Project: Fast Food Automated Ordering System

Students: Vimal Mayank and Deep Saraf
Faculty Advisors: Mark Austin and John Baras

TABLE OF CONTENTS

Introduction

Problem Statement

This case study looks at the problem of setting up a fast food restaurant. The basic problem in the food service industry is that restaurants are not realizing efficiencies that would result from better applications of technology in their daily operations. Every fast food has counter where you can place your order and then make the payment. So every fast food needs an employee for taking the order and processing the payment. Labor rates are increasing every now and then and it is difficult to find employees in the middle of the highway, hence to solve this problem we plan to design a "Self Served Fast Food System."

This self-service fast food restaurant will be equipped with a user-friendly touch screen, a credit/debit card reader, and software for completing the process at the backend. For this system there will be a system administrator who will have the rights to enter the menu with their current prevailing prices. He/she can enter anytime in the system by a secured system password to change the menu contents by adding or deleting an item or changing its price.

Now when the customer enters the restaurant, he will place his order with the help of the touch screen using the intuitive graphical user interface, right from the selection of language till the payment confirmation. He will select from the food options according to his choice and the system will display the payment amount he has to make once he has finished with his order. He will have the option of paying the bill by cash, debit card or a credit card. The user will slide his card and the system will check for the validity of the card and the payment will be made. A receipt will be printed containing the order number and the order will be sent in the kitchen for processing.

Anticipated Benefits

1. This will minimize the number of employees at the back of the counter.
2. The system will help to reduce the cost of labor.
3. The system will be less probable to make mistake, since it?s a machine.
4. This will avoid long queues at the counter due to the speed of execution and number of optimum screens to accommodate the maximum throughput.
5. The system will be available 24 hours for 365 days, because the machine is not going to take any sick or vacation leave.

Scope and Objectives

The purpose of this analysis is to demonstrate the extent to which high-level systems concept and UML notation/semantics can be used to describe the functionality of this system. This study lays out a framework for a new system to be developed and brought to the market for maximum use. The following issues, which are useful in performing a detailed analysis of the system, will be addressed in this study:

1. What should the system do?
2. What are the systems requirements?
3. How does the system work?
4. Can the system work?
5. What objects should be chosen and each of the subsystems functionality?
6. How should the object/subsystem interact?
7. How to verify and validate the system?

System Framework and Boundary

Goals, Scenarios and Use Cases

Goals and Scenarios

Goal 1. The system has a user-friendly user interface.

1. Scenario 1.1. A particular user of the system has no difficulty in reading the text on the display.
2. Scenario 1.2. The system is navigable through intuition.
3. Scenario 1.3. Menu choices are presented in form of buttons, which contain text as well as little pictures illustrating the choice for better understanding.

Goal 2. The system supports multi lingual capabilities.

1. Scenario 2.1. User is not a native speaker of the common spoken language in the country.
2. Scenario 2.2. System caters to English (universally accepted language for communication) and other native languages, which are commonly spoken in the country where the system is operative.

Goal 3. System takes order from the customer as per his/her choice.

1. Scenario 3.1. User selects a combo deal (i.e. a combination of main food, drink and side dish).
2. Scenario 3.2. User may want to make up his own order by selecting dishes.

Goal 4. The system calculates and displays the final bill based on the placed order.

1. Scenario 4.1. System calculates final bill based on the quantity of the items multiplied by their unit price topped up by the applicable taxes if any.
2. Scenario 4.2. User is given the option to either pay for the order or revise the order.

Goal 5. System handles the payment for the user-defined order.

1. Scenario 5.1: User decides to pay cash and system asks user to enter cash in the slot.
2. Scenario 5.2: System verifies the cash amount and gives refund if any after deducting the amount.
3. Scenario 5.3: User decides to pay through credit/debit card. System informs user to swipe card through card reader.
4. Scenario 5.4: The system verifies the card and charges the amount of the bill to the card. Asks user to sign the bill on the signing pad.
5. Scenario 5.5. The system prints out receipt containing a token number, details of the order, bill and the payment method with a terminal message (Thank you visit again or Store address).
6. Scenario 5.6: System communicates the order to the kitchen through the internal ordering system.

Goal 6. System offers the choice to change the menu items to the store manager.

1. Scenario 6.1. Store manager decides to add / delete an item from the menu.
2. Scenario 6.2. Store manager wants to put festive offers on some items because of which there is a change in the price of some of the items.

3. Scenario 6.3. Store manager notices that some dishes are out of stock. Consequently he updates the menu so that those items are deleted temporarily deleted from the display presented to the user.

Goal 7. The system is resistant to active/rigorous handling.

1. Scenario 7.1. The customer must be a minor or hacker who might want to get into the system to change it. It should provide security for these areas.
2. Scenario 7.2. The system will be used by different users and might experience some rough hand every now and then.

Identify Actors

An actor is anything that interfaces with the system externally and participates in use case modeling. In our Self Served Fast Food System the actors would be:

1. **Customer.** This actor is the principle customer who will order food and make the payment.
2. **Store Manager.** This actor will hold the rights to change the menu and enter the system to make any changes.
3. **Internal Order System.** This actor will read the order given by the customer and pass it to the food preparation person.
4. **Bank System.** This actor serve as a backbone for doing the credit / debit card transaction.
5. **Cash Collector.** This actor will accept the cash from the customer and gives back the change.
6. **Food Preparation Person.** This person receives the order placed by the customer through internal order system.

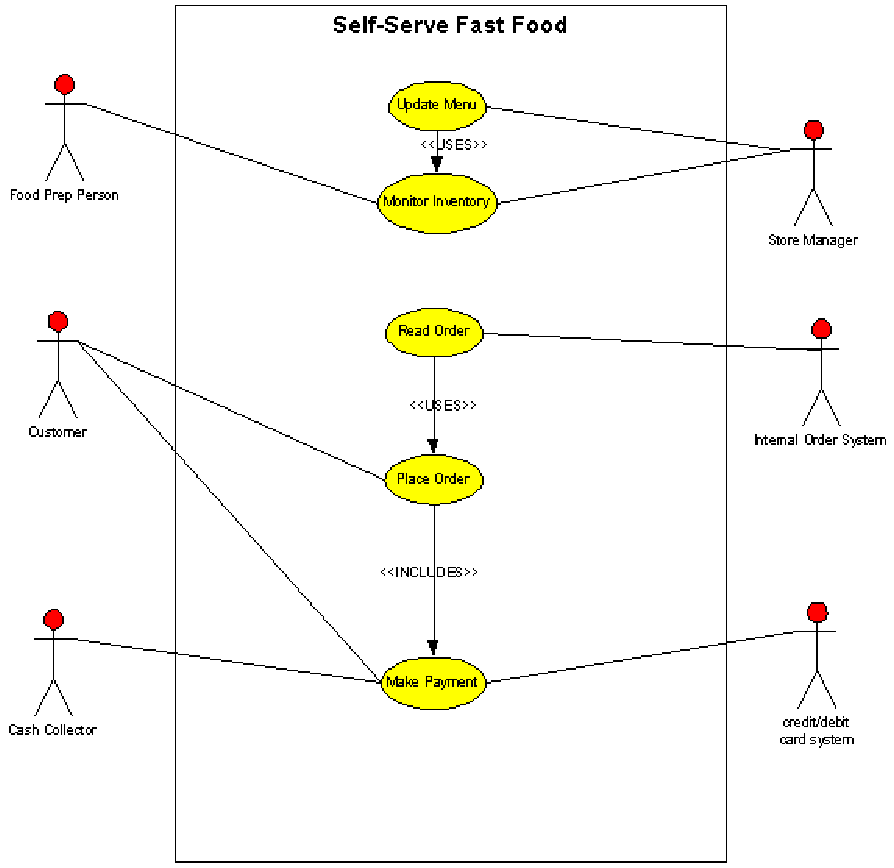
System Boundary

The system boundary is defined by the elevator itself.

Initial Use Case Diagram

A use case describes a single goal and all the things that can happen as the user attempts to reach that goal. Although use cases are neither requirements nor functional specification, they imply requirements, objects and object interactions in the stories they tell. Use cases are textual description of the interaction between external actors and a system.

Our initial use case diagram has six actors and five use cases.



Initial Use Case Diagram

Figure 1. Initial Use Case Diagram for Serve Fast Food.

As you can see that the Customer places the order, which is read by the internal order system, and the order is then sent to the food preparation person for execution. In the mean time the customer makes the payment. If the payment is made in cash there is a cash collector actor or there is a bank system for credit/debit card processing and keeping the balance cash amount.

Baseline (Textual) Use Cases with Activity Diagrams

When the flow of events is linear, a textual description of behavior is often sufficient to capture the system behavior. Activities diagram provide a visual documenting sequence of task making up a single activity. They especially are useful for activities governed by conditional logic, and flow of event running concurrently. We describe the basic system functionality with textual use cases, and employ activity diagrams for a visual representation of the corresponding sequence of task or flow of information.

Use Case 1. Place Order

- **Primary Actor:** Customer
  Description: Customer places an order from the available choices after indicating his language preference for the session.
  Pre-conditions: System is connected to a power source, display is turned on and system is configured to accept the inputs.
  Flow of Events:
  1. User selects his language preference for the session.
  2. User selects from the menu.
  3. User selects from the drinks menu
  4. User selects from the combo deals
  5. User confirms the order

  Alternative Flow of Events:
  1. User accidentally presses a wrong button and after realizing it he hits the backspace button.
  2. User enters a wrong order and wants to go back to the main menu.

  Post-condition: Order has been made that goes to the kitchen for processing.
  Assumption: User is familiar with how to enter values through mouse and has a general idea why the inputs are being provided and what is expected out of system.

Activity diagram for this use case is given as below:

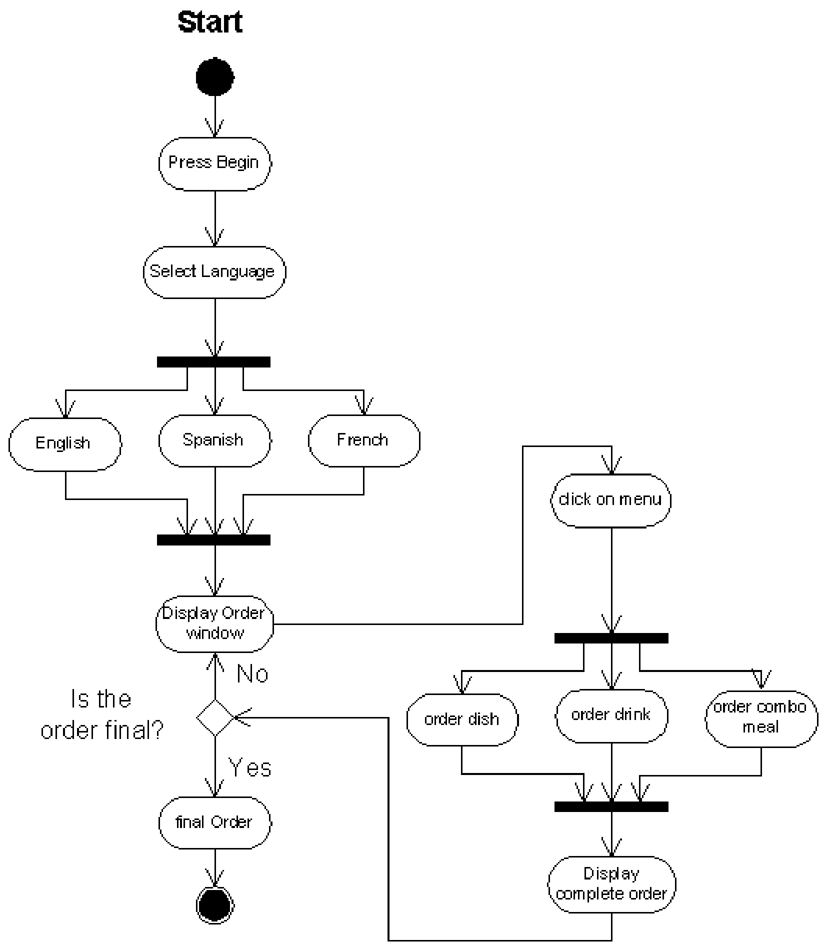


Figure 2. Activity Diagram for Place Order.

Use Case 2. Make Payment

- **Primary Actors:** customer, Credit/Debit system, cash collector.
  Description: The user is asked for the mode of payment. The payment is accepted in terms of credit/debit card or is collected by cash collector. And the customer is given a token with their order number.
  Pre-condition: The order has been confirmed and the total bill has been displayed on the screen to the customer. Costumer decides to go ahead with the order. **Flow of Events:**
  1. User enters the mode of payment. (Credit/Debit/Cash)
  2. User makes the payment in cash
  3. Cash collector collects the money and gives back the change if required.
  4. User makes the payment by credit/debit card.
  5. User receives a token number and final bill.

  Alternative Flow of Events:
  1. User selects the mode of payment.

  Post-condition: Customer waits for the order to be processed.
  Assumption: User is familiar with how the system works and what is expected out of system.

Activity diagram for this use case is given as below:

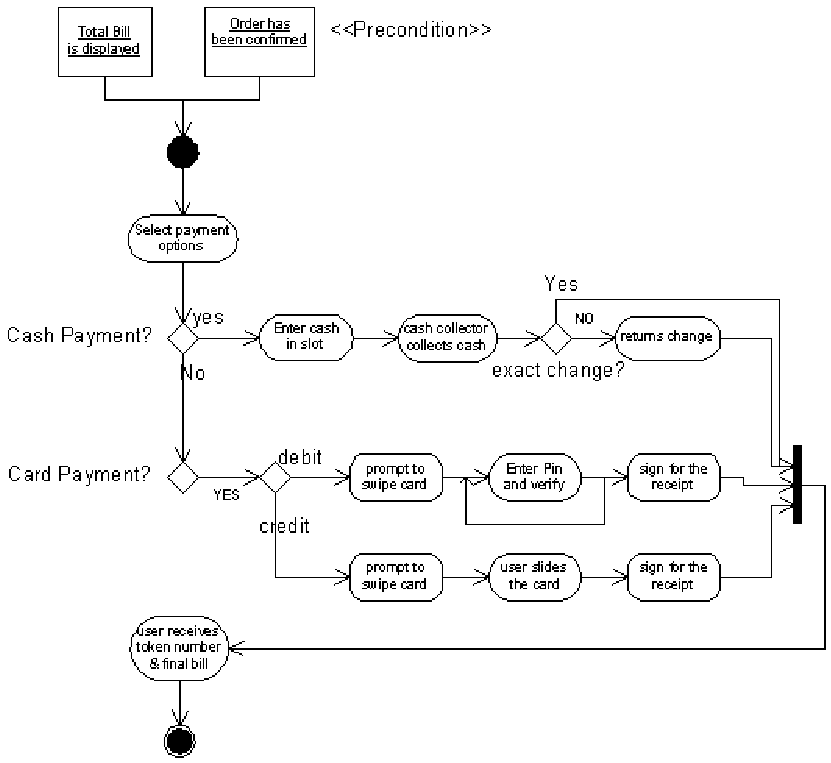


Figure 3. Activity Diagram for Payment Process Use Case

Use Case 3. Update Menu.

- **Primary Actor:** Store Manager.
  Description: The menu might change according to the inventories or add/delete items from menu and deals. The prices of each item might change for the period of time.
  Pre-condition: An order menu with their respective price already exists in the system in some particular format.
  Flow of Events:
  1. The Store manager enters the system with some password.
  2. The Store manager makes the required changes.
  3. The Store manager saves the changes and logs out.

  Alternative Flow of Events:
  1. Some of the menu might not need any change.
  2. User might enter invalid password and need to go back.

  Post-condition: A menu list will be displayed when the user enters the system.
  Assumption: The Store manager is given the rights and privileges to enter the system and make the required changes.

Activity diagram for this use case is given below:

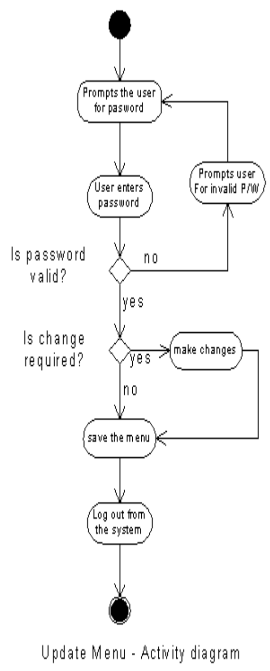


Figure 4. Activity diagram for Update Menu Use Case

Use Case 4. Monitor Inventory.

- **Primary Actor:** Food preparation person, Store Manager **Description:** This use case triggers when an item goes out of stock.
  Pre-condition: None
  Flow of Events:
  1. Food preparation person/Store manager notices an item out of stock
  2. Updates the menu accordingly.

  Post-condition: A new and updated menu list will be displayed.
  Assumption: The Store manager is given the rights and privileges to enter the system and make the required changes.

Activity diagram for this use case is given below:

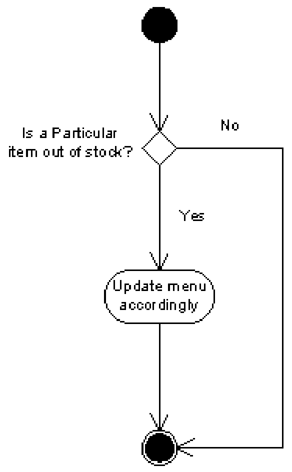


Figure 5. Activity Diagram for Monitor Inventory

Use Case 5. Read Order.

