March 29, 2011

Java
- released in 1995 Sun Microsystems
- built multi users operating system
- "write once, run anywhere"
- JRE - Java Runtime Environment

Computer 1
code
↓
compile
→
Computer 2
Run
compiled
code

Computer 1
Java
code
↓
Java Bite code
→
Computer 2
JRE
for machine

- as of 2007, it is open-source
under GNU license

- Java 2 was released in 1999
- J2EE - enterprise
- J2ME - mobile
- J2SE - standard

- no longer use number in name
- Java EE
" ME
" SE

- JDK - development kit
  - not the same as JRE

- .jar file - Java archive, similar to dll file
- .war file - web application archive
- POJO - Plain Old Java Objects
- ~~bean~~ beans = classes

applets - the original way to do dynamic stuff on the web; mini application inside another application; type of component; client side

servlet - server applets; run on app server

JSP - Java Server Pages
- have HTML mixed with Java

<% %> tags
<%! %>
<%= %>
<jsp: ← used for elements

<%@ include file = "mis320.html" %> ← include statement

- Spring Framework - Spring Source (VM ware)
  - largest framework
  - coupling is bad
  - dependency injection } founding
  - aspect orientation } principle

- dependency injection - one piece of code relies on another piece of code, done when code is ran, at the last possible moment

XML - Extensible Markup Language
- Told data
- used in Spring for configuration

- JDBC, Web, Servele

- annotation

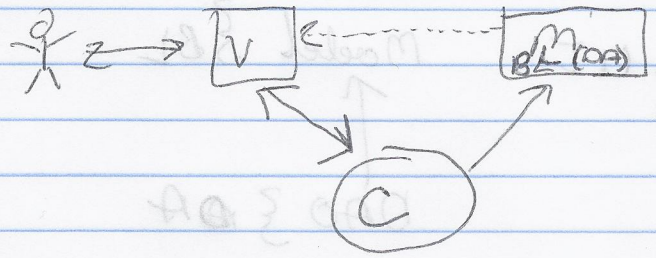
March 31, 2011

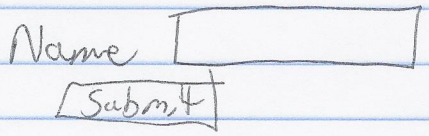
Project Stuff

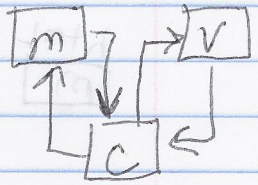
- can have user who doesn't have car
- car doesn't have listing
- user don't have to have listing
- user can have 0 cars

MVC — Model View Controller

- not the same as 3 tier architecture
- Model - representation of context; context specific representation of the data; mostly BL
  View is UI
- DA will also go in model
- controller listens for events from viewer (event listener)







- some business logic can ~~[illegible]~~ be in the controller
- controller passes info to model and then model updates the view
- when the model changes the view changes

April 5, 2011

view is jsp in Java, asp in MS

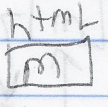
controller is a Java class

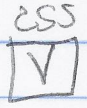
model " " " "

- in large organizations DAO are a seperate class in the model

View
controller } UI

Model } BL
↑
DAO } DA





controller responds to user request and ~~then~~ responds with a view or different model

jQuery - Javascript Framework

April 7, 2011

HTTP Methods (~~reg~~ major)
- get → Idempotent; used for queries

- post → not idempotent

- idempotent - it doesn't change regardless of the number of times it is ran
- idempotent - user's action is not responsible for changing ~~anything~~ anything

- get was originally in ASCII; only represent characters

For Package
Spring Template Project
Spring @ MVC Project

- com.BKM.nameofApp = for top level package
- src/main/java – classes
- " webapp – webpages jsp
- types are capitalized, variables are not
- requestmapping value to return which page;
- "controller" handle all the "@" in a document
- css file in webapp folder
- everything in Java is an object
- to start project, do "run on server"

April 12, 2011

- capstone 26th after 340
- the cookout (MIS) MIS 497 - Jane for Fall

- thursday at 8 am - Final Exam
- written portion in class during Dead Week
- programming during finals week

- SVN & version control; repository
- GIT

- MVC Showcase -

taglib - tag library
"c" is used for core; basic library
<%@ indicates a directive

