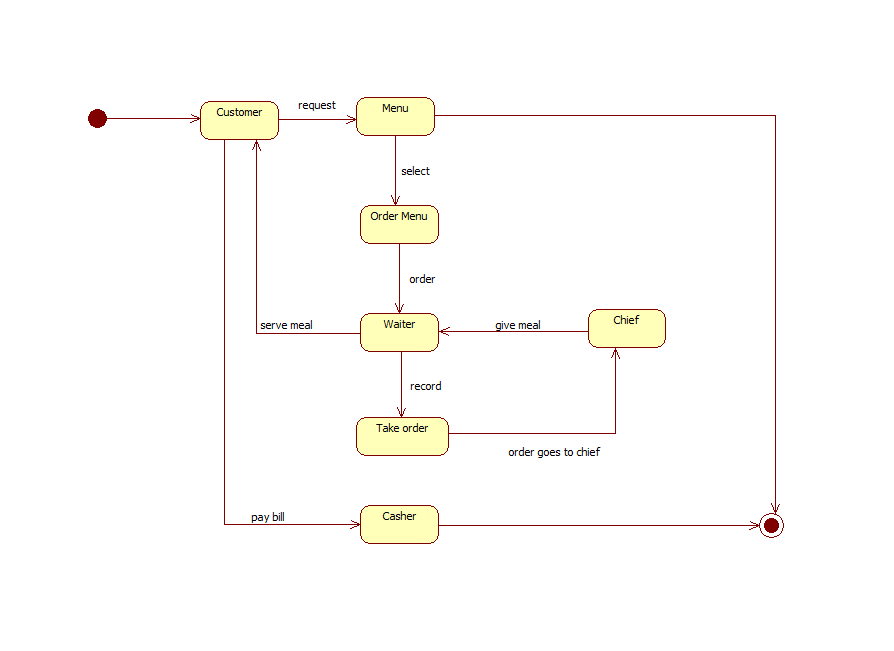
**Collaboration Diagram of the Restaurant Management System**



**Statechart Diagram**

A state diagram is a type of diagram used in computer science and related fields to describe the behaviour of systems. State diagrams require that the system described is composed of a finite number of states; sometiomes, this is indeed the case, while at other times this is a reasonable abstraction . Many forms of state diagrams exist, which differ slightly and have different semantics.