- **Primary Actor:** Food preparation person, Internal Order system.
  Description: Internal order system reads the order once the customer confirms his order and then he communicates the order to the food preparation person. **Pre-condition:** User confirms the order.
  Flow of Events:
  1. Internal order system reads the order
  2. Communicates the order to the food preparation person

Post-condition: The final order is being processed in the kitchen.
Assumption: Food preparation person is available to take the order and know the sequence of processing the orders.

Activity diagram for this use case is given below:

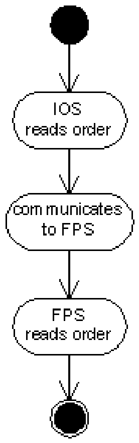


Figure 6. Activity diagram for Read Order

Use Case Task Component Interaction

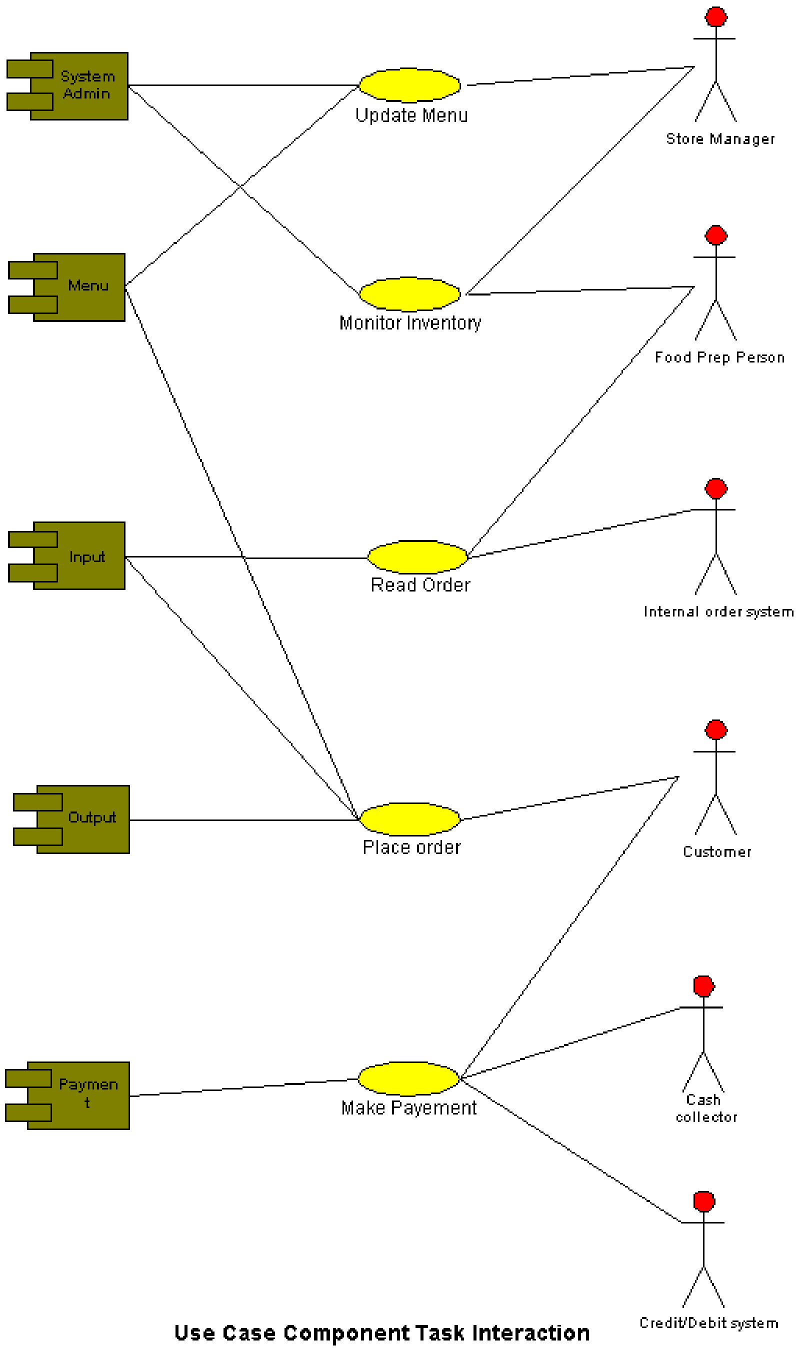


Figure 7. Use Case Task Component Interaction

Generation of Requirements from Use Cases

Having generated the baseline textual use cases and the scenarios for the current problem we can now generate the requirements for the Self Served Fast Food system. Requirements are derived from various goals and scenarios, use cases so it is important to trace back the source of requirement.

High-Level Requirements (ENSE 621 version)

These are the first draft of requirements written during the beginning of the semester when the system engineering principles were not very clear. These requirements are not quantified and are very ambiguous. They are not specific to any object or process and hence needs to be refined.

User Requirements

1. User should be able to navigate the system without any difficulty.
2. System supports native language of the country and other commonly spoken languages.
3. User should be able to place order according to his choices
4. User should be able to make payment using cash/credit/debit card.
5. User should get a receipt and a token number after making the payment.

Sources: Goals 1,2 & 5, Scenario 1.1,3.1, & 5.4, Use Case 1 & 2.

Performance Requirements

1. The system should be able to take any type of inputs, once the mouse is clicked on the respective button.
2. The system should be able to take any amount of order and display it when finished.
3. The system should be able to calculate the bill and prompt the user for the mode of payment and generate a receipt.
4. The system should be able to pass on the order in the kitchen for processing.
5. The system should be secured to restrict the number of people to enter the system to make changes in the menu and its items.
6. The system should be sturdy for rough usage.
7. System has a cash collector which gives refund up to 5 $ in coins.
8. System should be able to communicate to the central database to verify the authenticity of the credit/debit card.
9. System should allow Store manager to add/delete/alter system items.

Sources: Goals 5,6 & 7, Scenario 4.1 & 4.2, Use Case 2,3,4 & 5

User Interface Requirements

1. The system must be a graphical user interface for easy use and understanding.
2. The system must be able to prompt the user for the next step to be performed during the process of using the system.
3. The system must display the bill and final order for confirmation.

Sources: Goal 1, Scenario 4.3 & 5.4, Use Case 1 & 4

Ergonomics Requirements

1. The system interface layout must be self-explanatory
2. Horizontal and vertical distances between two adjacent buttons should be at least 5 pixels for better visibility and accessibility.
3. The mouse should be placed not below 3 feet above the ground.

Sources: Goal 1, Scenario 1.2 & 1.3, Use Case 1 & 4.

Requirements Traceability

Flowdown of Requirements from Use Cases/Scenarios

The detailed flowdown of requirements from use cases and scenarios is as follows (ENSE 621, version):

SOURCE		DESTINATION	
Use Case	**Scenario**	**Requirement No**	**Description**
Place Order	1.2	User 1	Easy navigability
	2.1, 2.2	User 2	Support of native language and others
	1.3, 3.1, 3.2	User 3	User places order as per his choice
	3.1, 3.2	Performance 1,2	Take any input and amount of order
	4.1	Performance 3	Able to calculate bill, prompt for mode of payment and give receipt.
	1.3, 1.1	User Interface 1	Supports GUI for easy use.
	1.2	User Interface 2	Prompts user for next step
	4.1	User Interface 3	Display bill and confirm order
	1.1, 1.3	Ergonomics 1	Self explanatory system interface
	1.3	Ergonomics 2	5 pixel distance b/w buttons
Make Payment	1.2	User 1	Easy navigability
	5.1, 5.3	User 4	Make payment using cash/credit/debit cards.

	5.5	User 5	Generation of receipt/token no.
	4.1	Performance 3	Able to calculate bill, prompt for mode of payment and give receipt.
	5.2	Performance 7	Equipped with cash refund device.
	5.4	Performance 8	Ability to connect with bank central database.
	1.2	User Interface 2	Prompts user for next step.
	4.1	User Interface 3	Display bill and confirm order.
	1.1 & 1.3	Ergonomics 1	Self explanatory system interface.
	1.3	Ergonomics 2	5 pixel distance b/w buttons Update Menu.
	7.1	Performance 5	Restricted access to change system.
	6.1, 6.2 & 6.3	Performance 9	Ability to add /delete / alter menu items.
	1.1 & 1.3	Ergonomics 1	Self explanatory system interface.
	1.3	Ergonomics 2	5 pixel distance b/w buttons Monitor inventory.
	6.1, 6.2 and 6.3	Performance 9	Ability to add /delete / alter menu items Read Order.
	5.6	Performance 4	Communicate order to kitchen.

Traceability of Requirements to Use Cases / Scenarios

Traceability from requirements back to originating use cases/scenarios is as follows (ENSE 621 version):

SOURCE		**DESTINATION**	
Requirement No	**Description**	**Scenario**	**Use Case**
User 1	Easy navigability	1.2	1,2
User 2	Support of native language & others.	2.1, 2.2	1
User 3	User places order as per his choice.	1.3, 3.1 & 3.2	1
User 4	Make payment using cash/credit/debit cards	5.1 & 5.3	2
User 5	Generation of receipt/token no.	5.5	2
Performance 1	Take any input.	3.1 & 3.2	1
Performance 2	Take any amount of order.	3.1 & 3.2	1
Performance 3	Able to calculate bill, prompt for mode of payment and give receipt.	4.1	1,2
Performance 4	Communicate order to kitchen.	5.6	5
Performance 5	Restricted access to change system.	7.1	3
Performance 6	Sturdy system for rough usage.	7.2	None
Performance 7	Equipped with cash refund device.	5.2	2

Performance 8	Ability to connect with bank central database.	5.4	2
Performance 9	Ability to add /delete / alter menu items.	6.1, 6.2 & 6.3	3,4
User Interface 1	Supports GUI for easy use.	1.1 & 1.3	1
User Interface 2	Prompts user for next step.	1.2	1,2
User Interface 3	Display bill and confirm order.	4.1	1,2
Ergonomics 1	Self explanatory system interface	1.1 & 1.3	1,2,3
Ergonomics 2	5 pixel distance b/w buttons.	1.3	1,2,3
Ergonomics 3	Vertical placing of mouse.	1.2	None

High-Level Requirements (ENSE 622 version)

Concept Requirement List (CRL) are the set of requirements which demonstrate the over all system, its needs and outputs. These are the set of requirements generally provided by the management to the designers. Following are the set of CRL for the fast-food ordering system that we plan to design and develop.

1. System supports native language of the country and other commonly spoken languages.
2. User should be able to place order according to his choices
3. User should be able to make payment using cash / credit / debit card.
4. User should get a receipt and a token number after making the payment.
5. The system should be able to take any type of inputs, once he touches the respective button.
6. The system should be able to calculate the bill and prompt the user for the mode of payment and generate a receipt.
7. The system should be able to pass on the order in the kitchen for processing.
8. The system should be secured to restrict the number of people to enter the system to make changes in the menu and its items.
9. The system should be sturdy for rough usage.
10. System has a cash return mechanism which gives refund up to 5 $ in coins.
11. System should be able to communicate to the central database to verify the authenticity of the credit/debit card.
12. System should allow Store manager to add/delete/alter system items.
13. The system must be a graphical user interface for easy use and understanding.
14. The system must be able to prompt the user for the next step to be performed during the process of using the system.

Generation of Specifications

Synthesis and Breakdown of Requirements

Detailed model of Requirements Flowdown : System-level requirements are assigned to elements in the system architecture, which in turn, flow down to subsystem elements. Appropriate test requirements are generated at each level of the system development. Starting ate the sub-system level, other stakeholder requirements are taken into account, perhaps because the subsystem elements will be used across a product line. Designers working on the sub-system elements provide feedback to the system-level designers.

We divide our system at different levels of requirements, to make sure that all the requirement specifications are covered and study the flowdown of requirements. You will notice that there are the requirements at the system level, which are very similar to the higher level requirements and then we study the requirements at the Sub-System level and component level. Later, in the project we will see how these requirements are tested and verified at there levels and then we integrate them to test the whole system.

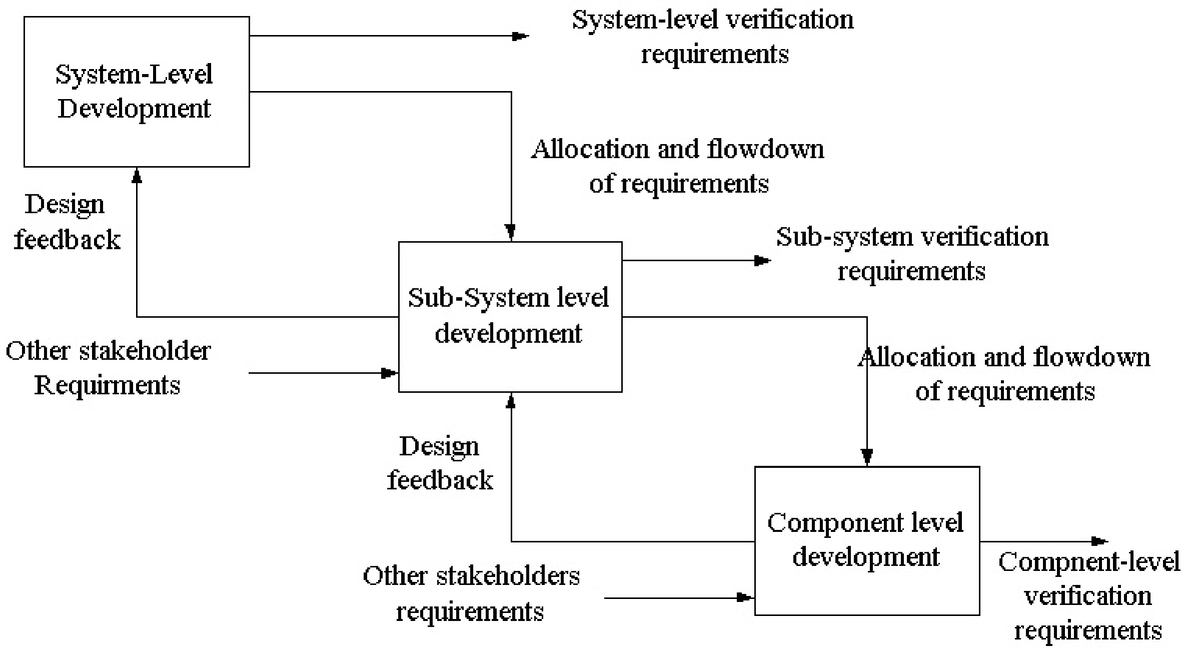

System-Level Development
System-level verification requirements
Allocation and flowdown of requirements
Design feedback
Sub-System level development
Sub-system verification requirements
Other stakeholder Requirments
Allocation and flowdown of requirements
Design feedback
Component level development
Other stakeholders requirements
Compnent-level verification requirements

Figure 8. Detailed model of Requirement Flowdown

1.0 System Level Requirements

System-level requirements are assigned to elements in the system architecture, which in turn, flow down to subsystem elements. Appropriate test requirements are generated at each level of the system development.

1.1. Restaurant will be open for 16 hours and will operate in four shifts as Morning (730 AM -1130 AM), Afternoon (1130 AM -330 PM), Evening (330 PM - 730 PM) and Night (730 PM -1130 PM).

1.2. System should be able to serve a throughput of 50, 125, 50, 75 customers per hour during these four shifts respectively.

1.3. Cooks, cleaners and assemblers will be the type of employees working the restaurant.

1.4. Cook will be paid at the rate of $5 per hour for the duration of their work.

1.5. Assemblers will be paid at the rate of $4.75 per hour for the duration of their work.

1.6. Cleaner will be paid at the rate of $4.5 per hour for the duration of their work.

1.7. Customer will leave without ordering if he sees 6 or more people in the line waiting to be served thereby causing a loss in revenue.

2.0 Sub-system Level Requirements

Starting at the sub-system level, other stakeholder requirements are taken into account, perhaps because the subsystem elements will be used across a product line. Designers working on the sub-system elements provide feedback to the system-level designers.

(a) 2.1 Ordering and Processing System

2.1.1. The system will provide queue management by passing orders sequentially to kitchen (FIFO) by assigning order numbers to them.

2.1.2. Customer should be able to order item either by name or by number (for combo deals).

2.1.3. System will support native and other commonly spoken language in the country.

2.1.4. An average order takes about 1 minute to complete with a variance of 10 seconds.

2.1.5. Customer takes 2 minutes on an average to complete and ordering process.

2.1.6. System will be equipped with a standard QWERTY keyboard for taking inputs.

(b) 2.2 Payment System

2.2.1. This Order number will be printed on the bill that customer receives.

2.2.2. System shall be able to accept cash and coins.

2.2.3. System should be able to accept debit / credit cards.

2.2.4. System will accept only $1, 5, 10, 20 bills and nickel, dime, quarter denomination of coins.

2.2.5. System will reject pennies & $50, $100 bills.

2.2.6. System won't allow putting in $1 coin.

2.2.7 Cash return will return change only in coins (up to $5 maximum)

3.0 Component Level Requirements

(a) 3.1 Touch Screen

3.1.1. Touch screen should be able to take inputs from users when they apply a light pressure (x lb) from their fingers corresponding to an item.