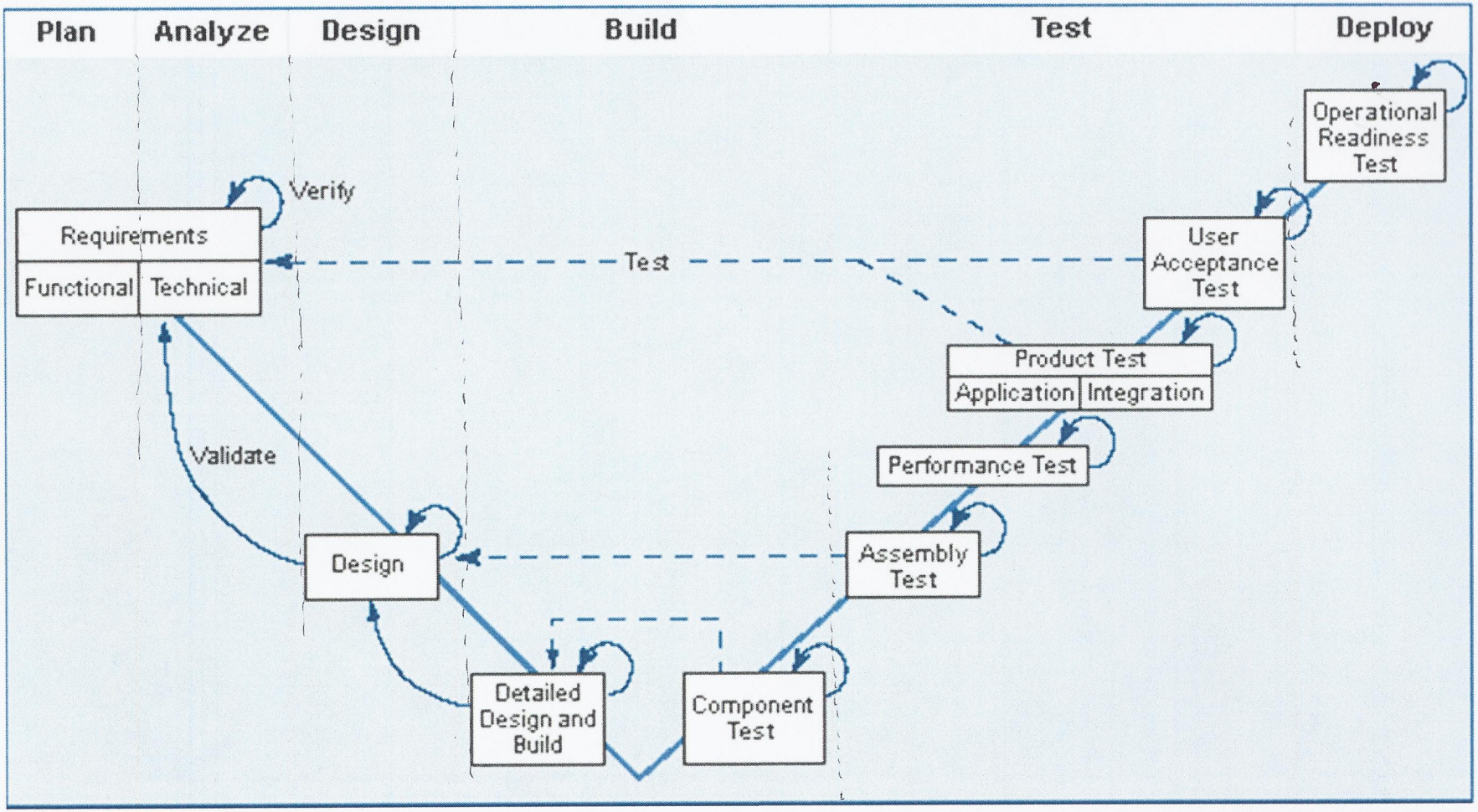
@ ResponseBody - sends it back to page that called it

* is used as wildcard when doing request mapping

Bean means model or class

-

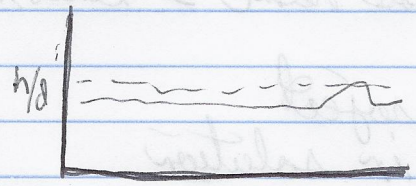
- no password
- ~~assume~~ ~~default~~ get username when loading
- document based on priorities
  - H are required
  - M, L are ~~optional~~ optional