**Statechart Diagram of the Restaurant Management System**



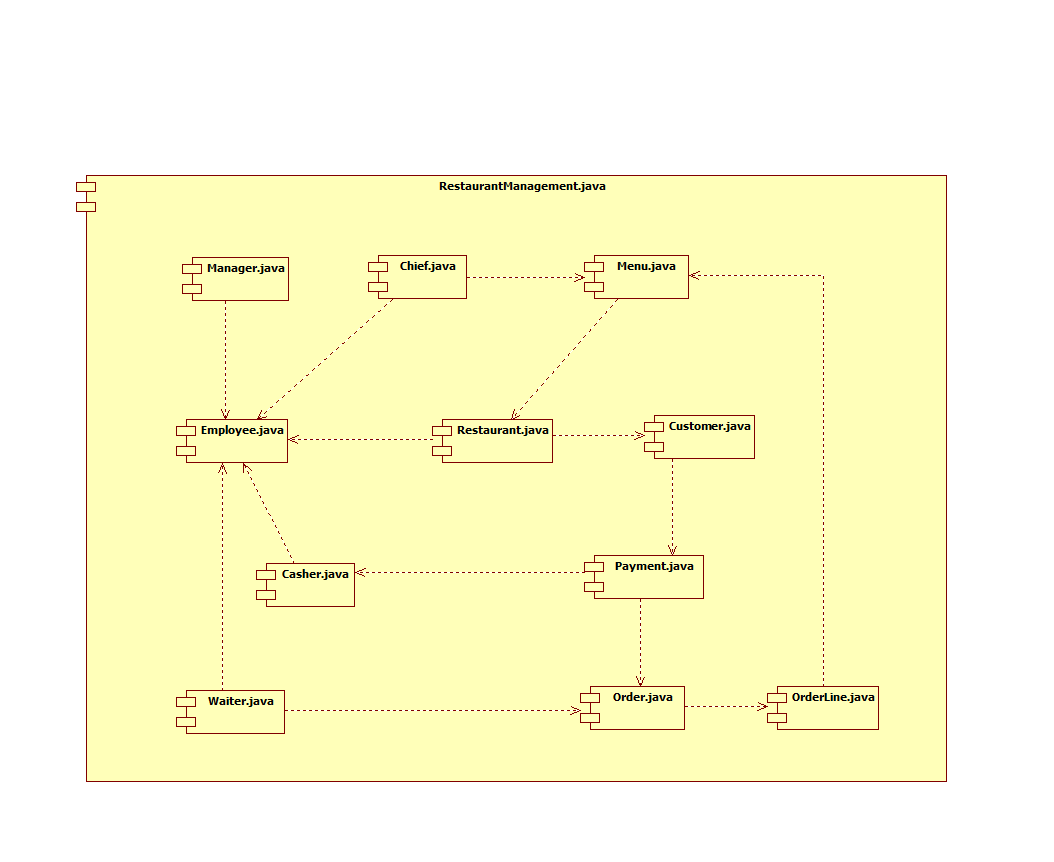
**Description of Statechart Diagram**

Customer request menu from waiter, select and order menu that he or she want. The waiter record order and send the order list to the chief. When the chief finish cooking, he gives the meal to the waiter. And then the waiter serves meal to the customer. If the customer want to pay the bill, he or she will go the cashier. The cashier will calculate the total amount and the customer will pay the bill.

**Component Diagram**

In unified modeling language(UML), a component diagram depicts how components wired together to form larger component software system. They are used to illustrate the structure of arbitrarily complex systems.

**Component Diagram of the Restaurant Management System**



**Coding for Restaurant Management System**

public class Cashier extends Employee

{

public void calculateAmount()

{ }

public void acceptPayment()

{ }

}

public class Chief extends Employee

{

public void acceptOrderMenu()

{ }

public void cookMenu()

{ }

public void createMenu()

{ }

public void modifyMenu()

{ }

}

public class Customer

{

public String customerId;

public String customerName;

public Menu select;

public void selectMenu()

{ }

public void orderMenu()

{ }

public void payment()

{ }

}

public class Employee

{

public String empId;

public String empName;

public String empContact;

public String empAddress;

public String empPosition;

public Integer salary;

public void setEmployee()

{ }

public void removeEmployee()

{ }

}

public class Manager extends Employee

{

public void manageEmployee()

{ }

public void salaryPayrole()

{ }

public void manageAnnoucement()

{ }

public void manageMenu()

{ }

}

public class Menu

{

public String menuId;

public Sstring menuName;

public String category;

public Integer price;

public Chief cook;

public void modifyMenu()

{ }

public void deleteMenu()

{ }

}

public class Order

{

public String customerId;

public String menuId;

public String menuName;

public Integer quantity;

public Integer totalPrice;

public void addOrderLine()

{ }

public void modifyOrderLine()

{ }

}

public class OrderLine

{

public String order;

public Integer quantity;

public String menuId;

}

public class Payment

{

public Integer totalPrice;

public void makePayment()

{ }

public void getPayment()

{ }

}

public class Restaurant

{

public String id;

public String Name;

public String Address;

public String ContactNo;

public void addNewMenu()

{ }

public void isOpen()

{ }

public void addNewEmployee()

{ }

public void recordOrder()

{ }

public void recordEmployeeInfo()

{ }

}

public class Waiter extends Employee

{

public Integer workHour;

public void getWorkHour()

{ }

public void takeOrder()

{ }

public void serviceCustomer()

{ }

public void sentOrderToChief()

{ }

}

**Conclusion**

Our system is constructed by six types of diagrams. The reason of using UML diagrams is largely graphical in form, because most people find such representations easier to work with than purely textual representations of complex structure. In our system, we apply use case, class, sequence, collaboration, statechart and component diagrams in representing the specific object, function, method and process. We just get a lot of skills such as analyzing, designing and concrete structuring by using and kind of UML diagram.

**References**

* Mark Priestly, “Practical Object\_Oriented Design With UML”, Second Edition.