MIS 320 – Spring 2011
The University of Alabama
Application and Information Architecture
Syllabus

Instructor: **Shane Givens**
Office: Bevill 1103
Phone: 348-5525 **E-Mail**: sgivens@cba.ua.edu
Office Hours: By appointment
Grader: Tedd Meyer; tameyer@cba.ua.edu
All classes meet T/TH 11:00am - 12:15 p.m., Bidgood 340

Pre-requisites: CS391 or CS114 or CS150, AND CS491 or CS124 or CS250

Description:
This course builds on your prior problem solving and programming exposure. It explores designing and constructing Enterprise Applications using defined application architectures and programming patterns. The class will cover topics including, but not limited to, and in no particular order:

- Application Architecture and Patterns
  - History of Application Architecture
  - Components
  - Coupling & Cohesion
  - Classes as components
  - Interfaces
  - Architecture in solution design
    - Reuse and Maintainability
    - Frameworks
  - Logical vs. Physical Architecture
  - Configuration vs. Convention
  - Examples
    - Physical
      - Desktop
      - Client-Server
      - Web Apps
      - "Cloud"-based
    - Logical
      - Legacy (COBOL)
      - Client Server
      - N-Tier
      - MVC
      - SOA/Web Services
- Web Programming Techniques
  - HTML, XML, CSS
  - Server-Side Scripting
    - ASP.NET
    - JSP
    - PHP

Textbooks: None. A student membership to ACM is required.
Additional Required Materials:
Thumb drive, Google docs or windows live skydrive to store files

GRADES: The final grades for this course will be based upon the following components:

Quizzes, Homework & Programs 50% Exams 40% Participation 5% Mentoring 5%

Final Exam: Your final exam will be team-graded. Your grade will be calculated by taking the average raw score of your Final Project teammates multiplied times 0.4 and adding your raw score multiplied times 0.6 (i.e. your teammates will make up 40% of your final exam score). The final will draw heavily from what you learn in doing your Final Project.

Late Assignments: NO assignments will be accepted late. Exceptions may be made for severe illness or a death in the family. If you anticipate other conflicts, make sure you start your assignments on the day they're assigned.

Student assignments are graded based on the following guidelines:

Class Grades:

A: exceeds standards (goes beyond stated measures)
B: meets stated standards with high quality work
C: marginally meets stated standards
D: does not meet stated standards
F: unacceptable

Student task-level performance evaluations will be measured on the following standards:

Task-level performance:

A: Can do assigned task with high degree of independence and high quality
B: Can do assigned task with structured approach and high quality
C: Can do assigned task with structured approach, but requires outside critique to improve quality
D: Can not do assigned tasks, and/or deliverable does not meet stated standards
F: unacceptable

Meaning of "grades" to external community (potential employers, parents, ...)

Suggestion for future professional development, if a particular student earns a ___ the recommended type of employment is _____:

A: recommended for positions that require innovation, unstructured approaches, & synthesis
B: recommended for positions that require normal amounts of training & supervision
C: recommended for positions with a great deal of supervision and structure

COMPUTER AND PROBLEM-SOLVING SKILLS: Students should plan to spend a great deal of time (this means several hours) working on the exercises and programs. Allow time for multiple iterations to produce the desired results.

LAB ASSIGNMENTS: A series of individual problem-solving exercises involving application software will be assigned. These exercises will provide students with hands-on experiences. The lab assignment exercises are mandatory and will be divided into grading categories:

1) *Basic assignments*, All basic assignments must receive a passing grade. Students who miss 1 assignment will have their final grade dropped one-half grade. Any student missing more than 1 assignment will not pass the course.
2) *Projects*, which will be graded on a traditional 0-100 basis. Note, it is possible that some of the assignments will be given in class and due that same class (that is quizzes are possible). A student must receive a passing grade on the final project to receive a passing grade in the course.
3) For assignments marked as "Redo," you have one week from the time to assignments are returned to the class to turn them back in. If this is not done, the assignment will be marked as a fail.

Peer Mentoring by MIS students will be available. Mentors will expect you to have prepared before you meet with them. They are to help structure your thoughts and help you when you may be confused about a concept. The time of the meeting will be determined by student availability.

Other Grading tips:

- **Handwritten assignments are NOT acceptable.**
- **Multi-paged assignments must be stapled**

ACADEMIC MISCONDUCT: Plagiarism is unacceptable and will receive a failing grade for the course. Academic misconduct is a serious offense and will not be tolerated. Any student found plagiarizing or otherwise cheating on assignments or exams will receive a grade of "F" and will be subject to disciplinary action under the University of Alabama regulations. Please review the University of Alabama Student Handbook.

Assignments designed to be done individually that are done in groups will also be subject to failing grades. Likewise group assignments done by multiple groups will be subject to failing grades.

EXAMINATIONS: Two examinations will constitute 40% of your grade. Exams will include definitional, conceptual, syntactical and logical components. Examinations must be taken on the scheduled dates and times.