Plan
Analyze
Design
Build
Test
Deploy
Verify
Requirements
Functional
Technical
Test
Validate
Design
Detailed Design and Build
Component Test
Assembly Test
Performance Test
Product Test
Application
Integration
User Acceptance Test
Operational Readiness Test

April 14, 2011

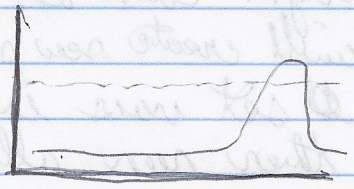
Methodology for projects
- value: consistency
- each type has its strengths and weaknesses

CMMI L1



ideal

CMMI L5



not ideal

Requirement

component test - run every line of code; unit test

assembly test (system test)
- verify traceability to design deliverables
  - function
  - interface

product test (End to End) (Regression)
- verify HLR / LLR requirements
- performance
- regression test for enhancement to existing system

User acceptance test

~~Compont~~ Component testing in Visual Studio
- right-click on class name > unit test
- will create new project
- add unit in test project
- then run all tests in solution

CSS Zen Garden - csszengarden.com

icsharpcode.com

Test Script
- key scenes
  - pre-reqs
  - input data
  - requirement mapping
  - detailed instructions
  - expected v actual results

April 19, 2011

- reviewed testing in C#
- MVC 3 version

DB connection info
Name: MIS320Spring11
user: MIS320
pwd: password
server: mis330 - ~~sqlcmd~~ sql
cil-server.mis330-sql.cba.ua.edu (OFF campus)

- model - BL, DAO
  View - UI
  Controller - eventHandler

- App_Data folder - data access

- JQuery - Java Framework

- ~~Cont~~ controller for each folder in view

- ViewBag

@model - directive, strong typed view

DbContext ~ database connection object

property - a method that the language treats as public

- DbSet ~ holds table of data from table

- System.Data.Entity.ModelConfiguration.Conventions, Edm.Db

modelBuilder, conventions, Remove

4/21/2011

Store Manager - has add, edit and delete records from DB

- look in exam app for db connection string formatting formatting

- "ViewBag" - holds data that are related to the current view

- Editor for Model
  - tied to a for in the editor template folder
  - editor template should be named for model

- [HttpPost] to define a post method (not get)

- GitHub - repository website

April 26, 2011

AJAX - Asynchronous Javascript and XML
- appears to ~~have~~ run ~~faster~~ faster

Final Exam Stuff (similar to midterm format)
- written part on Thursday
  - definitions ~~questions~~
  - costs
  - HTML stuff only (CSS, JS, etc)
  - DOM
  - client-side and server side
  - server types based on code
  - MVC questions
  - ~~request~~ ~~response~~ class that handles requests, cookies, etc
  - AJAX advantages
  - pseudocode
- there's mays @ 8 am Lloyd 16 (computer lab)
  - server project description
  - have to build model, view, and/or controller

- VIN, make, ~~model~~ model, year — required fields for car
- default to 0 on mileage
- ~~price~~ all #s should be defaulted to zero
-

March 31, 2011

Review of Bonus Midterm Question

- create class
- DAO object

```
Public class EventMgr
  Dim MyEDAO as new DAOEvent
  public function AddEvent(...)
    Dim success as boolean = false
    if startDate < EndDate then
      MyEDAO.SaveEvent(.....)
      success = true
    End If
    return success
  End Function
  public Sub removeEvent(ByVal EventID as int)
    MyEDAO.deleteEvent(EventID)
  End Sub
```

```
import Event Mgt
  Public Class EventForm
    Dim MyE as new EventMgr
    If MyE.AddEment(tbxEventID.text, ...) then
      MsgBox.Show("Successful")
    Else
      MsgBox.Show("Unsuccessful")
    End If
  End Sub

  Public Sub btnDelete_Click
    MyE.RemoveEvent(tbxEventID.text)
  End Sub
End Class
```

sub
Public btnSubmit_Click

(7)

Print Name: Kenneth Robinson

Section III: Writing Code (10 pts each)