3.1.2. Touch screen should be able to withstand rough use i.e. it will be scratch proof (i.e., will be resistant to nail scratching) and will not malfunction if a pressure of (x+5) lb is applied to the screen.

3.1.3. Touch screen will colored for visual appeal and will support at least 256 colors.

3.1.4. Resolution of the screen will be at least 640X480 pixels.

3.1.5. Touch screen will be at least 10 in size.

3.1.6. Touch screen should be able to display at least 15 rows and 60 columns of text when the font size of the text is 10.

(b) 3.2 Card Acceptor/Reader

3.2.1. System will be equipped with a card reader with built it keypad (containing all digits and special function keys corresponding to CANCEL, OK, # etc).

3.2.2. This card reader should be able to read the card information if swiped at a speed > 1 m/s.

3.2.3. Card reader will have a vertical slot on the right hand side of the keypad.

(c) 3.3. Display

3.3.1. System will display all the menu items in icons/graphics format for selecting.

3.3.2. System will prompt the user for mode of payment.

3.3.3. Order will be transferred to the kitchen touch screen instantaneously once the user does the payment.

3.3.4. System will display the order sequentially on the kitchen screen with a forward and back button at the bottom.

3.3.5. Touching an order on the kitchen screen will prompt the system that the order has been delivered and the screen will be rolled forward.

3.3.6. Any error message during such a process will be reported to the user on the touch screen informing him to take any further action.

3.3.7. There will be a high contrast between the foreground and the background of the display for easy reading capabilities.

3.3.8. Throughout the ordering process all the text displayed on the screen will be either greater than 10 or less than 18 font size so that all users (young, adult, old) can read it.

(d) 3.4. Modem

3.4.1. Modem should be able to complete a transaction (dialing, sending information, receiving information) in 15 seconds or less at all times.

3.4.2. Once connected to the bank system will supply the card information to the bank database, will query the card validity and will supply the amount to be charged to the card.

(e) 3.5 Software

3.5.1. System will be secured to grant access rights only to the system administrator. For this a login ID and a password will be assigned which could be changed. Password won't be visible to onlookers while typing for increased security.

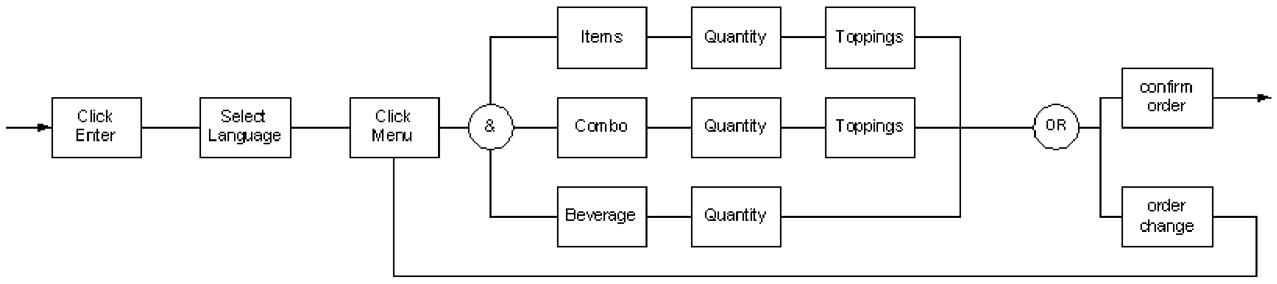
3.5.2. System will deny access to change the contents if the login/password both are incorrect by providing an error message

System Modeling and Analysis

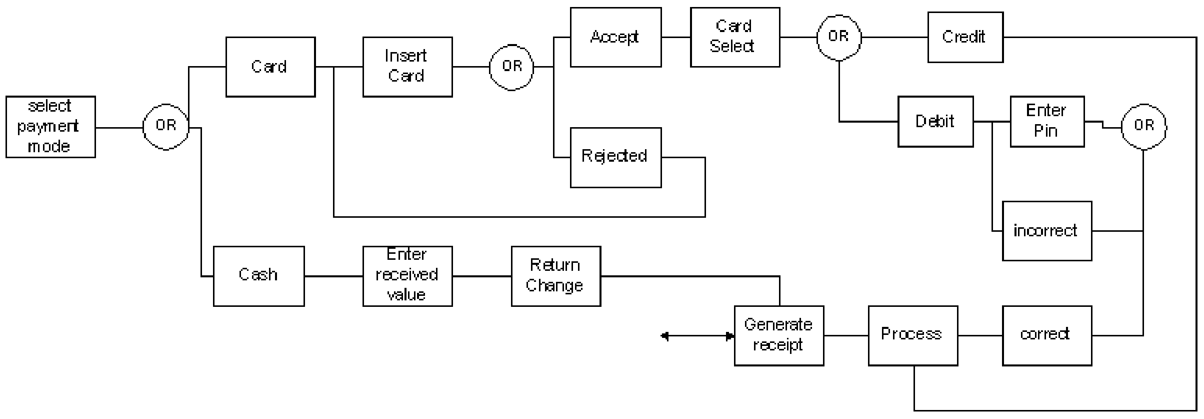
System Behavior

System behavior shows what a system does or appears to do, Its represented graphically by a model which integrates the functional model and the inputs and outputs. The figures below shows two versions of two different functional models. Our first model is the Order System and the second model show the payment activity. The 621 version is the basic version of the model with most of function not very well defined which you will notice when you view the 622 version of the Activities.

FFBD's for Order and Payment (ENSE 621 version)



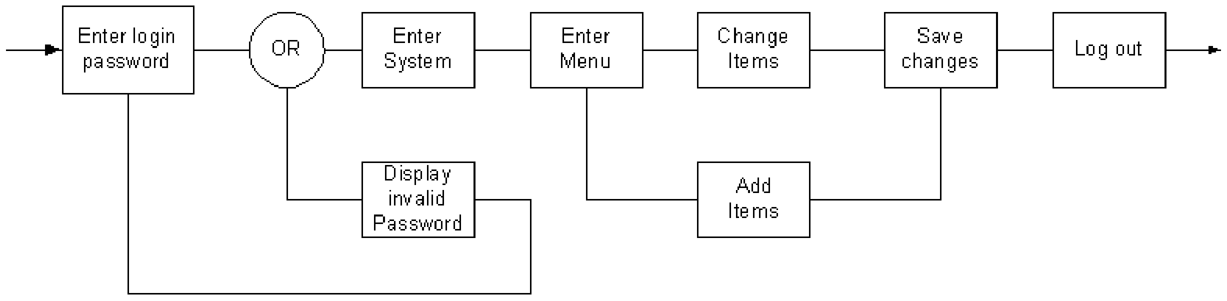
FFBD For Order



FFBD For Payment

Figure 9. FFBD's for Order and Payment (ENSE 621 Version)

FFBD for System Updates (ENSE 621 version)



FFBD For System Updates

Figure 10. FFBD for System Updates.

FFBD for Pay Bill (ENSE 622 version)

This is a more detailed functional flow block diagram of the sequence of functions that take place in the Order and Payment processes.

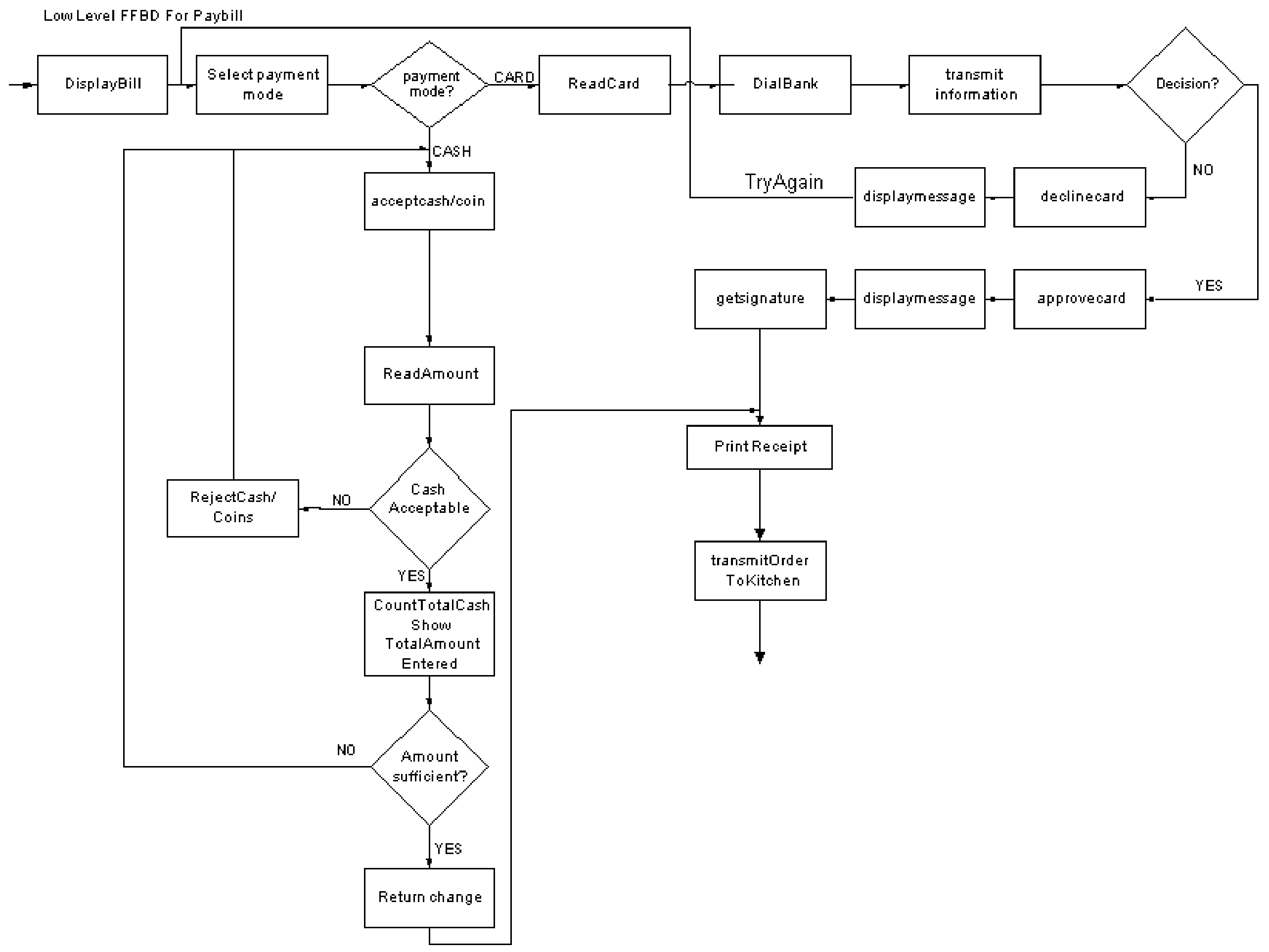


Figure 11. FFBD for Pay Bill (ENSE 622 version)

FFBD for System Administrator (ENSE 622 version)

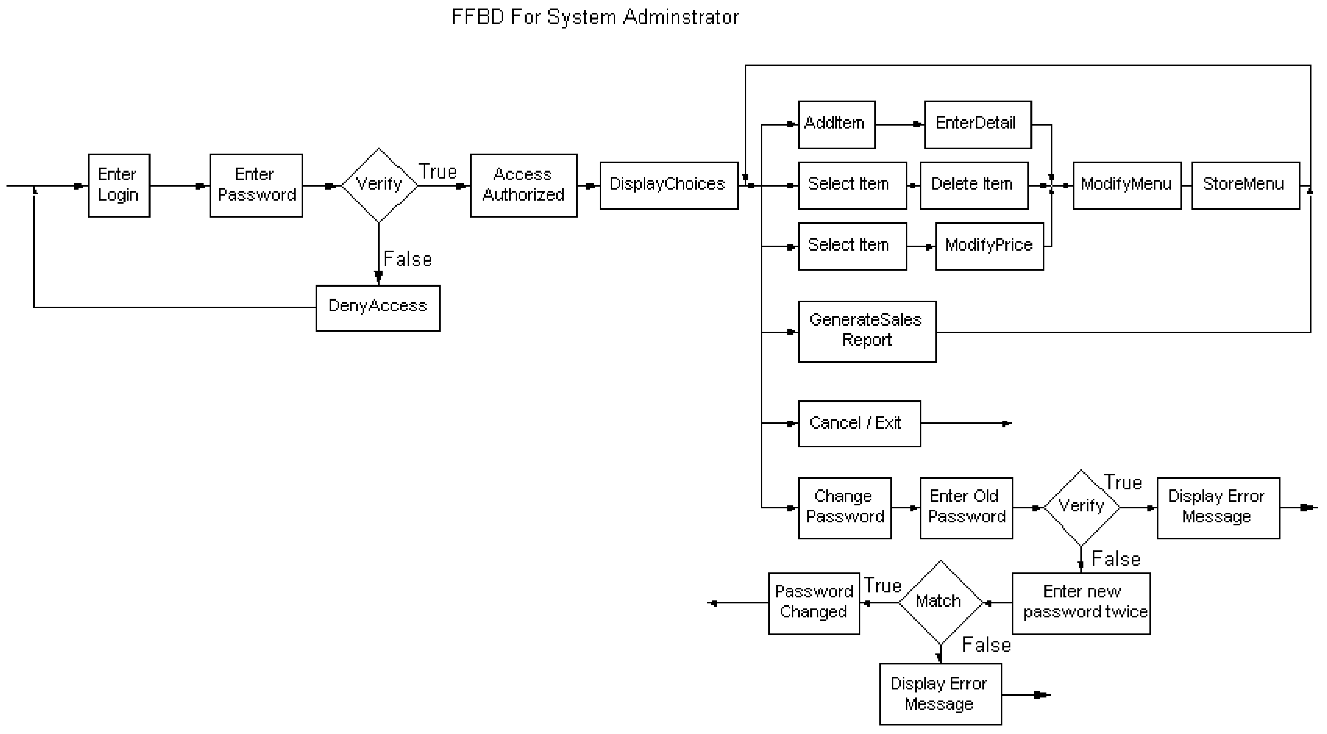


Figure 12. FFBD for System Administrator (ENSE 622 version)

Statechart Diagram

A statechart diagram (STD) describes the possible states of a single class and the events that cause state transitions. They are useful for showing the life cycle of the class. Statechart and activity diagrams both describe state transitions and share many of the same elements. An activity diagram is most effective for describing processes that involve more than one object. The following diagram shows the statechart diagram of our system.

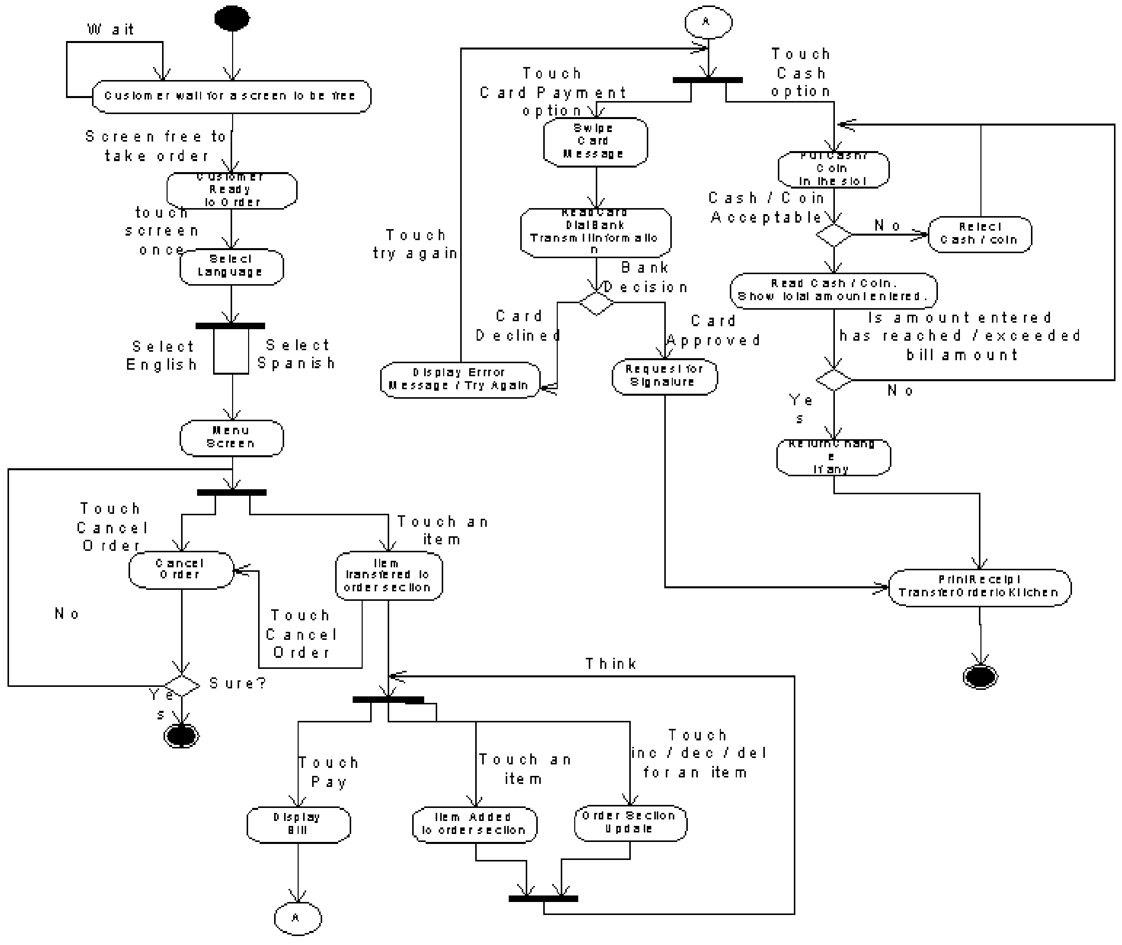

Wait
Customer wait for a screen to be free
Screen free to take order
Customer Ready to Order
touch screen once
Select Language
Select English
Select Spanish
Menu Screen
Touch Cancel Order
Touch an item
Cancel Order
Item transferred to order section
No
Touch Cancel Order
Sure?
Yes
Think
Touch Pay
Touch an item
Touch inc / dec / del for an item
Display Bill
Item Added to order section
Order Section Update
A
Touch Card Payment option
Touch Cash option
Swipe Card Message
Read Card Dial Bank Transmit Information
Touch try again
Bank Decision
Card Declined
Card Approved
Display Error Message / Try Again
Request for Signature
Put Cash / Coin in the slot
Cash / Coin Acceptable
No
Reject Cash / coin
Read Cash / Coin. Show total amount entered.
Is amount entered has reached / exceeded bill amount
No
Yes
Return Change if any
Print Receipt Transfer Order to Kitchen

Figure 13. Statechart diagram ....