MAKE-UPS: Make-up exams are allowed only in extreme cases. Students are responsible for any changes made to examination dates or assignments announced in class. Computer lab assignments and independent lab exercises are mandatory and are to be submitted at the beginning of the class in which they are due. A late penalty will be assessed as discussed previously. **Failure to complete any exercise, assignment, or test satisfactorily will result in failing the course.**

E-MAIL: Expect updates to class information throughout the semester. Material will be sent to your crimson mail accounts. Each student is responsible for the maintenance and security of their personal account.

SEVERE WEATHER and **SECURITY ALERTS** are a possibility. In the event of an emergency, we will adhere to the following actions in accordance with University policies.

TORNADO WARNING: Move to the Lower Level, inside classrooms, offices or corridors. Remain until the warning has expired. Classes are cancelled until the warning expires.

FIRE/FIRE ALARM: Evacuate the building and stay out of the building at a safe distance until authorized to return.

DISABLED STUDENTS: To request disability accommodations, please contact the Office of Disability Services at 348-4285. After consultation with that office, contact your professor. However, it is the student's responsibility to make arrangements for the accommodations on a timely basis. Special arrangements for exams must be made at least one week prior to the exam date or your instructor is not required to provide requested accommodations. Any request for special arrangements made less than one week prior to an exam date may not be able to be honored.

MIS 320

1/13/2011

- syllabus - download
- no textbook
- ACM - student membership
- 2 nights per semester for mentoring

- loose coupling, high cohesion
- framework - implementation of architecture

1/18/2011

Problem Solving

- define problem
  - write things down
- ask ~~questions~~ questions

- solves the problem (define) and create solution

- defining the problem -
  - high level requirements (Business Req)
  - low level requirements (Product Req)
  - process flows

- high level ~~req~~ requirement (Business Req)
  - top level process
  - not defined by technology implementation

- low level requirements

- process flows
  - programs
  - steps needed to solve
  - how problem is done now

= Design the Solution(s) =
- HLR
- LLR
- Process Flow
- I/O Design (Form Definition, Control Definition)
- Data Design (ERD)
= Class Design

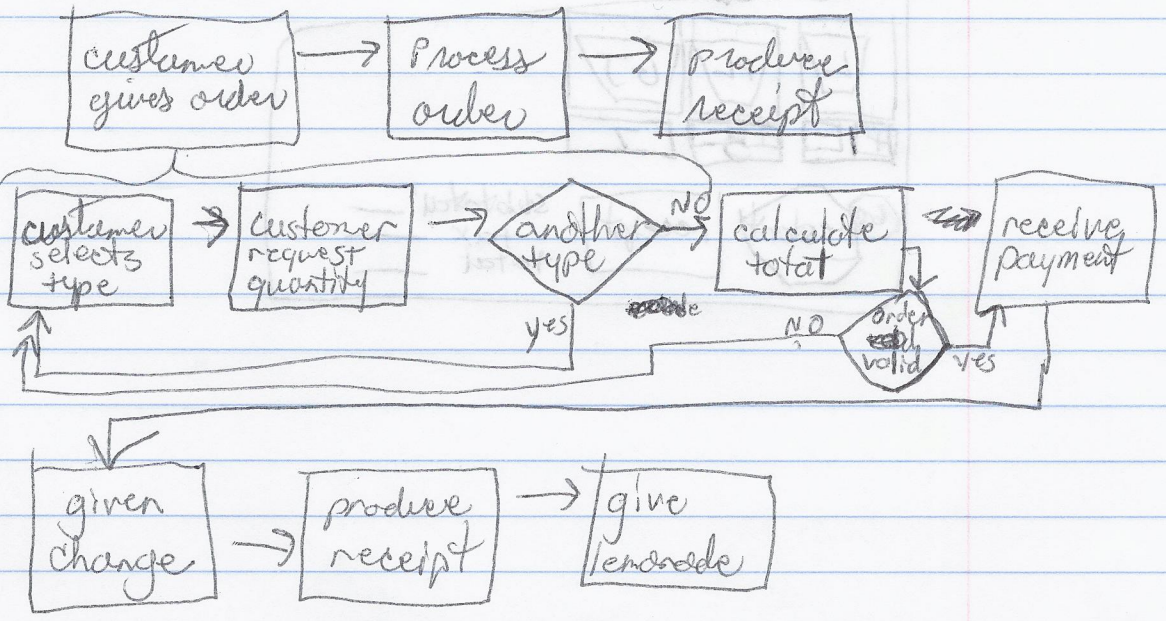
- Pseudo-code with definition

Lemonstand Example

- HLRs
1.0 ~~take an~~ process the order
2.0 return receipt (reporting)
~~3.0~~

- Product Requirements:
1.0 Process the order
1.1 Solution must handle types of drink
1.2 " " allow multiple units per sale
1.3 " " " " prices per product
1.4 " " " calculating sales tax
1.5 " only handles cash
1.6 Calculate total
1.7 ability to change quantity + recalculate
1.8 Calculate change