1. Write a Business Logic layer VB component in the EventMgt namespace that has methods to add an event and delete an event. The event entity of your database has fields for Event ID, Name, Location, Start Date, and End Date. Assume you have a DA component called DAOEvent that has methods SaveEvent(EventID as int, Name as string, Location as string, StartDate as date, EndDate as date), and DeleteEvent(EventID). Your component should ensure that the start date is before the end date before it tries to save the event to the database.

```
imports EventMgt
Public Class EventBL
  Dim Name, Location as String
  Dim StartDate, EndDate as Date
  Dim EventID, didRun as Integer
  Public Sub Save(EventID, Name, Location, StartDate, EndDate)
    if StartDate < EndDate then
DAOEvent.SaveEvent(EventID, Name, Location, StartDate, EndDate)
      didRun = 1
    End If
  End Sub
  Public Sub Delete(EventID)
    DAOEvent.DeleteEvent(EventID)
  End Sub
End Class
```

2. Write the code for a class that uses the component you wrote above. It should take a user's input from text boxes and implement both methods of your component. It should display some kind of message telling the user the method completed successfully or not.

```
Public Class EventUI
  Dim BLEvent as new EventBL
Sub btnSave Handles Click
  BLEvent.Save(tbxEventID.text, tbxName.text, tbxLocation.text, tbxStartDate.text, tbxEndDate.text)
  if didRun = 1 then
    MsgBox("Saved Successfully")
  else
    MsgBox("Did Not Save")
  End If
End Sub
Sub btnDelete Handles Click
  BLEvent.Delete(tbxEventID.text)
  if didRun = 1 then
    msgbox("Deleted Successfully")
  else
    msgbox("Did Not Delete")
  End If
End Sub
End Class
```

User
loginID (PK)
fname
lname
password
email
birthdate
billingstreet
billingcity
billingstate
billingzip
homephone
cellphone
shipstreet
shipcity
shipstate
shipzip

User_Rating
ratingID (PK)
loginID_ratee (FK)
loginID_rater (FK)
ratingNum
comment

Car
VIN (PK)
loginID (FK)
Make
Model
Year
Color_int
Color_ext
Desc
Mileage
MPG_city
MPG_highway
Transmission
Engine
Rebuild
Rotisserie
Wheels
Fuel_type
Doors
Drivetrain
Vehicle_title

Forum
topicID (PK)
loginID (FK)
subject

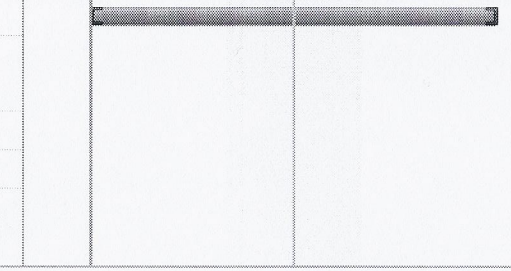
Post
postID (PK)
topicID (FK)
loginID (FK)
timestamp
message

Listing
listingID (PK)
VIN (FK)
loginID_buyer (FK)
loginID_seller (FK)
ReservePrice
startPrice
endPrice
buyNowPrice
auctType
startDate
endDate

Bid
bidID (PK)
loginID (FK)
listingID (FK)
bidPrice
timestamp

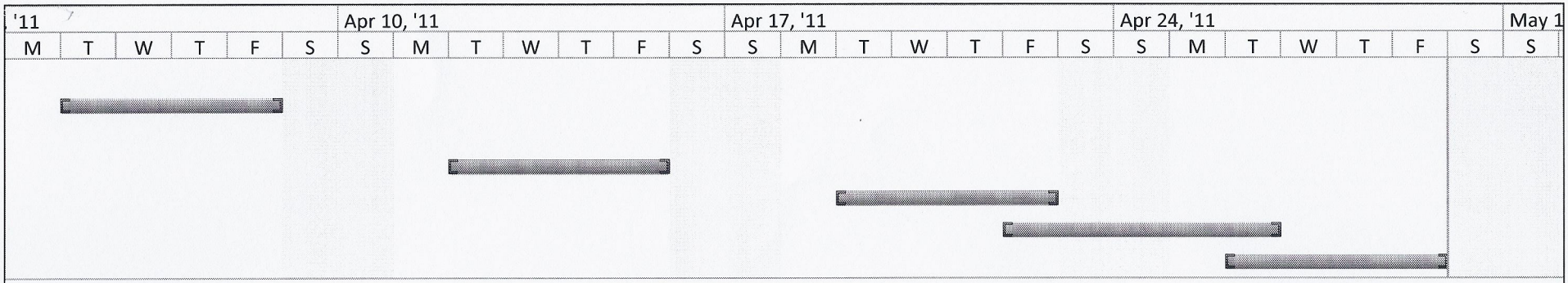
- You can be a member but you don't have to list a car (buyer only) or participate in forum
- if you don't sell a car, you can relist it
- forum doesn't have to have subforums
- every transaction is good