System Structure

The model of system structure has evolved through two versions.

Preliminary System Structure (ENSE 621 version)

In the preliminary implementation (ENSE 621), the "automated ordering system" was partitioned into elements for input, menu, payment, system administration and output.

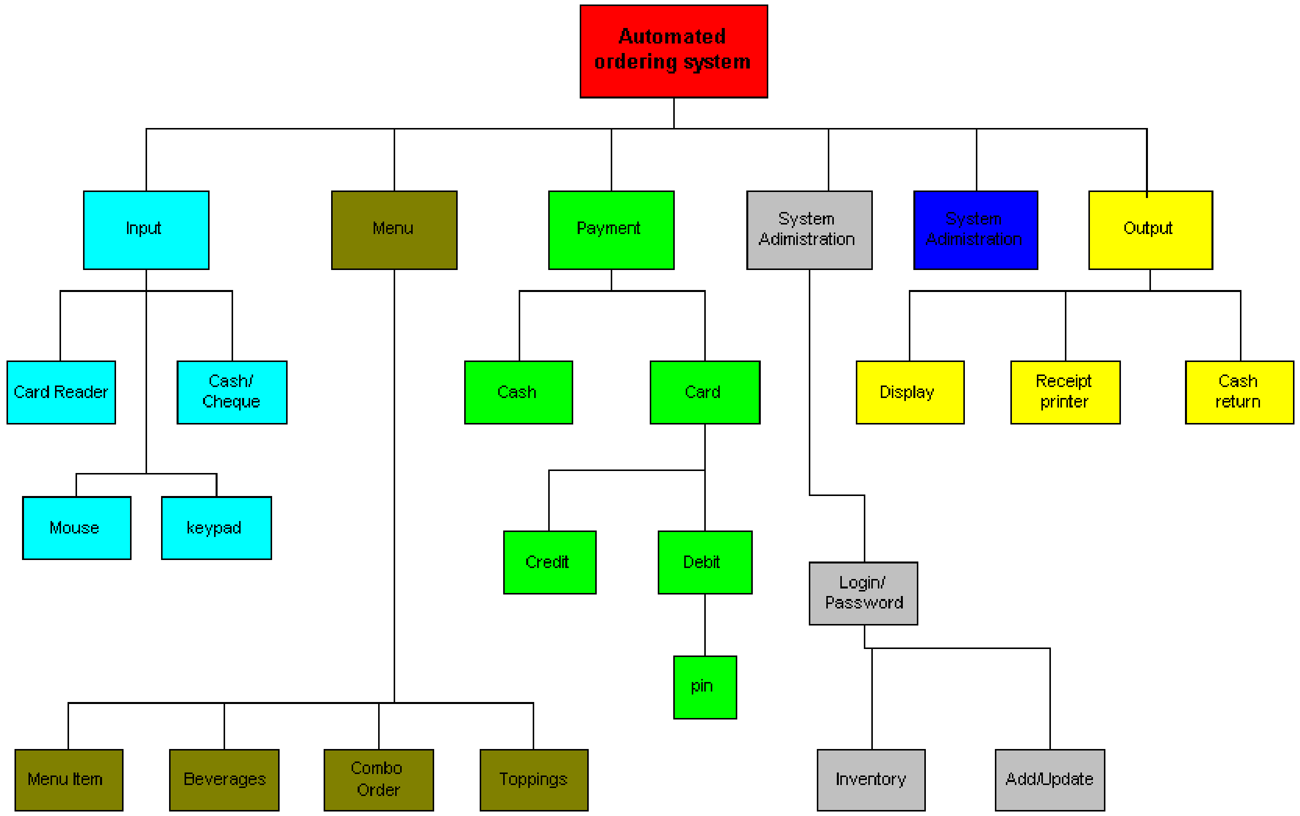


Figure 14. System Structure (ENSE 621 Version).

Notice that the system structure does not include the environment within which the ordering system works and also some of the objects that are considered at lower level are not objects of the system, instead they are attributes.

Revised System Structure (ENSE 622 version)

Version two of the system structure has a larger scope. The highest-level of system structure, "fast food system" is a composition of three systems -- staff, kitchen and automated food ordering system.

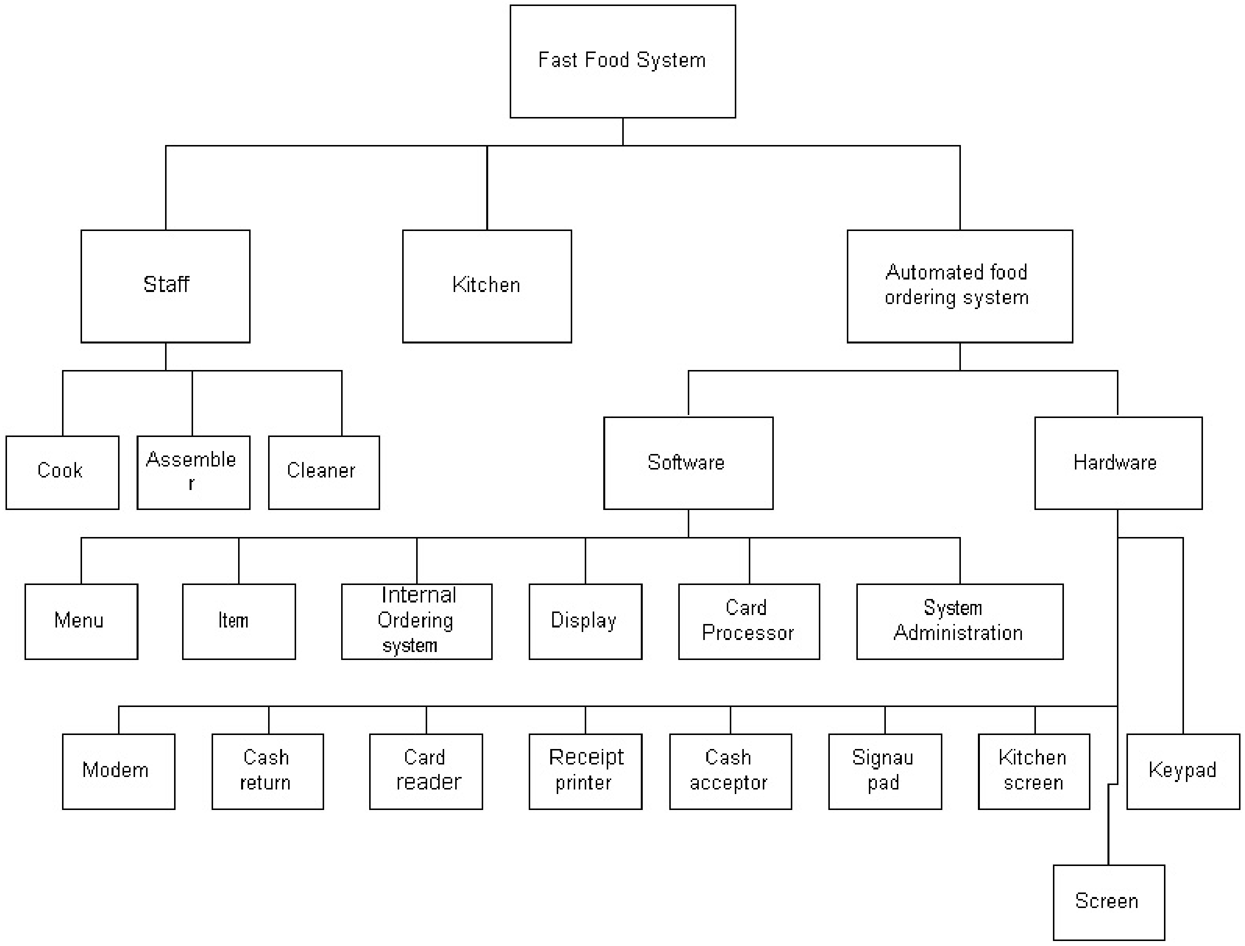


Figure 15. Revised System Structure (ENSE 622 version)

Attributes and functions are assigned to classes in the system hierarchy, as shown in Figure 15.

System-Level Design

We now map chunks of behavior onto the system structure and show the flow of messages/data among system components with sequence diagrams. As you can see in the diagram below the system structure have been revised again to trace all the functions of the system behavior. We have also added the attributes related to each object in the system structure.

Assigning Fragments of Behavior to the Object Structure

System-Level Design (ENSE 622 version)

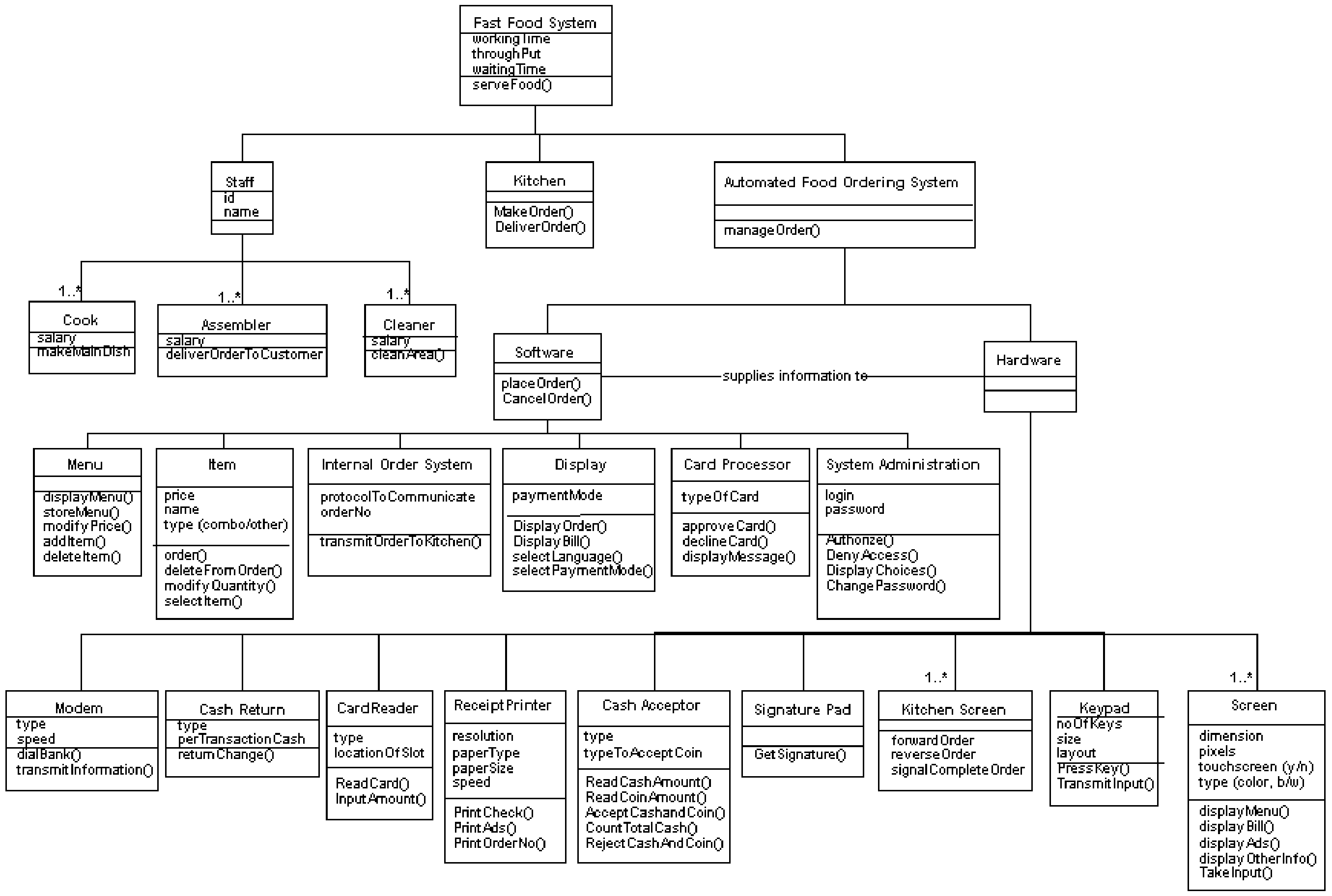


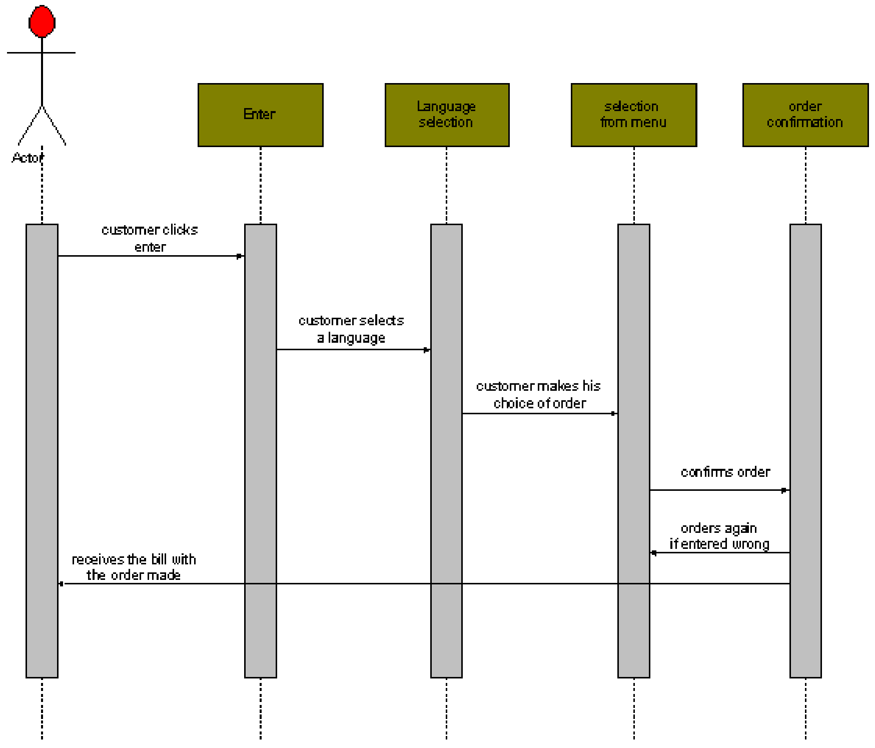
Figure 16. System Structure

Fragments of Behavior for the System-Level Design

Sequence diagram provides a graphical representation for how a task is accomplished by passing a sequence of messages among objects. These interactions define behavior as implemented by the fragments of the system structure.