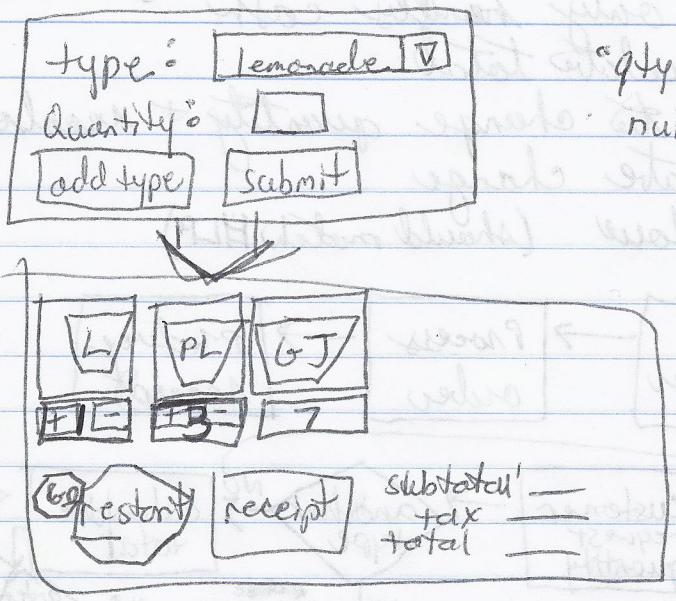
- Process Flow (should match HLR)



- update older models before going on to the ~~next~~ next step

- additional Product Requirements
  1.9 customer must receive lemonade when paid for
  1.1.1 ~~1.1.1~~ solution must handle multiple ~~drinks~~ types per transaction

- I/O Design -



"qty: is there a max; enter numbers;

Jan 20, 2011

- check for mentoring dates for ES 120, 220

cont. from previous day

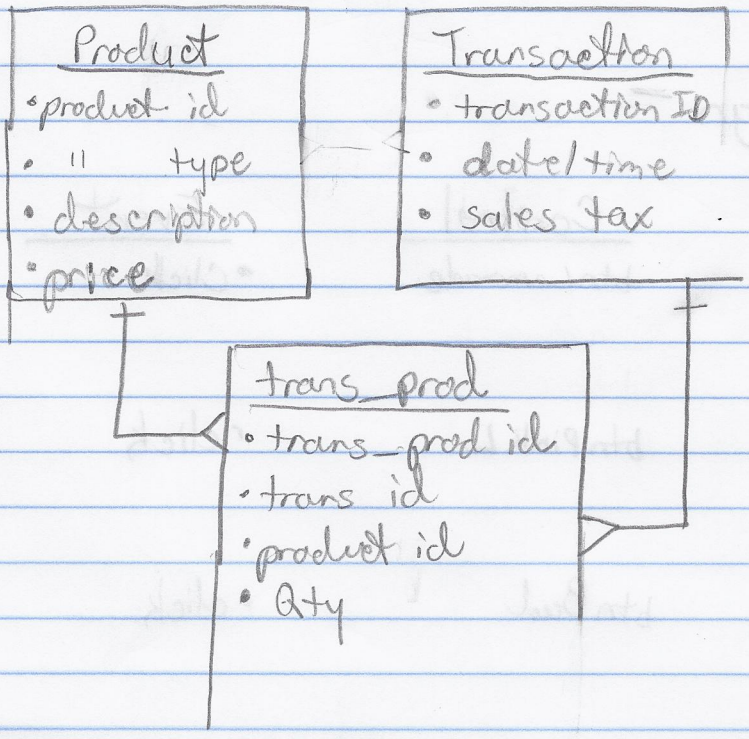
1.8.1 must display change amount
1.8.2 User must be able to enter amount tendered

1.11 display error message if wrong ~~tendered~~ amount is entered

- Data Design consists of ERD, Data Dictionary

i.e. FirstName - tbl Customer - Represents the customer's first name

entities →



- Class Design
  - blueprint of a object; something we want to know something about and do something with

Product
• type • description • price • ID • Qty
• gets/sets

Transaction
• ID • date/time • sales tax • array of products
• get/set • calculate total • " sales tax • " subtotal • " change

- I/O Design -

Form	Control	Event
Order Entry	btnLemonade	• Click
	btnPinkL	• click
	btnBeet	• click

- do pseudocode for classes, the pseudocode for I/O Design

Class Definition
- Class transaction

- members
  - ID
  - DateTime

- methods
  - Func calcSubtotal()
  -

- Check eLearning for homework
  - hand draw everything or on computer

1/25/2011

AIMS meeting
- International Paper
- DAXKO

Architecture architecture

framework

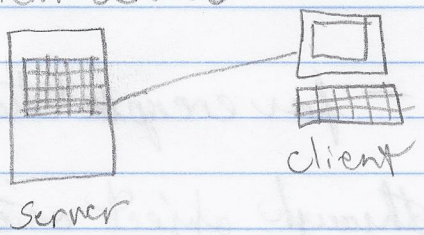
- physical architecture - how machines are arranged
- logical architecture - how the software is arranged

- 8 bits = 1 byte
- first processors had 8 channels, hence the standard of 8 bits
- "debug" comes from moths getting in vacuum tubes