ID	Task Mode	Task Name	Duration	Start	Finish
1		Project Charter / ERD	4 days	Thu 3/24/11	Tue 3/29/11
2		Project Plan / Product Requirements	4 days	Tue 4/5/11	Fri 4/8/11
3		Design Layout	4 days	Tue 4/12/11	Fri 4/15/11
4		Prototype	4 days	Tue 4/19/11	Fri 4/22/11
5		Final Project	3 days	Fri 4/22/11	Tue 4/26/11
6		Revised Final Project	4 days	Tue 4/26/11	Fri 4/29/11



Project: MIS320 Timeline.mpp
Date: Sun 3/27/11

Task	External Milestone	Manual Summary Rollup
Split	Inactive Task	Manual Summary
Milestone	Inactive Milestone	Start-only
Summary	Inactive Summary	Finish-only
Project Summary	Manual Task	Deadline
External Tasks	Duration-only	Progress



Project: MIS320 Timeline.mpp
Date: Sun 3/27/11

Task		External Milestone		Manual Summary Rollup	
Split		Inactive Task		Manual Summary	
Milestone		Inactive Milestone		Start-only	
Summary		Inactive Summary		Finish-only	
Project Summary		Manual Task		Deadline	
External Tasks		Duration-only		Progress	

Robinson, Kenneth

67/80

MIS320-001 Print Name: Kenneth Robinson

Midterm Exam – March 10, 2011

You have 1 hour and 15 minutes to complete this exam. WRITE YOUR NAME ON EVERY PAGE! -4

Section I: Short Answer (2pts each)

1. A component is modular, deployable, and replaceable.
2. How many ways should there be to return from a function? 1
3. What does it mean to say a component encapsulates implementation? Why does it do that? -2
   It can be added to an application or system and will work as long as it has the inputs it needs to function.
4. What does it mean to say designing an application is an iterative process?
   It requires going back and forth, doing the same steps multiple times. Updating, removing, and adding things as you build the application.
5. In a classic 3-tiered application, the three tiers are user interface, business logic, and data access.
6. Write the single line of VB code that declares an interface that accepts two integers to be added together: -2
   Dim myAdd as New IntegerAdd(1, 2)
7. In a physical architecture that requires updates to be made to each user's computer, the business logic resides on the client machine.
8. Name at least two advantages to building software in components instead of monolithic blocks
   Don't have to replace the entire system when updating. Multiple can work on building the system at the same time.
9. Coupling is a measure of the extent to which one software component relies on another software component to do its job.
10. The point of this class is to teach you to do what?
    Effectively and properly design desktop and web applications.

Print Name: Kenneth Robinson

on II: Reading Code (20 pts) – Answer the questions following this piece of code

-3

```
Imports orderprocessing

Public Class OrderForm

    Dim myorder As New orderprocessing.order
    Dim cups As Integer
    Dim type As String
    Dim subtotal As Decimal
    Dim taxAmt As Decimal
    Dim total As Decimal

    type = tbxType.text
    cups = tbxNumOfCups.text

    subtotal = myorder.calcSubTotal(type, cups)
    taxAmt = myorder.calcSalesTax(subtotal)
    total = myorder.calcTotal(subtotal, taxAmt)

    console.writeline("Kim's Koffee Kettle")
    console.writeline("You purchased " & cups & " of " & type & " coffee.")
    console.writeline("Subtotal: " & subtotal)
    console.writeline("Tax: " & taxAmt)
    console.writeline("Total: " & total)


End Class
```

11. (5 pts) What is the namespace of the class of which myorder is an object? orderprocessing

12. (5 pts) If this class was written in a 3-tiered application, what tier would this class belong to? UI

13. (10 pts) In the space below, write the API definition for the order component (assume no members):

-3 class description

calcSubTotal(drinkType, NoOfCups)
Calculates the subtotal of the order based on the number of cups and the type of drink.

calcSalesTax(purchaseSubtotal)
Calculates the amount of sales tax due for purchasing drinks.

calcTotal(subtotal, taxAmt)
Calculates the grand total of the purchase.