Our system can be divided into two different sequence diagram as shown below:

Sequence Diagrams for Placing an Order (ENSE 621 version)

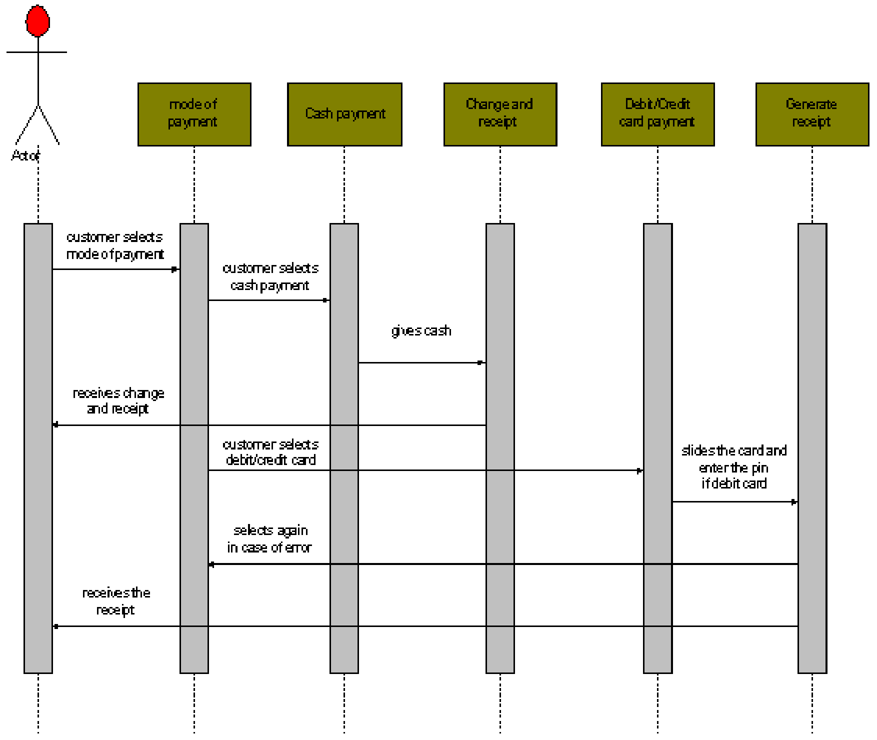

Actor
Enter
Language selection
selection from menu
order confirmation
customer clicks enter
customer selects a language
customer makes his choice of order
confirms order
orders again if entered wrong
receives the bill with the order made

Sequence Diagram for placing the Order

Figure 17. Sequence of Messages for Placing an Order

As seen in the diagram the customer enters the system by clicking begin button and selects his language choice. He then clicks on the menu button to see the menu items and then makes a selection. He then confirms his order by clicking on the confirm order button.

Sequence Diagrams for Making the Payment (ENSE 621 version)


Actor
mode of payment
Cash payment
Change and receipt
Debit/Credit card payment
Generate receipt
customer selects mode of payment
customer selects cash payment
gives cash
receives change and receipt
customer selects debit/credit card
slides the card and enter the pin if debit card
selects again in case of error
receives the receipt

Sequence Diagram for making the Payment

Figure 18. Sequence of Message for Making the Payment

Customer has been prompted for the mode of payment. If he selects cash then he has to give cash to the cash collector and receive change, if any. He also receives a receipt from the cash collector. If the customer selects debit/credit card payment mode, he is asked again to select from debit or credit. If he selects debit card then he is asked a pin number or else he is asked to slide the card. After checking for the validity of the card the payment is made and the customer receives a receipt.

Traceability of Requirements to Attributes and Functions

System Level Requirements	Object	Attribute	Function
1.1. Restaurant will be open for 16 hours and will operate in four shifts as Morning (730 AM -1130 AM), Afternoon (1130 AM -330 PM), Evening (330 PM - 730 PM) and Night (730 PM -1130 PM).	Fast Food System	Time	
1.2. System should be able to serve a throughput of 50, 125, 50, 75 customers per hour during these four shifts respectively.	Fast Food System	Throughput	
1.3. Cooks, cleaners and assemblers will be the type of employees working the restaurant.	Employee	type	
1.4. Cook will be paid at the rate of $5 per hour for the duration of their work.	Cook	Salary	
1.5. Assemblers will be paid at the rate of $4.75 per hour for the duration of their work.	Assemblers	Salary	
1.6. Cleaner will be paid at the rate of $4.5 per hour for the duration of their work.	Cleaner	Salary	
1.7. Customer will leave without ordering if he sees 6 or more people in the line waiting to be served thereby causing a loss in revenue.	Fast Food System	Waiting time	
Sub-System Level Requirements	**Object**	**Attribute**	**Function**
2.1.1. The system will provide queue management by passing orders sequentially to kitchen (FIFO) by assigning order numbers to them.	Internal Ordering System	Order No	TransmitOrder ToKitchen
2.1.2. Customer should be able to order item either by name or by number (for combo deals).	Item	Type/Name	Order
2.1.3. System will support native and other commonly spoken language in the country.	Display	Language	
2.1.4. An average order takes about 1 minute to complete with a variance of 10 seconds.	Kitchen		Make Order
2.1.5. Customer takes 2 minutes on an average to complete and ordering process.			place order
2.1.6. System will be equipped with a standard QWERTY keyboard for taking inputs.	Keypad	layout	
2.2.1. This Order number will be printed on the bill that customer receives.	Receipt Printer	Order No	PrintOrderNo
2.2.2. System shall be able to accept cash and coins.	Cash Acceptor	Type	AcceptCash-AndCoin
2.2.3. System should be able to accept debit / credit cards.	Card Reader	Type	ReadCard
2.2.4. System will accept only $1, 5, 10, 20 bills and nickel, dime, quarter denomination of coins.	Cash Acceptor	TypeTo AceeptCash-AndCoin	AcceptCash-AndCoin
2.2.5. System will reject pennies & $50, $100 bills.	Cash Acceptor		RejectCash AndCoin

2.2.6. System won't allow putting in $1 coin.	Cash Acceptor		RejectCash AndCoin
2.2.7. Cash return will return change only in coins.	Cash Return	type	returnChange
Component Level Requirements	**Object**	**Attribute**	**Function**
3.1.1. Touch screen should be able to take inputs from users when they apply a light pressure (x lb) from their fingers corresponding to an item.	Touch Screen		TakeInput
3.1.2. Touch screen should be able to withstand rough use i.e. it will be scratch proof (i.e., will be resistant to nail scratching) and will not malfunction if a pressure of (x+5) lb is applied to the screen.	Touch Screen		
3.1.3. Touch screen will colored for visual appeal and will support at least 256 colors.	Touch Screen	type	
3.1.4. Resolution of the screen will be at least 640X480 pixels.	Touch Screen	pixels	
3.1.5. Touch screen will be at least 10" in size.	Touch Screen	dimension	
3.1.6. Touch screen should be able to display at least 15 rows and 60 columns of text when the font size of the text is 10.	Touch Screen	dimension	
3.2.1. System will be equipped with a card reader with built it keypad (containing all digits and special function keys corresponding to CANCEL, OK, # etc).	Card Reader	type	
3.2.2. This card reader should be able to read the card information if swiped at a speed > 1 m/s.	Card Reader		ReadCard
3.2.3. Card reader will have a vertical slot on the right hand side of the keypad.	Card Reader	LocationOf Slot	
3.2.4. System will read the swiped card, will retrieve the total bill amount and initiate the modem to dial the bank to complete the transaction.	Modem / Card Processor		Dial Bank
3.3.1. System will display all the menu items in icons/graphics format for selecting.	Display		selectItem
3.3.2. System will prompt the user for mode of payment.	Display	Payment Mode	SelectPayment Mode
3.3.3. Order will be transferred to the kitchen touch screen instantaneously once the user does the payment.	Internal Ordering System	ProtocolTo-Communicate	TransmitOrder ToKitchen
3.3.4. System will display the order sequentially on the kitchen screen with a forward and back button at the bottom.	KitchenScreen		ForwardOrder ReverseOrder
3.3.5. Touching an order on the kitchen screen will prompt the system that the order has been delivered and the screen will be rolled forward.	KitchenScreen		SignalComplete-Order
3.3.6. Any error message during such a process will be reported to the user on the touch screen informing him to take any further action.	Display		ReportError Message
3.3.7. There will be a high contrast between the foreground and the background of the display for easy reading capabilities.	Display		
3.3.8. Throughout the ordering process all the text displayed on the screen will be either greater than 10 or less than 18 font size so that all users (young, adult, old) can read it.	Display	Text/Size	
3.4.1. Modem should be able to complete a transaction (dialing, sending	Modem	Speed	Dial Bank/

information, receiving information) in 15 seconds or less at all times.			Transmit Information
3.4.2. Once connected to the bank system will supply the card information to the bank database, will query the card validity and will supply the amount to be charged to the card.	Modem		Transmit Information
3.5.1. System will be secured to grant access rights only to the system administrator. For this a login ID and a password will be assigned which could be changed. Password won't be visible to onlookers while typing for increased security.	Software	Login/ Password	Authorize
3.5.2. System will deny access to change the contents if the login/password/ both are incorrect by providing an error message.	System Administrator/ Software		Deny Access

Design Structure Matrix

Systems engineering of product, process, and organizations require tools and techniques for system decomposition and integration. A design structure matrix provides a simple compact and visual representation of a complex system that supports innovative solutions to decomposition and integration problems. The techniques of DSMs have led to there increasing use in a variety of contexts, including product development, project planning, project management, systems engineering, and organization design.

There are two main categories of DSMs: **Static** and **Time-based.** Static DSMs represent system elements existing simultaneously, such as components of a product architecture or groups in an organization. In time-based DSMs, the ordering of the rows and columns indicate a flow through time: upstream activities in a process precede downstream activities, and terms like 'Feedforward' and 'Feedback' become meaningful when referring to interfaces.

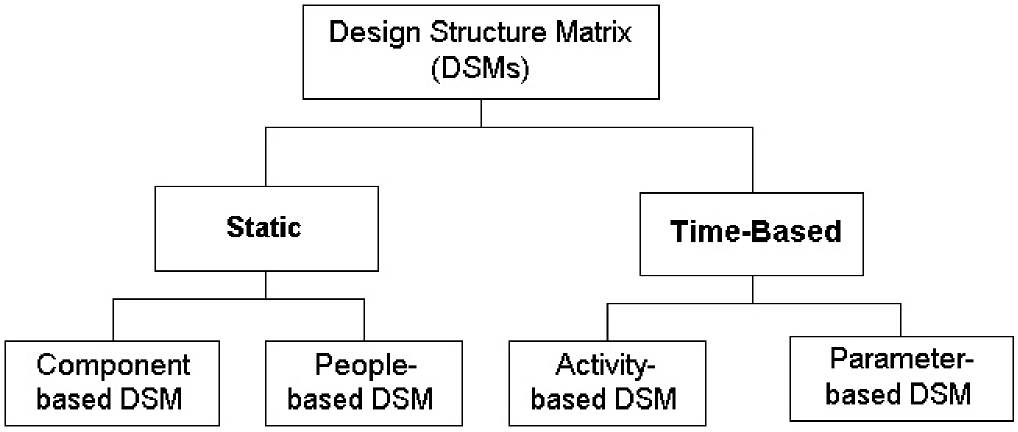


Figure 19. DSM Taxonomy

1. Component -Based or Architecture DSM: Used for modeling systems architectures based components and/or subsystems and their relationships.

2. Team-Based or Organization DSM: Use for modeling organization structures based on people and/or groups and their interactions.

3. Activity-Based or Schedule DSM: Used for modeling processes and activity based networks based on activities and their information flow and other dependencies.

4. Parameter-Based (or Low-Level Schedule) DSM: Used for modeling low-level relationships between design decisions and parameters, systems of equations, subroutine parameter exchanges.

Now let us apply one of the DSM techniques to our existing system. Let us consider the Pay Bill activity of our system and apply the **'Information Flow based Process Modeling' using the *Activity-Based DSM.*** Processes - especially product development processes are complex systems. A prerequisite to process improvement is process understanding. Process structure or architecture affects process efficiency and effectiveness. Therefore, process architecture can be an important source of competitive advantage. Improved understanding of process architecture can be gained by using process models, particularly ones that support process decomposition and integration analysis. Process decomposition requires and understanding of process activities and their interfaces, because the interfaces are what give a process its added value. therefore process models must capture flows.

Modeling a process requires two representation steps, followed by integration analysis

1. Decompose the process into activities

2. Document the information flow among the activities
3. Analyze the sequencing of the activities into a maximally-feed-forward process flow.

Activity-Based DSM of the payment design process:

Activities		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Display Bill	1		X															
Select Mode of payment	2			X										X				
Read Card	3				X													
Dial bank	4					X												
Transmit information	5						X		X									
Decline card	6							X										
Display error message	7		X															
Approve card	8									X								
Display Approved message	9										X							
Get signature	10											X						
Print receipt	11												X					
Transmit order to kitchen	12													X				
Accept cash/coins	13																	
read amount	14															X	X	
Count total amount and display	15													X				X
Reject Cash/Coin	16													X				
Return change	17											X						

Figure: Activity Based DSM for Fast Food Payment System

With a reasonably accurate model of a process, one then uses the model to look for improvements, expecting that they can be implemented in the real process. the primary goal in basic DSM analysis is to minimize feedbacks and their scope by restructuring and re-architecting the process, that is, by re-sequencing the rows and columns of the matrix. This widely practiced initial step in analysis is called partitioning, block diagonalization or block triangularization.

In our model since there are very few feedback paths and amongst the existing ones every path is important hence we cannot minimize the number of feedback paths. But after frequenting DSM we have the following block-diagonalized DSM

Activities		4	5	6	1	7	2	3	8	9	10	15	11	17	13	14	16	12
Dial bank	4		X															
Transmit information	5			X					X									
Decline card	6					X												
Display Bill	1						X											
Display error message	7						X											
Select Mode of payment	2							X							X			
Read Card	3	X																
Approve card	8									X								
Display Approved message	9										X							
Get signature	10												X					
Count total amount and display	15													X	X			
Print receipt	11																	X
Return change	17												X					
Accept cash/coins	13															X		
read amount	14											X					X	
Reject Cash/Coin	16														X			
Transmit order to kitchen	12																	

Figure: Revised Activity Based DSM for Fast Food Payment System

An activity based DSM provides a systematic method for designing a data-driven project schedule such that information transfer is timely and the design more rapidly converges to the desired performance specifications along multiple dimensions.

System Trade-Off Analysis

System Performance Characteristics

1. **Minimizing the Cost of Operating the System (per day/per shift).**
   This directly relates to the salary of the employees and the other variable cost like operating cost of the touch screen and the raw materials
2. **Maximizing the Number of customers (Throughput),**
   Which can be handled by the system in a given shift: This in turn again relates to the number of employees and the number of operating touch screen.
3. **Minimizing the Queue Length**
   This is a direct measure of the customer satisfaction with the system and may cause loss of revenue if a customer decides it is not worth waiting for the system to be free and will leave.
4. **Maximizing the Operator Utilization**
   That is we would not like to pay for the employees, which are sitting idle and cause the cost of operating the system to go up without generating revenue.
5. **Minimizing the Maximum Time spent by the Customer in the System**
   This might be a bottleneck consideration while designing the system. A customer will not come back again if he sees that he has to spend a lot of time get his order completed. Also this is an important characteristic in time critical setting (for e.g. at airport where a customer has to catch a connecting flight and he drops in to get a quick snack or the operation of the fast food restaurant in the business neighborhood where people are always running late for meetings and other important work).
6. **Minimizing the Average Waiting Time**
   This is more or less directly relates to the queue length minimization problem. But it is important to consider the time it takes for an average customer to complete the order on the touch screen. So queue moves quickly. So this in turn relates to designing a user interface, which causes a user to navigate smoothly across the screens without fumbling for the things, he is looking for.
7. **Minimize the Queue Length at the Pickup Counter**
   It is not simply enough to state that minimizing the queue on the touch screens or average waiting time on the touch screens will do the job. We also have to consider employing enough assemblers so that they deliver the prepared order quickly and efficiently.
8. **Minimizing the Number of Lost Customers**
   This directly relates to the requirement on the percentage of customer who will be served by the system. Obviously a lost customer is lost revenue and as this number increases so is the increasing dissatisfaction with the restaurant operation. After all no owner would like to listen to a grapevine that its restaurant fails to meet the demand of that particular area.

Decision Variables

The decision variables are as follows:

1. **Number of Cooks**
   This variable directly relates to the speed of making the food. A large number will ensure that a customer does not have to wait for their order to be prepared. But on the other hand a large number may signify that cooks are sitting idle if the demand is not so big there by increasing the cost of operation.
2. **Number of Assemblers**
   These people are needed to assemble food and delivering them in an efficient fashion. Sometimes these people are also required to deliver instant entities in the order like soda and fries. For a system design without touch screens these are the people, which take the order. A higher number is good from the customer point of view, as they don?t have to wait but is bad from the owner's perspective because it means a higher idle time for slump periods and increasing cost of operation.
3. **Number of Cleaners**
   Obviously as per the federal guidelines a restaurant has to ensure a degree of cleanliness. So this number increases as the number of customer increase to utilize the system.
4. **Number of Touch Screens**
   There is an upfront cost of installing and networking the screens via a central server, which carries all the data. So it is important of the owner to decide upon the number of customer he expects in a particular business setting and employ a most cost effective solution. A higher number means customer has to wait in small queue and their order will be sent to the kitchen promptly. But there is a operating cost and a fixed cost of installing each of the touch screens.

CPLEX Problem Formulation

Identifying the above performance characteristic and the decision variable now we have to conduct a trade off analysis to optimize the system design with respect to all of the performance characteristics. As seen above this is a multi objective optimization problem with competing objectives with respect to the decision variables.

Trade off analysis with CPLEX: In this we will analyze the performance of the system with respect to the decision variables. Specifically we chose Cost of operating the system, number of customers that can be handled and the length of the queue.

As per the given requirement document the following model equations were obtained. See an explanation of the details of the equation below.