Physical Architecture

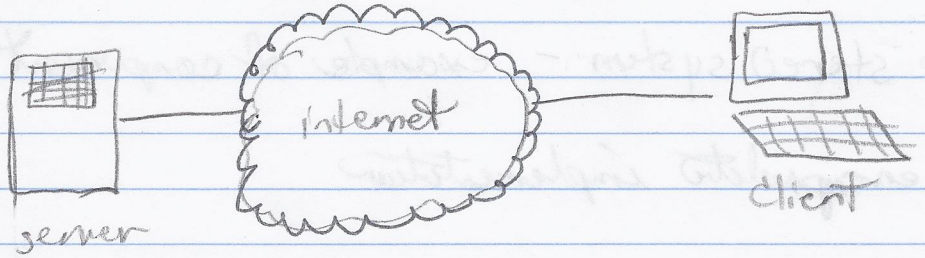
- Client Only
  - everything on one machine

- Client-server



  - data on server
  - software on client

-



Web Apps
- client-side scripting: HTML, JavaScript
- server-side scripting: PHP, ASP

Logical Architecture

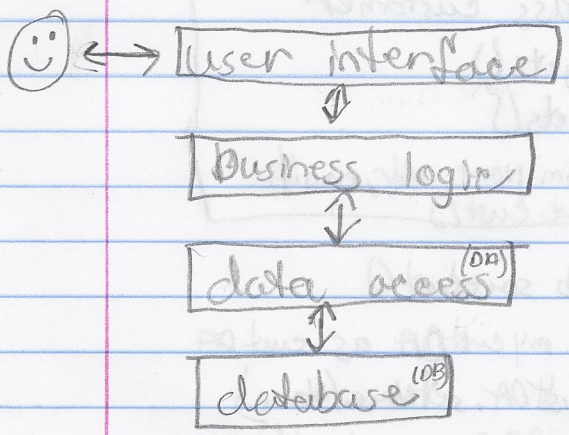
- component - piece of software that is modular deployable and replaceable that encapsulates implementation and exposes a set of interfaces
- cohesion is good
- cohersion is bad
- members and methods - for encapsulation
- implement encapsulate through object orientation
- stereo system - example of component
- encapsulate implementation

1/27/2011

- on a computer is example of component inside a component
- low cohesion
- zero coupling - components aren't connected to other components

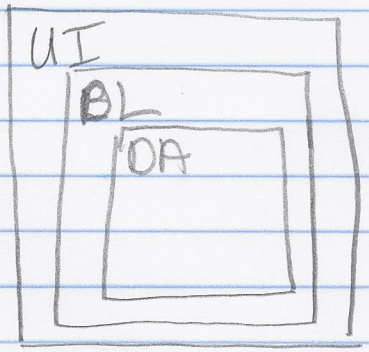
- if an item has low cohesion (all functionality in one box) then you have to replace the entire item (i.e. add cd player to cassette radio); also more expensive to replace