Print Name: Kenneth Robinson

Section III: Writing Code (10 pts each)

14. Write a VB component in the Inventory namespace that has methods to check whether current levels of an inventory item are below the reorder point, and to add quantities to a given item. You can hardcode the reorder point, but the first method must accept the quantity on hand and the second must accept the quantity to add. Also assume you have a data access class called DAOInventory you can use to save the added quantity. It has a method called AddQuantity(Qty as integer).

```
Class InventoryClass
  Public Sub NeedsReordering(ByVal CurrentQty as Integer)
    If CurrentQty < 25 then
      msgbox ("Reorder Product")
    EndIf
  End Sub
  Public Sub UpdateInventoryLevel(ByVal AddQty as Integer)
    Dim InvDAO as New DAOInventory

    InvDAO.AddQuantity(AddQty)
  End Sub
```

15. Write a class that uses the component you wrote above. It should take a user's input from text boxes and implement both methods of your component.

```
Imports Inventory
Public Class frmInventory
    Sub btnCheckLevel_Click
      Dim IReport as New Inventory.InventoryClass

      IReport.NeedsReording(tbxCurrent.text)
    End Sub
Sub btnAddToLevel_Click
  Dim IReport as New Inventory.InventoryClass
    IReport.UpdateInventoryLevel(tbxAddLevel.text)
End Sub

End Class
```

Print Name: Kenneth Robinson

Section IV: Specifications (40 points)

Shane, the most interesting entrepreneur in the world, is opening yet another business. This time he's opening a restaurant that serves classic southern cuisine. He has hired you to develop a system that will allow servers to input the customers' orders, tell the chefs what to make, alert the server when the order is ready, print the check, and process the payment.

For a given table, the server will first take the guests' drink orders. The server will enter that information, and then return to enter the meal orders once the guests have ordered their food. If the restaurant has run out of the meal chosen, the system will alert the server before they move on to the next guest.

Use the information above to complete the models below:

I. High Level Requirements

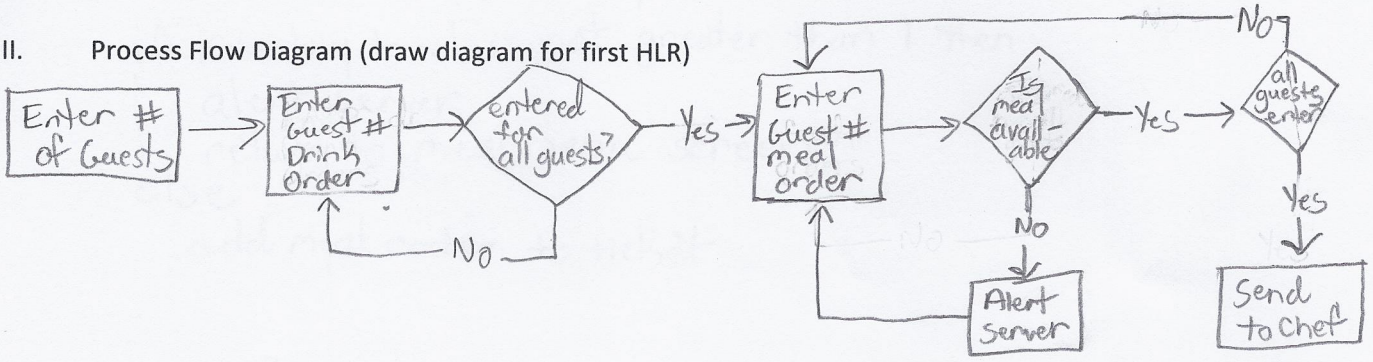
1) Input Customer Order
2) Display Order to Chefs
3) Alert When Order is Ready
4) Print Receipt
5) Process Payment