Objective Function

- Minimize the cost of operating the system per day.

  ```
  20(X1+X2+X3+X4)+18(Y1+Y2+Y3+Y4)+19(Z1+Z2+Z3+Z4)+16W
  ```

- Maximize the number of customers which can handled by the system (throughput)

  ```
  240(X1+X2+X3+X4)
  ```

- Minimize the length of queue generated at the counter: This in effect becomes maximizing W. We will show it in a moment.

where:

- X1: Number of cooks working in the breakfast shift
- X2: Number of cooks working in the lunch shift
- X3: Number of cooks working in the snacks shift
- X4: Number of cooks working in the dinner shift
- Y1: Number of cleaners working in the breakfast shift
- Y2: Number of cleaners working in the lunch shift
- Y3: Number of cleaners working in the snacks shift
- Y4: Number of cleaners working in the dinner shift
- Z1: Number of assemblers working in the breakfast shift
- Z2: Number of assemblers working in the lunch shift
- Z3: Number of assemblers working in the snacks shift
- Z4: Number of assemblers working in the dinner shift
- W: Number of touch screens installed in the system

Equation for the operating cost was obtained as follows:

As per the requirement document cooks, cleaners and the assemblers are paid a salary of $5, $4.5 and $4.75 per hour. We have 4 hours per shift and there are four shifts. So salary expenses for a typical day becomes:

```
Salary = 4*5(X1+X2+X3+X4)+4*4.5(Y1+Y2+Y3+Y4)+4*4.75(Z1+Z2+Z3+Z4)
       = 20 (X1+X2+X3+X4) + 18 (Y1+Y2+Y3+Y4)+ 19 (Z1+Z2+Z3+Z4)
```

Operating cost of a touch screen was determined to be $1 per hour based on its useful life and the cost of buying it. So for 16 hours operation of w screens we have an operating cost of 16W per day added to the above equation resulting in the complete equation of the cost.

Equation for the number of customers that can be handled by the system per day was obtained as follows:

Bottleneck process for the system is preparing food, which is carried out by the employed cooks in a particular shift. It was estimated that for an average order a cook takes about 1 minute to prepare the main dish. So for a shift one cook can handle about 240 customers based on the 4-hour duration of the shift. Again to keep the problem simple small breaks taken by the employees during the work were neglected. We are designing the system to operate at its peak capacity and then see its performance. Based on this data average throughput of the system per day is given by

```
240*( X1 + X2 + X3 + X4)
```

Any demand over this value will cause long lines at the pickup counter and possibly loss of the customers.

As noted above in the objective formulation we now explain how minimizing the length of queue relates to the maximizing number of touch screen installed. Though it is very intuitive.

Customer arrival and customer service follow poison and exponential distribution respectively. The queue under consideration is called a multi-channel single stage queue as shown below:

That is, customers arrive according to a poison process with a mean rate of arrival into the system and join a single queue. The person at the front of the queue seizes the first available resource (touch screen) to place his order. The process of placing the order is exponential with a mean rate of about 2 minutes. This is symbolized as a M/M/W type of multi-channel single stage queue. Length of this queue is calculated as follows:

Customers arrive according to a Poisson process with rate l

There are s servers.

There is a single line of entities awaiting service

Service time is exponential with rate m

Let r = l/sm

We have used an Excel template to calculate Lq

Result of the CPLEX Runs

We have used the constraint method to solve this multi-objective optimization problem. i.e., we have minimized the cost taking the throughput and the number of the screens as the constraints. Following matrix shows the result of the cplex run (A log of the cplex file containing the runs can be seen in Appendix 1. LP formulation file used to run the cplex optimization can be seen in Appendix 2).

No.	Design Variables													Cost	Meal Throughput					Queue Length				
	W	X1	X2	X3	X4	Y1	Y2	Y3	Y4	Z1	Z2	Z3	Z4		Breakfast	Lunch	Snacks	Dinner	Total	Bf	Lun	Nn	Din	Avg
1	7	1	3	1	2	1	1	1	1	1	2	1	1	419	200	500	200	300	1200	0.001	0.235	0.001	0.009	0.0615
2	7	1	3	1	2	1	1	1	1	1	2	1	1	419	240	720	240	480	1680	0.002	3.683	0.002	0.189	0.969
3	7	1	3	2	2	1	1	1	1	1	2	1	1	439	200	600	250	350	1400	0.001	0.81	0.003	0.024	0.2095
4	7	2	3	2	2	1	1	1	1	1	2	1	1	459	250	650	250	450	1600	0.003	1.48	0.003	0.119	0.40125
5	7	2	4	2	3	1	1	1	1	1	2	1	2	518	250	750	300	500	1800	0.003	5.847	0.009	0.235	1.5235
6	7	2	4	2	4	1	1	1	1	1	2	1	2	538	300	800	350	750	2200	0.009	17.223	0.024	5.847	5.77575
7	8	1	3	1	2	1	1	1	1	1	2	1	1	435	200	500	200	300	1200	0	0.078	0	0.002	0.02
8	8	1	3	1	2	1	1	1	1	1	2	1	1	435	240	720	240	480	1680	0	1.071	0	0.059	0.2825
9	8	1	3	2	2	1	1	1	1	1	2	1	1	455	200	600	250	350	1400	0	0.279	0.001	0.006	0.0715
10	8	2	3	2	2	1	1	1	1	1	2	1	1	475	250	650	250	450	1600	0.001	0.494	0.001	0.038	0.1335
11	8	2	4	2	3	1	1	1	1	1	2	1	2	534	250	750	300	500	1800	0.001	1.494	0.002	0.078	0.39375
12	8	2	4	2	4	1	1	1	1	1	2	1	2	554	300	800	350	750	2200	0.002	2.663	0.006	1.494	1.04125
13	9	1	3	1	2	1	1	1	1	1	2	1	1	451	200	500	200	300	1200	0	0.026	0	0	0.0065
14	9	1	3	1	2	1	1	1	1	1	2	1	1	451	240	720	240	480	1680	0	0.39	0	0.019	0.10225
15	9	1	3	2	2	1	1	1	1	1	2	1	1	471	200	600	250	350	1400	0	0.101	0	0.002	0.02575
16	9	2	3	2	2	1	1	1	1	1	2	1	1	491	250	650	250	450	1600	0	0.182	0	0.012	0.0485
17	9	2	4	2	3	1	1	1	1	1	2	1	2	550	250	750	300	500	1800	0	0.536	0	0.026	0.1405
18	9	2	4	2	4	1	1	1	1	1	2	1	2	570	300	800	350	750	2200	0	0.895	0.002	0.536	0.35825

Where

- Throughput = Breakfast + Lunch + Snacks + Dinner
- Lenbf: Average length of queue for breakfast
- Lenlun: Average length of queue for lunch
- Lensn: Average length of queue for snacks
- Lendin: Average length of queue for dinner
- Average: (Lenbf + Lenlun + Lensn + Lendin)/4.0

Plot of Non Inferior Solution in the Objective Space

Trade-Off between Throughput and Cost

Trade-Off between Cost and Length of Queue

Trade-Off between Throughput and Length of Queue

Analysis of Pareto Points of the System

From the above matrix and the graphs plotted taken two objective at a time following pareto points were obtained (shown in the above graphs with a circle around it):

No.	Cost	Throughput	Length of Queue	W	X1	X2	X3	X4	Y1	Y2	Y3	Y4	Z1	Z2	Z3	Z4
1	419	1200	0.0615	7	1	3	1	2	1	1	1	1	1	2	1	1
2	419	1680	0.969	7	1	3	1	2	1	1	1	1	1	2	1	1
7	435	1200	0.02	8	1	3	1	2	1	1	1	1	1	2	1	1
8	435	1680	0.2825	8	1	3	1	2	1	1	1	1	1	2	1	1
13	451	1200	0.0065	9	1	3	1	2	1	1	1	1	1	2	1	1
14	451	1680	0.1022	9	1	3	1	2	1	1	1	1	1	2	1	1

A Pareto point correspond to a point in the objective space which belongs to a set of non-inferior solutions meaning if you move away from this point to any other point in the objective space you will have to sacrifice at least one objective.

Final System Design

From the owners perspective point 2 seems to be a best compromise among all the available system design because it has a low cost of operation per day and can handle a high throughput at a reasonable queue length, which is acceptable by the requirement document.

No.	Cost	Throughput	Length of Queue	W	X1	X2	X3	X4	Y1	Y2	Y3	Y4	Z1	Z2	Z3	Z4
2	419	1680	0.969	7	1	3	1	2	1	1	1	1	1	2	1	1

System-Component Testing

Primary Verification Plan

Testing and product delivery procedures begin at the component level and work toward the system and stakeholder tests. Primary verification and validation plan for the above system in proposed as follows:

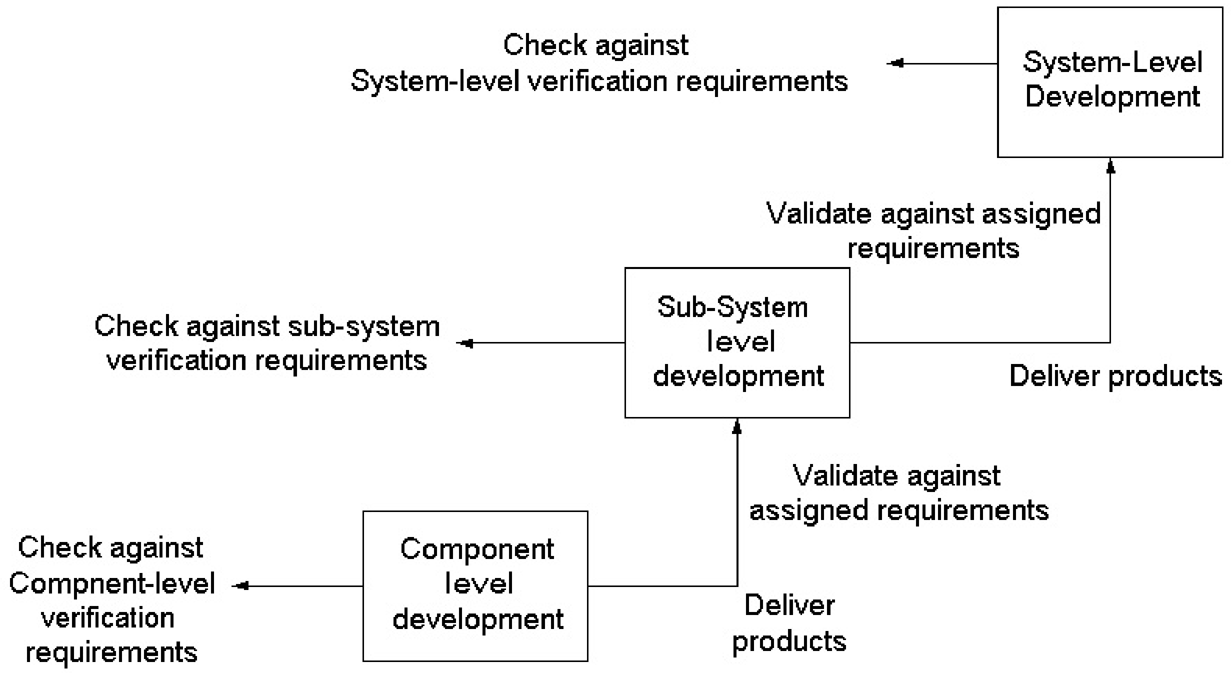

Check against
System-level verification requirements
System-Level Development
Validate against assigned requirements
Check against sub-system verification requirements
Sub-System level development
Deliver products
Validate against assigned requirements
Check against Compnent-level verification requirements
Component level development
Deliver products

Figure 20. Detailed Model of Product Testing and Delivery.

Testing and product delivery procedures include:

- Verification of the products against "specified requirements" and;
- Validation of the assembled (sub)system against assigned requirements.

Below are some examples to demonstrate how we can test our requirments at different levels:

Requirement 1.1. Restaurant will be open for 16 hours and will operate in four shifts as Morning (730 AM -1130 AM), Afternoon (1130 AM -330 PM), Evening (330 PM - 730 PM) and Night (730 PM -1130 PM).

Examination.1.1.1 Inspect the restaurant operation where this proposed system will be installed (As per the requirement supplied by the customer).

- System will fail if the operating time is changed once the system in installed (might reflect in high operating cost or low efficiency and utilization of system resources making the operation non profitable).

Requirement 1.2. System should be able to serve a throughput of 50, 125, 50, 75 customers per hour during these four shifts respectively.

Simulation.1.2.1. Run a system wide simulation (as done in arena for example)

- System fails if the maximum number in line (bottleneck) exceeds 10 or if there are lost customers.

Examination.1.2.1 After the prototype is installed in one of the restaurant observe the total throughput and the maximum number in queue.

Requirement 1.3. Cooks, cleaners and assemblers will be the type of employees working the restaurant.

Examination.1.3.1. See the type of workers working in the restaurant where we plan to install such a system.

- System fails if a new level of hierarchy is introduced by the management at a later point of time, which causes the operating cost per day to be changed.

Requirement 1.4. Cook will be paid at the rate of $5 per hour for the duration of their work.

Examination.1.4.1. Find out the rate of the pay of the cook for the restaurant where we plan to install the system.

Simulation.1.4.1. See if the direct labor cost corresponding to cook resource and divide it by number of hours worked. This number should be $5/hr.

- System fails otherwise.

...... details of verification plan removed ....

Requirement 3.4.2 Once connected to the bank system will supply the card information to the bank database, will query the card validity and will supply the amount to be charged to the card.

Demo.3.4.2.1 After the modem connects to the bank touch screen display changes to (transmitting, authorizing, approved) in succession when user is paying with a credit card with remaining credit on it.

- System fails if above sequence does not take place even though user has swiped a card with a remaining credit on it

Simulation.3.4.2.1. When executing the scenario of paying by the credit card, software subsystem supplies the card information, total bill amount to be charged to the credit card, query its validity and debit the amount.

- System fails if such functionality does not exist while paying with a credit card.

Requirement 3.5.1 System will be secured to grant access rights only to the system administrator. For this a login ID and a password will be assigned which could be changed. Password won't be visible to onlookers while typing for increased security.

Demo.3.5.1.1: Enter into system, administrator menu and change password

(Assumption – password known to tester)

- System fails if it does note provide system admin a way to change password

Examination.3.5.1.1: A bystander tries to observe password while it is being typed.

Expected output: He cant see it because instead of actual letters X's are seen on the screen every time a key is pressed

- System fails if bystander can see the password while it is being typed because instead of echoing X's actual letters are echoed on screen.

Simulation.3.5.1.1: FFBD provides a branch of activity wherein password can be changed.

- System fails if such a provision does not exist in the behavior diagram of the system.

Test.3.5.1.1: Try to read the file where this login / password is stored in a text editor such as notepad by connecting to a windows operating system.

Expected output: Text editor splits some garbage value when that file is opened.

- System fails if the text file stores in a neat manner the login and password.

Requirement 3.5.2. System will deny access to change the contents if the login / password / both are incorrect by providing an error message.

Demo.3.5.2.1.: In the system administrator menu type the correct login but invalid password

Expected output: Error message displayed showing either login / password is incorrect.

- System fails if it either grants access with invalid entry or displays and exact error message that password is wrong (so that the intruder gets to know that he is typing the correct login name but the invalid password)

Demo.3.5.2.2.: In the system administrator menu type the incorrect login but correct password.

Expected output: Error message displayed showing either login / password is incorrect.

- System fails if it either grants access with invalid entry or displays and exact error message that login is wrong (so that the intruder gets to know that he is typing the incorrect login name but a valid password).

Demo.3.5.2.3: In the system administrator menu type the incorrect login and incorrect password

Expected output: Error message displayed showing either login / password is incorrect.

- System fails if it either grants access with invalid entry or displays and exact error message that both login and password are wrong (so that the intruder gets to know that he is typing the incorrect login name and password).

Simulation.3.5.2.1: System administrator menu simulation passes through a checkpoint where login / password are matched against the one specified in the file.

- System fails if this checkpoint is bypassed in the simulation.

Verification Traceability Matrix

The abbreviated verification traceability matrix is:

Design Requirement	**Verification Method**				**Verification Requirement**	**Level of Application**
	Test	**Analysis**	**Demo**	**Exam**		
1.1.				1.1.1.	Examination.1.1.1	**1.0 System Level Requirements**
1.2.		1.2.1		1.2.1	Simulation.1.2.1, Examination.1.2.1	**1.0 System Level Requirements**
....	....				...	...
3.4.2		3.4.2.1	3.4.2.1		Demo.3.4.2.1 Simulation.3.4.2.1	3.4 Modem-Component Level Requirements
3.5.1	3.5.1.1	3.5.1.1	3.5.1.1	3.5.1.1	Demo.3.5.1.1	3.5 Software-

					Simulation.3.5.1.1 Examination.3.5.1.1 Test.3.5.1.1	Component Level Requirements
3.5.2		3.5.2.1	3.5.2.1 -3.5.2.3		Demo.3.5.2.1 Demo.3.5.2.2 Demo.3.5.2.3 Simulation.3.5.2.1	3.4 Software-Component Level Requirements

VSN's

After specifying the primary verification plan the next stage is to group some of these verification tasks to form a verification string called VSN. These VSN?s are a way to save time and money with respect to the validation and verification process, which are a critical performance measure of system design.

VSN 1. (Touch Screen - 3.1 Component Level Requirements)(This VSN corresponds to tests done in a lab on the touch screen component of the hardware subsystem) This VSN was chosen in such a manner because it facilitates accelerated testing in a lab and saves time and money by conducting the experiments sequentially.