n-tier architecture
- basic is 3-tier

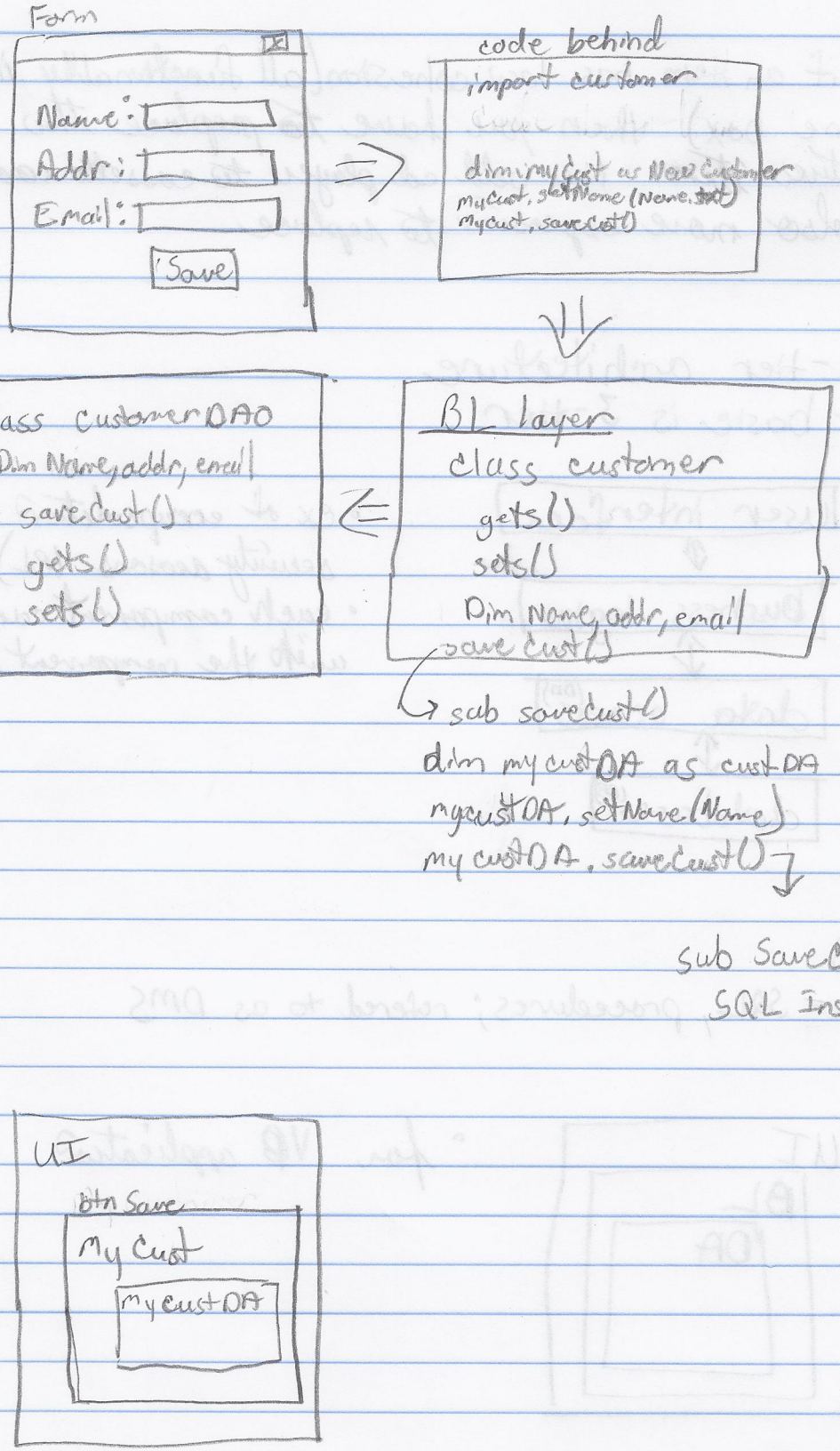


- ex of encapsulation - for security reasons (BL)
- each component can only deal with the component next to it

DB
DA - SQL, procedures; refered to as DMS



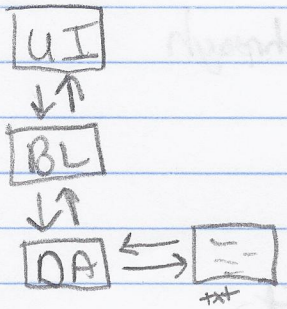
- for VB application


Form
Name:
Addr:
Email:
Save
code behind
import customer
dim myCust as New Customer
myCust.setName(Name.text)
myCust.saveCust()
Class customerDAO
Dim Name, addr, email
saveCust()
gets()
sets()
BL layer
class customer
gets()
sets()
Dim Name, addr, email
saveCust()
sub saveCust()
dim myCustDA as custDA
myCustDA.setName(Name)
myCustDA.saveCust()
sub SaveCust()
SQL Insert
UI
btnSave
myCust
myCustDA

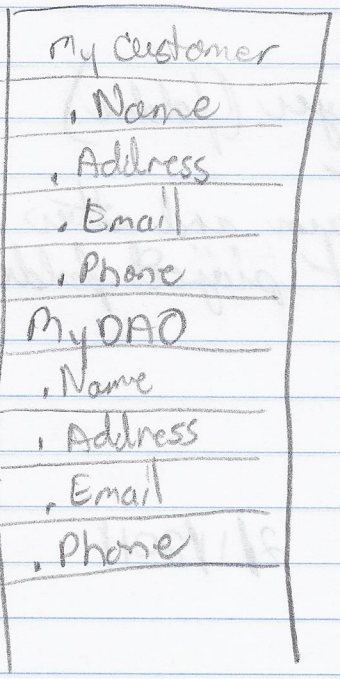
- New Project
- add three folders
  - BL
  - UI
  - DA
- move default form
- add new class to each layer (folder)
- each BL class for DA class
- imports statement for form and other classes
- put documentation in root of project folder

2/1/2011

- component
- declaration of function is an interface
- interface specification - jacks must match