II. Product Requirements (for first HLR only, at least 5 additional requirements)

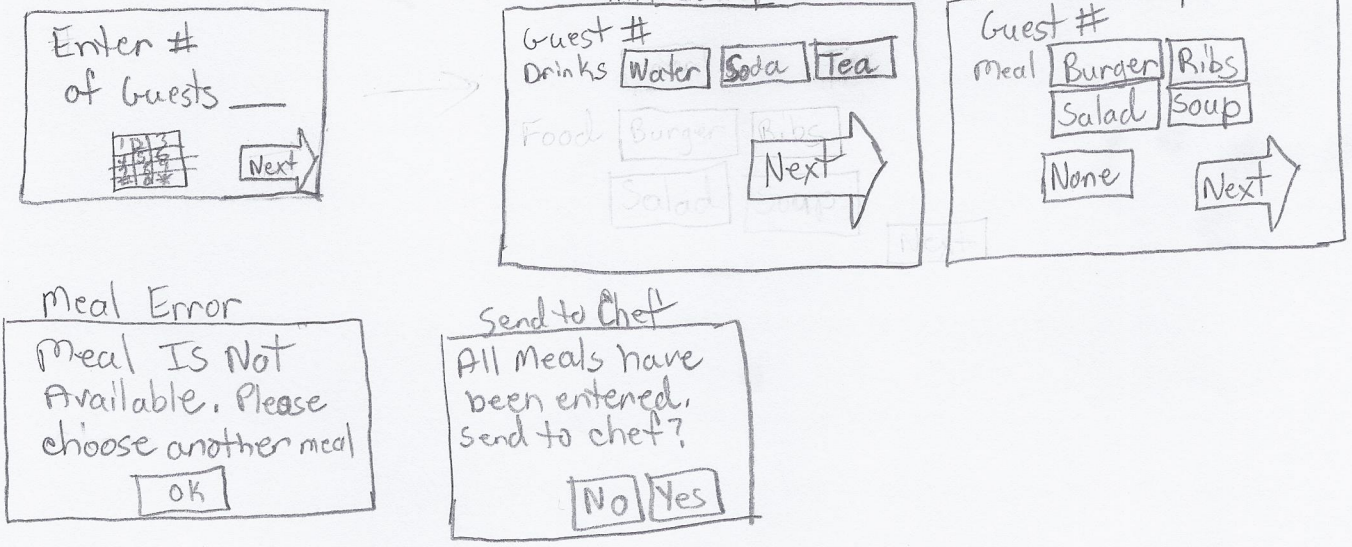
a. HLR1-Input Customer Order
   i. Allow server to input number of guests
   ii. Allow server to input drink selection of each guest
   iii. Allow server to input meal order for each guest
   iv. Inform server if meal is unavailable
   v. Allow server to submit meal to chefs
   vi. Allow server to enter special instructions for each meal
   vii. Allow server to skip guests' meal (mistake)

Print Name: Kenneth Robinson

III. Process Flow Diagram (draw diagram for first HLR)



IV. Input/Output Design (draw screens to handle first HLR)



Print Name: Kenneth Robinson

-6

V. Psuedocode (write the psuedocode for the event that submits one guest's meal order to the system)

-6 Detail

```
btnSubmit_Click
    check current inventory level for meal
    if inventory level is not greater than 1 then
        alert server
        return to meal menu screen
    else
        add meal order to ticket
```

VI. Product Requirements Revisited (add any requirements for the first HLR you discovered through building the previous models)

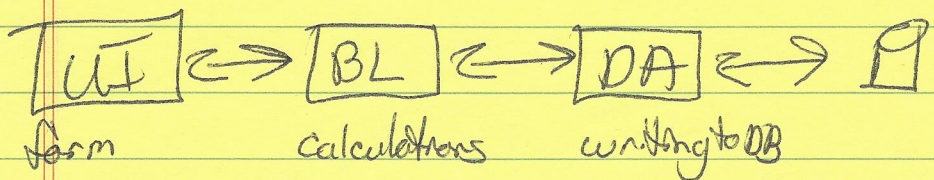
viii) meal preparedness (well-done, medium-rare, no sauce, condiments, etc)
ix) remove a meal (server makes entry error)

MIS 320 Midterm Study

HLR ↔ LLR ↔ Process flows ↔ I/O Design ↔
... Data Design ↔ Class Design

I/O is interface
Data Design is ERD (Entity Relationship Diagram)

- members are variables; methods are functions and sets
- physical architecture is machine
- logical " " software
- high cohesion, low coupling
- Component - piece of software that is modular, deployable, and replaceable which encapsulates implementation and exposes a set of interfaces
  ex: stereo system with CD player, record player, etc



- buffer - area of memory that the system maintains before writing to file
- encapsulates - black box - don't have to know how it works in order to use it