1. Test.3.1.3.1 (Check the number of color screen can support)
2. Test.3.1.4.1 (Check the highest resolution of the screen)
3. Test.3.1.5.1 (Measure the size of the touch screen by a scale)
4. Test.3.1.6.1 (Check the maximum amount of text that can be displayed)
5. Test.3.5.1.1 (Read the password file in a text editor)
6. Test.3.1.2.1 (Strike the glass of the screen with x+5 lb of force for 200 times)
7. Test.3.1.2.2 (Strike the glass of the screen with nail like object for 200 times)

VSN-2. (Ordering System - 2.1 Sub-system Level Requirement) This VSN corresponds to simulating the scenario wherein a customer places an order and decides to pay cash for his order). This VSN tests the hardware and the software subsystem of the automated food ordering system by testing interaction between them by executing the tests in the following sequences.

1. Simulation.2.1.3.1 (Account for language selection)
2. Execute the scenario of placing an order by adding a combo and proceed to pay bill.
3. Simulation.3.3.2.1 (Make a choice for mode of payment)
4. Simulation.2.1.3.1 (Account for the language selection)
5. Simulation.2.2.2.1 (Execute the payment by cash choice)
6. Simulation.2.1.3.1 (Account for language selection)
7. Simulation.2.2.5.1 (Execute the reject bill scenario)
8. Execute the acceptance of subsequent cash entered to the bill amount.
9. Simulation.2.2.4.1 (Execute change of display on the touch screen)
10. Simulation.2.1.3.1 (Account for language selection)
11. Simulation.2.2.1.1 (Execute printing of receipt with order number on it)
12. Simulation.3.3.3.1 (Execute Transfer order to kitchen)
13. Simulation.2.1.1.1 (Execute Display on the kitchen screen)
14. Simulation.3.3.4.1 (Add the new order last in the list of existing order)

VSN-3. (This VSN simulates the kitchen screen component and its interaction with the software subsystem. This shows a component level verification by executing the scenario where an assembler signals a completed order).

1. Simulation.3.3.5.1 (Execute removing of item from kitchen screen when touched and fill its display with next order)

VSN-4. (This VSN simulates the system administration component of the software subsystem by executing the scenario of entering a system administrator menu and changing the menu)

1. Simulation.3.5.2.1 (Execute matching of input login / password with that supplied in the file)
2. Simulation.3.5.1.1 (Execute change of password of system administrator)

VSN-5. (1.0 System Level Requirement) (This is a system wide simulation, which notes the interaction of the staff, hardware and software subsystem. In this simulation we can show that component level verification does not essentially mean the system level verification. Different system designs options for the tradeoff analysis were obtained by this VSN.)

1. Simulation.1.2.1 (In a system wide simulation note the number of customer served designated as throughput and maximum people waiting in the line.)
2. Simulation.1.4.1 (Divide the total cost of cook resource by the total number of hours worked to see if it comes out to be $5/hr.)

3. Simulation.1.5.1 (Divide the total cost of assembler resource by the total number of hours worked to see if it comes out to be $4.75/hr.)
4. Simulation.1.6.1 (Divide the total cost of cook resource by the total number of hours worked to see if it comes out to be $4.5/hr.)

VSN-6. (Card Acceptor/Reader and Modem - 3.2 and 3.4 Component Level Requirements) This VSN simulates the interaction between software and hardware subsystem by seeing the interaction between card processor, card reader and the modem and its effective interpretation by the software by changing the display on the touch screen display).

1. Execute place order scenario by adding some item.
2. Execute pay bill.
3. Execute the credit card payment scenario.
4. Simulation.3.5.3.1 (Initialize the modem to dial the bank and supply the information)
5. Simulation.3.4.2.1 (Transmit, authorize and approval of transaction)
6. Simulation.3.4.1.1 (See the result of maximum time taken to complete simulation.3.4.3.1 and simulation.3.4.1.1.)

VSN-7. (Payment System - 2.2. Sub-system Level Requirements) (This VSN demonstrates the comprehensive testing of the software and hardware subsystem of the automated ordering process. In particular it stresses on the graphical user interface display on the touch screen and the cash payment/cash return system and their interfacing with the software which causes the display to be changed. We test a particular scenario in which a user places an order of about $17 (assumption) and decides to pay cash by giving some unusual inputs as specified in the primary verification plan. This sequence was chosen because it involves only one prototype and one user and can be done sequentially.)

1. Demo.2.1.2.1 (Check for language selection)
2. Examination.3.3.7.1 (Contrast requirement between foreground and background)
3. Examination.2.1.2.1 (Can you place order by item name or number)
4. Test.3.3.8.1 (Measure the smallest font of the screen)
5. Demo.3.1.1.1 (Touching the screen lightly causing no action to happen)
6. Demo.3.1.1.2 (Apply normal pressure on the touch screen for an input)
7. Demo.2.1.2.1 (Selects an item on the screen by touching it)
8. Demo.2.1.3.1 (Check if language selection made in beginning is accounted)
9. Test.3.3.8.1 (Measure smallest font size on the screen by a scale)
10. Demo.2.2.2.5 (Click on the pay bill button to see if you have an option to pay with cash)
11. Demo.2.2.2.2 (Insert a fake bill to see if the cash acceptor rejects it)
12. Demo.2.2.2.3 (Insert a currency from other country to see if the cash acceptor rejects it)
13. Demo.2.2.2.4 (Insert a coin from other country in the cash slot to see if the cash acceptor rejects it)
14. Demo.2.1.3.1 (Inspect if the language selection is being adhered to)
15. Test.3.3.8.1 (Measure the smallest font size on the screen)
16. Demo.2.2.2.1 (Finally enter a correct bill. See if touch screen display changes)
17. Demo.2.2.4.1 (Enter a $1 bill face up. See if touch screen display changes)
18. Demo.2.2.4.2 (Enter a $1 bill face down. See if touch screen display changes)
19. Demo.2.2.4.1 (Enter a $5 bill face up. See if touch screen display changes)
20. Demo.2.2.4.2 (Enter a $5 bill face down. See if touch screen display changes)
21. Demo.2.2.5.1 (Enter a penny with face up. See if cash acceptor rejects it)
22. Demo.2.2.5.2 (Enter a penny with face down. See if cash acceptor rejects it)
23. Demo.2.2.5.3 (Enter a $50 bill with face up to see if cash acceptor rejects it)
24. Demo.2.2.5.4 (Enter a $50 bill with face down to see if cash acceptor rejects it)
25. Demo.2.2.5.5 (Enter a $100 bill with face up to see if the cash acceptor rejects it)
26. Demo.2.2.5.6 (Enter a $100 bill with face down to see if the cash acceptor rejects it)
27. Demo.2.2.6.1 (Try to enter a $1 coin)
28. Examination.2.2.6.1 (Inspect to find that the slot is smaller) Enter a quarter
29. Demo.2.2.7.2 (Enter a $10 bill. Observe the message that the change required to be given exceeds $5. Please try again and all cash comes out)
30. Demo.2.2.7.1 (Enter a $20 bill. Count change refunded)
31. Examination.2.2.1.1 (Check the receipt printed to see if the order number is printed on it)

VSN-8. (Ordering System - 2.1 Sub-system Level Requirement) (This VSN demonstrates the comprehensive testing of the software and hardware subsystem of the automated ordering process. This VSN is executed concurrently with VSN-7 to test the FCFS queuing requirement to be satisfied by the ordering subsystem. This VSN tests the scenario where a user has placed a combo order and then is trying to make the payment using his credit card. This VSN comprehensively tests the payment system by swiping fake, expire, over the limit card or the cards with invalid pin etc to see if software subsystem works in harmony with bank database and can generate appropriate error message. Apart from the payment system GUI and display requirements are also tested.)

1. Demo.2.1.3.1 (Make Spanish language selection to start)
2. Demo.3.1.1.5 (Take a combo food option by almost hitting the screen with the finger)
3. Demo.2.1.2.2 (Select the combo option. This has to be performed simultaneously with Demo.9.5 as mentioned above)
4. Demo.3.3.2.1 (Take the pay bill option and see if system provides a screen to choose the mode of the payment)
5. Examination.3.2.1.1 (Inspect if system has a card reader where you can swipe you credit / debit card)
6. Examination.3.2.3.1 (Inspect the type of the slot in the card reader)
7. Demo.2.2.3.1 (Try to swipe a store card or university ID with a magnetic strip at the back to observe if the system rejects it)

8. Demo.2.2.3.2 (Swipe a valid debit card. See response)
9. Demo.2.1.3.1 (Throughout Demo.5.1 and Demo.5.2 to check if the language selection made in the beginning is adhered to or not)
10. Test.3.3.8.1 (Measure the smallest font size on the screen)
11. Demo.3.3.6.3 (Type invalid pin when asked for it after swiping debit card)
12. Demo.3.3.6.1 (Swipe a credit card with balance past its credit limit and see the response)
13. Demo.3.3.6.2 (Swipe an expired credit card and see the response)
14. Demo.2.2.3.3 (Finally swipe a credit card with remaining credit balance on it and see the response)
15. Demo.3.2.2.1 (To be done simultaneously with Demo.5.3. Swipe the credit card very slowly)
16. Demo.3.2.2.2 (Swipe the card at normal speed this time)
17. Test.3.4.1.1 (Measure with a stop watch the maximum time it takes to complete the transaction)
18. Demo.3.5.3.1 (See the changing display on the touch screen in accordance with the language selection made in the beginning as the modem is initialized and talks with the bank to approve the transaction)
19. Demo.3.4.2.1 (Continue noting the display on the touch screen after the modem is connected to bank and bank now responds with the result of the requested transaction)
20. Examination.2.2.1.1 (Check if the order number is printed on the receipt)
21. Examination.3.3.3.1 (Signal the person standing on the kitchen touch screen once placing of order and payment is completed)
22. Examination.3.3.4.1 (Display on the kitchen screen changes and this new order is added last in the existing queue)
23. Demo.3.3.5.1 (Person standing at the kitchen touch screen to signal to the software sub system that the order has been delivered)
24. Demo.2.1.1.1 (Since VSN-8 takes less time to complete than VSN-7 which are taking place simultaneously on two different prototypes)

VSN-9. (This VSN test the system administrator component of the software subsystem and ascertains it correct working)

1. Demo.3.5.2.1 (In Sys admin menu type correct login / incorrect password)
2. Demo.3.5.2.2 (In Sys admin menu type incorrect login / correct password)
3. Demo.3.5.2.3 (In Sys admin menu type incorrect login / incorrect password)
4. Demo.3.5.1.1 (In Sys admin menu type correct login / password then change it)
5. Examination.3.5.1.1 (A person standing nearby sees X's while password is being typed on the screen)
6. Examination.2.1.6.1 (Inspect the keyboard layout)

VSN-10. (1.0 System Level Requirements) (This is a system wide testing when the prototype in put in the operational context. This testing reveals the inherent flaws in the system in terms of output performance measures like throughput and length of queue generated at the touch screens on which the trade off of the system is performed among the available choices in the preceding section.)

1. Examination.1.1.1 (Inspect the time of operation)
2. Examination.1.2.1 (Inspect the maximum throughput)
3. Examination.1.3.1 (Inspect the type of employees working)
4. Examination.1.4.1 (Inspect the salary paid to cook)
5. Examination.1.5.1 (Inspect the salary paid to assembler)
6. Examination.1.6.1 (Inspect the salary paid to cleaner)
7. Examination.1.7.1 (Inspect the length of queue which causes a customer to leave)

Coverage and Completeness

Now since the VSN?s are completed we move to verify the higher-level systems requirement, which were supplied by the end user to check them against the VSN?s and primary verification plan.

1. System supports native language of the country and other commonly spoken languages. (Demo 2.1.3.1, Simulation 2.1.3.1)
2. User should be able to place order according to his choices. (Demo 2.1.2.1,2.1.2.2, Examination 2.1.2.1)
3. User should be able to make payment using cash / credit / debit card. (Demo 2.2.2.1-2.2.2.5, 2.2.3.1-2.2.3.3, 2.2.4.1-2.2.4.2, 3.3.2.1, Simulation 2.1.2.5,2.2.4.1,3.3.2.1)
4. User should get a receipt and a token number after making the payment. (Simulation 2.2.1.1, Examination 2.2.1.1)
5. The system should be able to take any type of inputs, once he touches the respective button. (Demo 2.1.2.1,2.1.2.2, Examination 2.1.2.1)
6. The system should be able to calculate the bill and prompt the user for the mode of payment and generate a receipt. (Demo 3.3.2.1, Simulation 2.2.1.1,3.3.2.1, Examination 2.2.1.1)
7. The system should be able to pass on the order in the kitchen for processing. (Simulation 2.1.1.1,2.2.1.1,3.3.4.1, Demo 2.1.1.1, Examination 3.3.3.1,3.3.4.1)
8. The system should be secured to restrict the number of people to enter the system to make changes in the menu and its items. (Demo 3.5.1.1,3.5.2.1,3.5.2.2,3.5.2.5, Examination 3.5.1.1, Simulation 3.5.1.1,3.5.2.1, Test 3.5.1.1)
9. The system should be sturdy for rough usage. (Test 3.1.2.1,3.1.2.2, Demo 3.1.1.5)
10. System has a cash return mechanism which gives refund up to 5 $ in coins. (Demo 2.2.7.1,2.2.7.2)
11. System should be able to communicate to the central database to verify the authenticity of the credit/debit card. (Demo 3.4.2.1, Simulation 3.4.2.1)
12. System should allow Store manager to add/delete/alter system items. (Demo 3.5.1.1, Examination 3.5.1.1, Simulation 3.5.1.1)
13. The system must be a graphical user interface for easy use and understanding. (Examination 3.3.1.1)
14. The system must be able to prompt the user for the next step to be performed during the process of using the system. (Demo 3.3.6.1-3.3.6.4)

Hence we see that tracing the design back to the higher-level requirements supplied by the user they are tested by the given verification plan. Hence, we ascertain that the system is completely covered with respect to the verification and validation.

We have also shown the verification of subsystem and the system level while specifying the verification string. So while VSN 1, 3, 6, 4&9 correspond to component level verification of touch screen, kitchen screen, Card Acceptor/Reader and modem other VSN's correspond to either sub system level verification and validation. VSN 2, 7 and 8 either simulate or demonstrate the subsystem level verification of software and hardware subsystem. Again VSN's 5 and 10 simulate and test system level verification criteria. So we can say with confidence that our verification plan is complete and addresses the issues of bottom up and top down verification. So if the system can be passed by the above verification plan it will satisfy the user requirement and all components will act in harmony with each other.

References and Web Resources

1. Excel template to calculate the length of queue of MMS system http://www.courses.vcu.edu/MATH327/Shells/Mms.xls
2. Multi-Objective Optimization
   http://www.isr.umd.edu/Courses/BARAS-ENSE623/secured/Class%20Handouts/Trade-Off-1.pdf
3. Queuing Systems: Math528 Stochastic Operation Research
   http://www.courses.vcu.edu/MATH-jrm/MATH528/Slides/528QueuingAnalysis.ppt
4. CS 6751 Group Project
   http://www.cc.gatech.edu/computing/classes/cs6751_94_summer/davidz/project.html
5. Creating and running a simple model simulation using Arena
   See http://www.acsu.buffalo.edu/~thill/IE477/StartARENA.htm
6. CPLEX Online Manual http://www.dmi.usherb.ca/laboratoires/documentations-logiciels/cplex75/cplex75/doc/homepage/manuals.html
7. Fast Food ATM http://www.halfbakery.com/idea/Fast_20Food_20atm

Appendices

LP file for CPLEX run

```
 Minimize

   20 X1 + 20 X2 + 20 X3 + 20 X4 + 18 Y1 + 18 Y2 + 18 Y3 +
   18 Y4 + 19 Z1 + 19 Z2 + 19 Z3 + 19 Z4 + 16 W
st
   240(X1 + X2 + X3 + X4) >= 1680

   240 X1 >= 240
   240 X2 >= 720
   240 X3 >= 240
   240 X4 >= 420
   2 Z1 - X1 >= 0
   2 Z2 - X2 >= 0
   2 Z3 - X3 >= 0
   2 Z4 - X4 >= 0

bounds

   X1 >= 1
   X2 >= 1
   X3 >= 1
   X4 >= 1
   Y1 >= 1
   Y2 >= 1
   Y3 >= 1
   Y4 >= 1
   Z1 >= 1
   Z2 >= 1
   Z3 >= 1
   Z4 >= 1
   W = 7

generals

   X1 X2 X3 X4 Y1 Y2 Y3 Y4 Z1 Z2 Z3 Z4 W

end
```