- how you close out a process
- composition - one class is part of another class



← Memory stack example

- when sets are called a class is copied to another

parameter - what variable is being passed in function

argument - what is being passed through

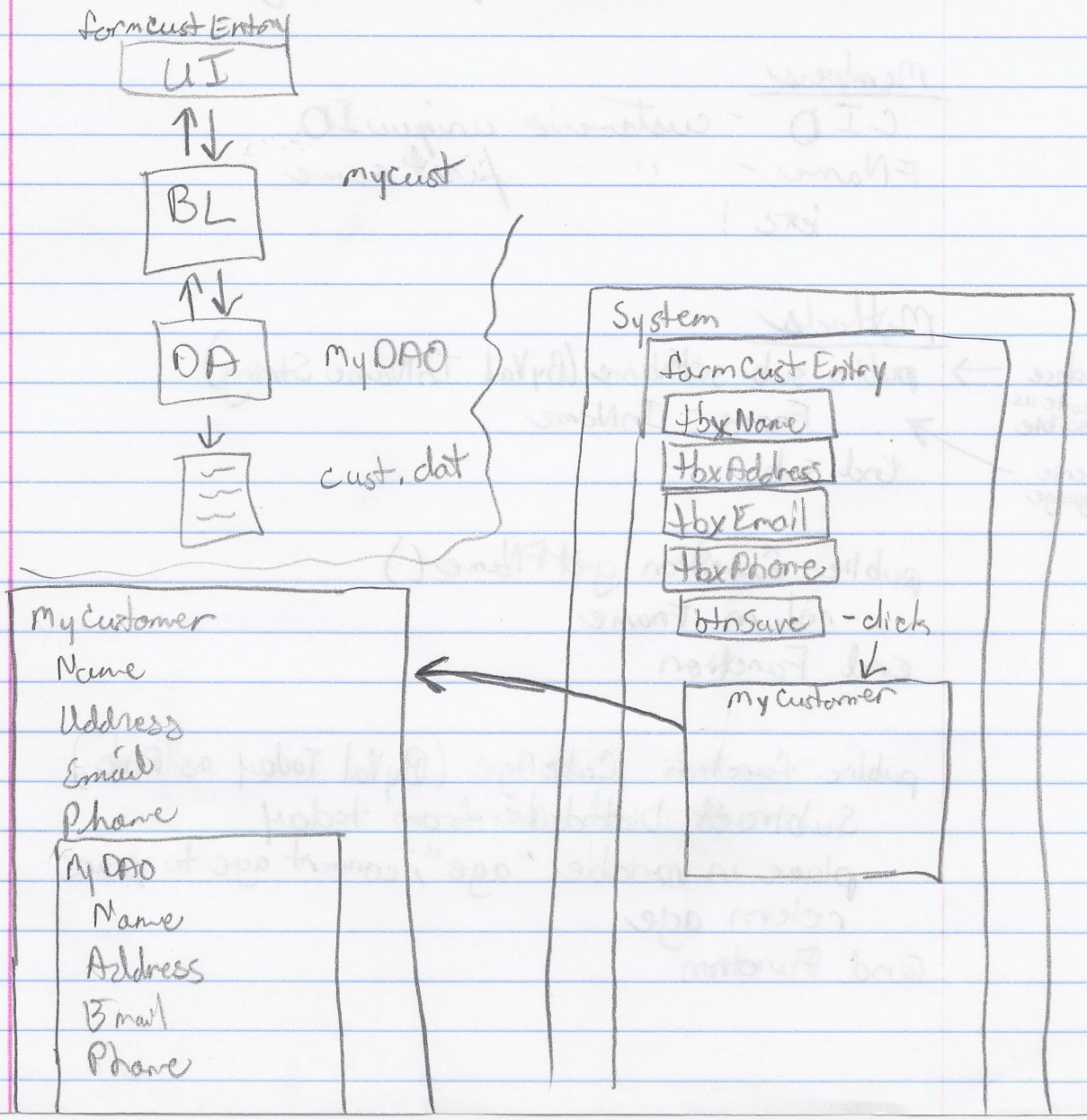
ex. fitting shape into board

- append only works with write

- overloaded constructor - pass parameter to a new class constructor

- buffer - area of memory that the operating system maintains before writing
  - come from time costs

- check elearning for example



2 | 34 / 68 / 6

34 / 68 / 102

2/3/2011

Assignment 1 Review
- start from last point

Class Definition Example → Class Customer - represents customers holding contact data and having the ability to save and delete customers from storage

Members
- CID - customer's unique ID
- FName - " first name
- etc!

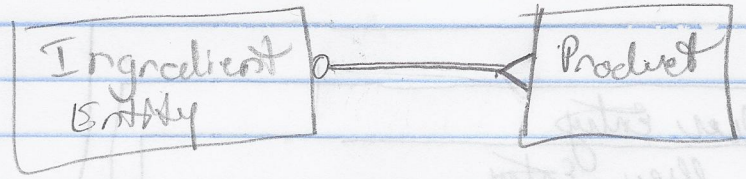
Methods

get close to code as possible → generic language

```
public sub setFName (ByVal InName String)
    Fname = InName
End Sub

public function getFName ()
    return Fname
End Function

public function CalcAge (ByVal Today as Date)
    Subtract birthdate from today
    place in variable "age"; convert age to years
    return age
End Function
```



- document is not an entity; store info with order
- cust Entry ex for a Control List

2/8/2011

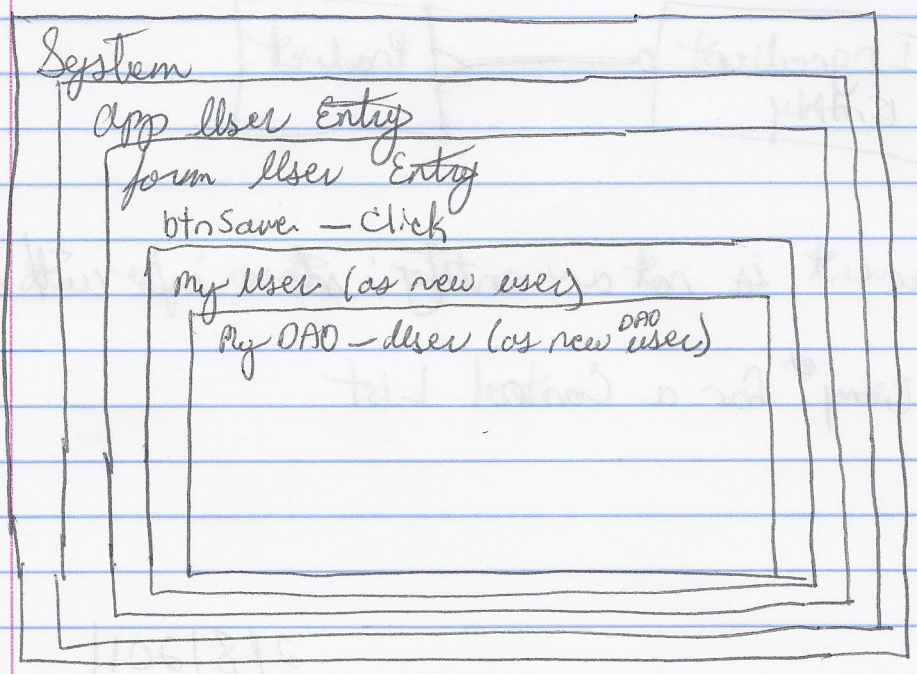
- encapsulation = the black box
- API = application program interface

2/10/2011

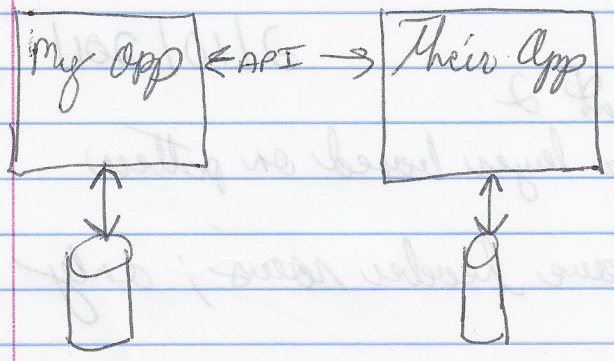
Review of Assignment 2

- Business Logic — a layer based on pattern
- data files don't have header rows; only reports do

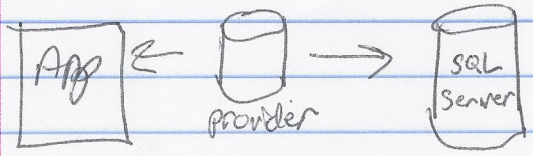
12
12
24
8
32
15
47
15
62
13
75
15
90



Interfaces

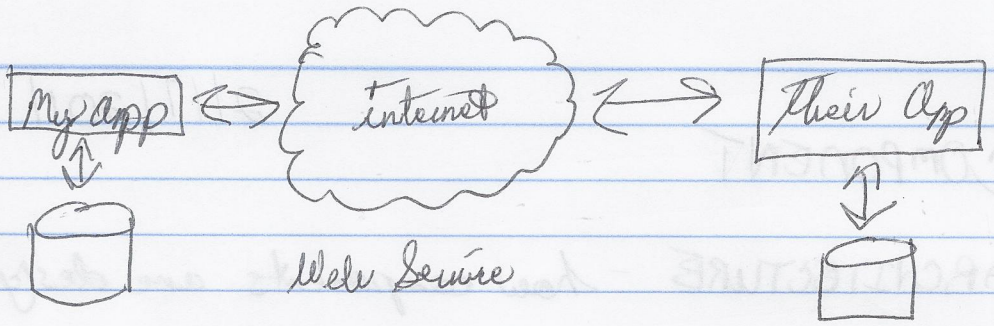


• could connect directly to database if not encrypted



Connection String = "myserver, username, passwd, mydb, mssql, engine"

provider



- web service - opens some functionality over the internet; can be used by a browser or application
- Application Programming Interface - API

Class User
~
~
~
} members

Sub SaveUser()

- Google apps Script
= uses Javascript

- No Class

2/17/2011

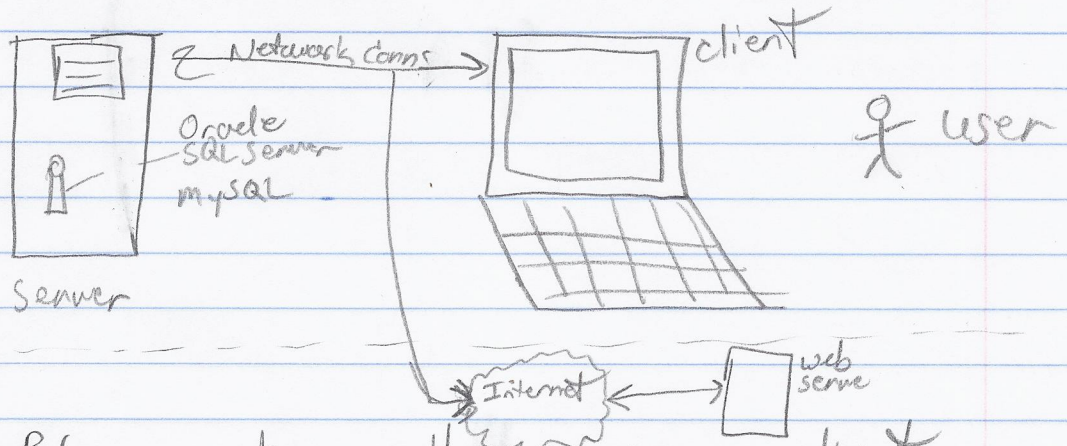
- COMPONENT
- ARCHITECTURE - how components are designed
- Framework - set of libraries
- .Net is a framework
- Mid-Term Thu (or Tue) before Spring Break
- namespace - simplifies organization of the components
- one exit point out of a function; otherwise might cause spaghetti code