Log file of the CPLEX runs

```
 Log started (V7.1.0) Fri Dec 13 15:03:20 2002

 Problem 'work.lp' read.
 Read time = 0.01 sec.

 C1:  240 X1 + 240 X2 + 240 X3 + 240 X4 >= 1200
 C2:  240 X1 >= 200
```

```
C3:  240 X2 >= 500
C4:  240 X3 >= 200
C5:  240 X4 >= 300
C6:  W  = 7

Tried aggregator 1 time.

MIP Presolve eliminated 10 rows and 13 columns.
MIP Presolve modified 6 coefficients.
All rows and columns eliminated.

Presolve time = 0.00 sec.

Integer optimal solution:  Objective = 4.1900000000e+02

Solution time = 0.00 sec.  Iterations = 0 Nodes = 0

Variable Name           Solution Value

X1                            1.000000
X2                            3.000000
X3                            1.000000
X4                            2.000000
Y1                            1.000000
Y2                            1.000000
Y3                            1.000000
Y4                            1.000000
Z1                            1.000000
Z2                            2.000000
Z3                            1.000000
Z4                            1.000000
W                             7.000000

Log started (V7.1.0) Fri Dec 13 15:05:07 2002

Problem 'work.lp' read.

Read time = 0.01 sec.

C1:  240 X1 + 240 X2 + 240 X3 + 240 X4 >= 1680
C2:  240 X1 >= 240
C3:  240 X2 >= 720
C4:  240 X3 >= 240
C5:  240 X4 >= 480
C6:  W  = 7

Tried aggregator 1 time.

MIP Presolve eliminated 10 rows and 13 columns.
MIP Presolve modified 2 coefficients.
All rows and columns eliminated.

Presolve time = 0.00 sec.

Integer optimal solution:  Objective = 4.1900000000e+02
Solution time = 0.00 sec.  Iterations = 0 Nodes = 0

Variable Name           Solution Value

X1                            1.000000
X2                            3.000000
X3                            1.000000
X4                            2.000000
Y1                            1.000000
Y2                            1.000000
Y3                            1.000000
Y4                            1.000000
Z1                            1.000000
Z2                            2.000000
Z3                            1.000000
Z4                            1.000000
W                             7.000000

Problem 'work.lp' read.

Read time = 0.00 sec.

C1:  240 X1 + 240 X2 + 240 X3 + 240 X4 >= 1400
C2:  240 X1 >= 200
C3:  240 X2 >= 600
C4:  240 X3 >= 250
C5:  240 X4 >= 350
C6:  W  = 7
```

```
Tried aggregator 1 time.

MIP Presolve eliminated 10 rows and 13 columns.
MIP Presolve modified 7 coefficients.

All rows and columns eliminated.
Presolve time = 0.00 sec.

Integer optimal solution:  Objective = 4.3900000000e+02
Solution time = 0.00 sec.  Iterations = 0 Nodes = 0

Variable Name           Solution Value

X1                            1.000000
X2                            3.000000
X3                            2.000000
X4                            2.000000
Y1                            1.000000
Y2                            1.000000
Y3                            1.000000
Y4                            1.000000
Z1                            1.000000
Z2                            2.000000
Z3                            1.000000
Z4                            1.000000
W                             7.000000

Problem 'work.lp' read.

Read time = 0.01 sec.

C1:  240 X1 + 240 X2 + 240 X3 + 240 X4 >= 1600
C2:  240 X1 >= 250
C3:  240 X2 >= 650
C4:  240 X3 >= 250
C5:  240 X4 >= 450
C6:  W  = 7

Tried aggregator 1 time.

MIP Presolve eliminated 10 rows and 13 columns.
MIP Presolve modified 7 coefficients.

All rows and columns eliminated.

Presolve time = 0.00 sec.

Integer optimal solution:  Objective = 4.5900000000e+02
Solution time = 0.01 sec.  Iterations = 0 Nodes = 0

Variable Name           Solution Value

X1                            2.000000
X2                            3.000000
X3                            2.000000
X4                            2.000000
Y1                            1.000000
Y2                            1.000000
Y3                            1.000000
Y4                            1.000000
Z1                            1.000000
Z2                            2.000000
Z3                            1.000000
Z4                            1.000000
W                             7.000000

Problem 'work.lp' read.

Read time = 0.01 sec.

C1:  240 X1 + 240 X2 + 240 X3 + 240 X4 >= 1800
C2:  240 X1 >= 250
C3:  240 X2 >= 750
C4:  240 X3 >= 300
C5:  240 X4 >= 500
C6:  W  = 7

Tried aggregator 1 time.

MIP Presolve eliminated 10 rows and 13 columns.
MIP Presolve modified 7 coefficients.
```

```
All rows and columns eliminated.

Presolve time = 0.00 sec.

Integer optimal solution:  Objective = 5.1800000000e+02

Solution time = 0.00 sec.  Iterations = 0 Nodes = 0

Variable Name           Solution Value

X1                            2.000000
X2                            4.000000
X3                            2.000000
X4                            3.000000
Y1                            1.000000
Y2                            1.000000
Y3                            1.000000
Y4                            1.000000
Z1                            1.000000
Z2                            2.000000
Z3                            1.000000
Z4                            2.000000
W                             7.000000

Problem 'work.lp' read.

Read time = 0.01 sec.

C1:  240 X1 + 240 X2 + 240 X3 + 240 X4 >= 2200
C2:  240 X1 >= 300
C3:  240 X2 >= 800
C4:  240 X3 >= 350
C5:  240 X4 >= 750
C6:  W  = 7

Tried aggregator 1 time.

MIP Presolve eliminated 10 rows and 13 columns.
MIP Presolve modified 5 coefficients.

All rows and columns eliminated.

Presolve time = 0.00 sec.

Integer optimal solution:  Objective = 5.3800000000e+02

Solution time = 0.00 sec.  Iterations = 0 Nodes = 0

Variable Name           Solution Value

X1                            2.000000
X2                            4.000000
X3                            2.000000
X4                            4.000000
Y1                            1.000000
Y2                            1.000000
Y3                            1.000000
Y4                            1.000000
Z1                            1.000000
Z2                            2.000000
Z3                            1.000000
Z4                            2.000000
W                             7.000000

Problem 'work.lp' read.

Read time = 0.02 sec.

C1:  240 X1 + 240 X2 + 240 X3 + 240 X4 >= 1200
C2:  240 X1 >= 200
C3:  240 X2 >= 500
C4:  240 X3 >= 200
C5:  240 X4 >= 300
C6:  W  = 8

Tried aggregator 1 time.

MIP Presolve eliminated 10 rows and 13 columns.
MIP Presolve modified 6 coefficients.

All rows and columns eliminated.
Presolve time = 0.00 sec.
```

```
Integer optimal solution:  Objective = 4.3500000000e+02
Solution time = 0.00 sec.  Iterations = 0 Nodes = 0

Variable Name           Solution Value

X1                            1.000000
X2                            3.000000
X3                            1.000000
X4                            2.000000
Y1                            1.000000
Y2                            1.000000
Y3                            1.000000
Y4                            1.000000
Z1                            1.000000
Z2                            2.000000
Z3                            1.000000
Z4                            1.000000
W                             8.000000

Problem 'work.lp' read.

Read time = 0.00 sec.

C1:  240 X1 + 240 X2 + 240 X3 + 240 X4 >= 1680
C2:  240 X1 >= 240
C3:  240 X2 >= 720
C4:  240 X3 >= 240
C5:  240 X4 >= 480
C6:  W  = 8

Tried aggregator 1 time.

MIP Presolve eliminated 10 rows and 13 columns.
MIP Presolve modified 2 coefficients.

All rows and columns eliminated.

Presolve time = 0.00 sec.

Integer optimal solution:  Objective = 4.3500000000e+02

Solution time = 0.00 sec.  Iterations = 0 Nodes = 0

Variable Name           Solution Value

X1                            1.000000
X2                            3.000000
X3                            1.000000
X4                            2.000000
Y1                            1.000000
Y2                            1.000000
Y3                            1.000000
Y4                            1.000000
Z1                            1.000000
Z2                            2.000000
Z3                            1.000000
Z4                            1.000000
W                             8.000000

Problem 'work.lp' read.

Read time = 0.01 sec.

C1:  240 X1 + 240 X2 + 240 X3 + 240 X4 >= 1400
C2:  240 X1 >= 200
C3:  240 X2 >= 600
C4:  240 X3 >= 250
C5:  240 X4 >= 350
C6:  W  = 8

Tried aggregator 1 time.

MIP Presolve eliminated 10 rows and 13 columns.
MIP Presolve modified 7 coefficients.

All rows and columns eliminated.
Presolve time = 0.00 sec.

Integer optimal solution:  Objective = 4.5500000000e+02
Solution time = 0.00 sec.  Iterations = 0 Nodes = 0

Variable Name           Solution Value
```

```
X1                            1.000000
X2                            3.000000
X3                            2.000000
X4                            2.000000
Y1                            1.000000
Y2                            1.000000
Y3                            1.000000
Y4                            1.000000
Z1                            1.000000
Z2                            2.000000
Z3                            1.000000
Z4                            1.000000
W                             8.000000

Problem 'work.lp' read.

Read time = 0.01 sec.

C1:  240 X1 + 240 X2 + 240 X3 + 240 X4 >= 1600
C2:  240 X1 >= 250
C3:  240 X2 >= 650
C4:  240 X3 >= 250
C5:  240 X4 >= 450
C6:  W  = 8

Tried aggregator 1 time.

MIP Presolve eliminated 10 rows and 13 columns.
MIP Presolve modified 7 coefficients.

All rows and columns eliminated.
Presolve time = 0.00 sec.

Integer optimal solution:  Objective = 4.7500000000e+02

Solution time = 0.00 sec.  Iterations = 0 Nodes = 0

Variable Name           Solution Value

X1                            2.000000
X2                            3.000000
X3                            2.000000
X4                            2.000000
Y1                            1.000000
Y2                            1.000000
Y3                            1.000000
Y4                            1.000000
Z1                            1.000000
Z2                            2.000000
Z3                            1.000000
Z4                            1.000000
W                             8.000000

Problem 'work.lp' read.

Read time = 0.01 sec.

C1:  240 X1 + 240 X2 + 240 X3 + 240 X4 >= 1800
C2:  240 X1 >= 250
C3:  240 X2 >= 750
C4:  240 X3 >= 300
C5:  240 X4 >= 500
C6:  W  = 8

Tried aggregator 1 time.

MIP Presolve eliminated 10 rows and 13 columns.
MIP Presolve modified 7 coefficients.

All rows and columns eliminated.

Presolve time = 0.00 sec.

Integer optimal solution:  Objective = 5.3400000000e+02
Solution time = 0.00 sec.  Iterations = 0 Nodes = 0

Variable Name           Solution Value

X1                            2.000000
X2                            4.000000
X3                            2.000000
X4                            3.000000
Y1                            1.000000
```

```
Y2                            1.000000
Y3                            1.000000
Y4                            1.000000
Z1                            1.000000
Z2                            2.000000
Z3                            1.000000
Z4                            2.000000
W                             8.000000

Problem 'work.lp' read.

Read time = 0.02 sec.

C1:  240 X1 + 240 X2 + 240 X3 + 240 X4 >= 2200
C2:  240 X1 >= 300
C3:  240 X2 >= 800
C4:  240 X3 >= 350
C5:  240 X4 >= 750
C6:  W  = 8

Tried aggregator 1 time.

MIP Presolve eliminated 10 rows and 13 columns.
MIP Presolve modified 5 coefficients.

All rows and columns eliminated.

Presolve time = 0.00 sec.

Integer optimal solution:  Objective = 5.5400000000e+02

Solution time = 0.00 sec.  Iterations = 0 Nodes = 0

Variable Name           Solution Value

X1                            2.000000
X2                            4.000000
X3                            2.000000
X4                            4.000000
Y1                            1.000000
Y2                            1.000000
Y3                            1.000000
Y4                            1.000000
Z1                            1.000000
Z2                            2.000000
Z3                            1.000000
Z4                            2.000000
W                             8.000000

Problem 'work.lp' read.

Read time = 0.01 sec.

C1:  240 X1 + 240 X2 + 240 X3 + 240 X4 >= 1200
C2:  240 X1 >= 200
C3:  240 X2 >= 500
C4:  240 X3 >= 200
C5:  240 X4 >= 300
C6:  W  = 9

Tried aggregator 1 time.

MIP Presolve eliminated 10 rows and 13 columns.
MIP Presolve modified 6 coefficients.

All rows and columns eliminated.

Presolve time = 0.00 sec.

Integer optimal solution:  Objective = 4.5100000000e+02

Solution time = 0.01 sec.  Iterations = 0 Nodes = 0

Variable Name           Solution Value

X1                            1.000000
X2                            3.000000
X3                            1.000000
X4                            2.000000
Y1                            1.000000
Y2                            1.000000
Y3                            1.000000
Y4                            1.000000
```

```
Z1                              1.000000
Z2                              2.000000
Z3                              1.000000
Z4                              1.000000
W                               9.000000

Problem 'work.lp' read.

Read time = 0.01 sec.

C1:  240 X1 + 240 X2 + 240 X3 + 240 X4 >= 1680
C2:  240 X1 >= 240
C3:  240 X2 >= 720
C4:  240 X3 >= 240
C5:  240 X4 >= 480
C6:  W  = 9

Tried aggregator 1 time.

MIP Presolve eliminated 10 rows and 13 columns.
MIP Presolve modified 2 coefficients.

All rows and columns eliminated.

Presolve time = 0.00 sec.

Integer optimal solution:  Objective = 4.5100000000e+02

Solution time = 0.00 sec.  Iterations = 0 Nodes = 0

Variable Name           Solution Value

X1                              1.000000
X2                              3.000000
X3                              1.000000
X4                              2.000000
Y1                              1.000000
Y2                              1.000000
Y3                              1.000000
Y4                              1.000000
Z1                              1.000000
Z2                              2.000000
Z3                              1.000000
Z4                              1.000000
W                               9.000000

Problem 'work.lp' read.

Read time = 0.01 sec.

C1:  240 X1 + 240 X2 + 240 X3 + 240 X4 >= 1400
C2:  240 X1 >= 200
C3:  240 X2 >= 600
C4:  240 X3 >= 250
C5:  240 X4 >= 350
C6:  W  = 9

Tried aggregator 1 time.

MIP Presolve eliminated 10 rows and 13 columns.
MIP Presolve modified 7 coefficients.

All rows and columns eliminated.

Presolve time = 0.00 sec.

Integer optimal solution:  Objective = 4.7100000000e+02

Solution time = 0.00 sec.  Iterations = 0 Nodes = 0

Variable Name           Solution Value

X1                              1.000000
X2                              3.000000
X3                              2.000000
X4                              2.000000
Y1                              1.000000
Y2                              1.000000
Y3                              1.000000
Y4                              1.000000
Z1                              1.000000
Z2                              2.000000
Z3                              1.000000
```

```
Z4                             1.000000
W                              9.000000

Problem 'work.lp' read.

Read time = 0.01 sec.

C1:  240 X1 + 240 X2 + 240 X3 + 240 X4 >= 1600
C2:  240 X1 >= 250
C3:  240 X2 >= 650
C4:  240 X3 >= 250
C5:  240 X4 >= 450
C6:  W  = 9

Tried aggregator 1 time.

MIP Presolve eliminated 10 rows and 13 columns.
MIP Presolve modified 7 coefficients.

All rows and columns eliminated.

Presolve time = 0.00 sec.

Integer optimal solution:  Objective = 4.9100000000e+02

Solution time = 0.00 sec.  Iterations = 0 Nodes = 0

Variable Name           Solution Value

X1                             2.000000
X2                             3.000000
X3                             2.000000
X4                             2.000000
Y1                             1.000000
Y2                             1.000000
Y3                             1.000000
Y4                             1.000000
Z1                             1.000000
Z2                             2.000000
Z3                             1.000000
Z4                             1.000000
W                              9.000000

Problem 'work.lp' read.

Read time = 0.00 sec.

C1:  240 X1 + 240 X2 + 240 X3 + 240 X4 >= 1800
C2:  240 X1 >= 250
C3:  240 X2 >= 750
C4:  240 X3 >= 300
C5:  240 X4 >= 500
C6:  W  = 9

Tried aggregator 1 time.

MIP Presolve eliminated 10 rows and 13 columns.
MIP Presolve modified 7 coefficients.

All rows and columns eliminated.

Presolve time = 0.00 sec.

Integer optimal solution:  Objective = 5.5000000000e+02

Solution time = 0.00 sec.  Iterations = 0 Nodes = 0

Variable Name           Solution Value

X1                             2.000000
X2                             4.000000
X3                             2.000000
X4                             3.000000
Y1                             1.000000
Y2                             1.000000
Y3                             1.000000
Y4                             1.000000
Z1                             1.000000
Z2                             2.000000
Z3                             1.000000
Z4                             2.000000
W                              9.000000
```

```
Problem 'work.lp' read.

Read time = 0.01 sec.

C1:  240 X1 + 240 X2 + 240 X3 + 240 X4 >= 2200
C2:  240 X1 >= 300
C3:  240 X2 >= 800
C4:  240 X3 >= 350
C5:  240 X4 >= 750
C6:  W  = 9

Tried aggregator 1 time.

MIP Presolve eliminated 10 rows and 13 columns.
MIP Presolve modified 5 coefficients.

All rows and columns eliminated.

Presolve time = 0.00 sec.

Integer optimal solution:  Objective = 5.7000000000e+02

Solution time = 0.00 sec.  Iterations = 0 Nodes = 0

Variable Name           Solution Value

X1                            2.000000
X2                            4.000000
X3                            2.000000
X4                            4.000000
Y1                            1.000000
Y2                            1.000000
Y3                            1.000000
Y4                            1.000000
Z1                            1.000000
Z2                            2.000000
Z3                            1.000000
Z4                            2.000000
W                             9.000000
```

Developed by Vimal Mayank and Deep Saraf
Reformatted and slightly modified by Mark Austin, May 2003
Last Modified : 11/09/2018 19:24:46