HW
- new project within solution
- new class library
- build the class library
- multiple classes within a dll

2/22/2011

Midterm - March 10, Thursday

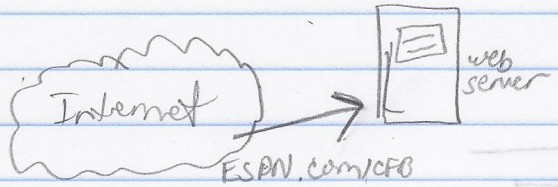
- in APIs the interface is a method
- component
- servers provide services
- " " things for other computers



eBL can be on the server or client

Web Applications 2/22/2011

Web server - Apache ~ Java, open source
IIS ~ MS



HTML
- formatted other documents, text
- allowed linking to other files/documents

- Client-side scripting
  - Javascript

- server can't do server side scripting

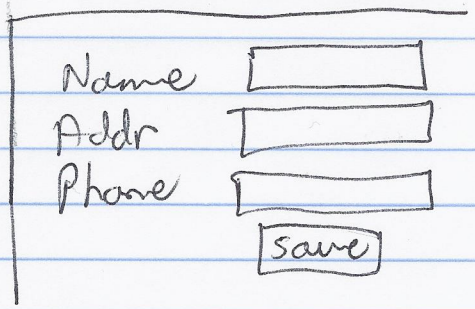
- application server ~ Tomcat, IIS
  - produce webpages
- database server (DBMS)

- server-side provided dynamic content

2/24/2011

- tags
- nested elements

do a wireframe



CSS 3/1/2011

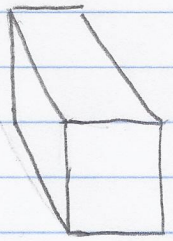
- HTML – functionality
- CSS – style

</script>

Javascript
- client-side (browser) scripting
- performance – slower

Java Example

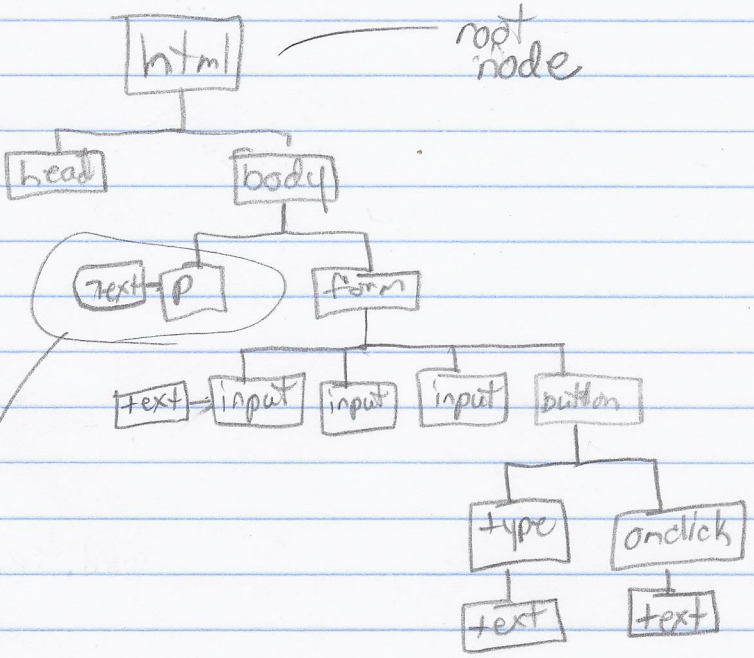
3/3/2011



- Reviewed CSS and Javascript
- more Javascript

- Document Object Model (DOM)
  - core DOM
  - HTML "
  - XML "

i.e. document.form.element.value



document.body.p.text

paras = document.getElementsByTagName("p");
paras[0].innerHtml

```
function change(){
  var alertp = document.getElementById("Alert")
  alertp.innerHtml = "Storm coming"
}
```

```
<p id = "alert"> Welcome </p>
<input type="button" onclick = "change()" value="Change" />
```

Test 1 Review 3/8/2011

~~3 essay questions~~

Format

- short answer (10-15 questions)
- tracing a program ~ interview, quality, VB or pseudocode
- hand written code
- design scenario - design documents (control list, class definition, process flows, event lists, etc)

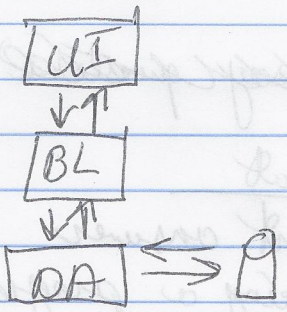
- everything up to dlls will be on the test
- ~~comp~~ component definition
- What is a dll? What are they used for
- DLLs allow you to distribute components of code
- coupling, cohesion - not specific levels
- overloading class constructor
- servers - physical vs logical architecture

  machines and wires between them — coding and logic

- centralized applications - pros and cons
- three tier architecture - ~~pros~~ pros and cons, customer effects
- frameworks

- framework vs architecture → way app is broken up

code that someone else has written with a set of components to do things



- study notes and assignments
- get example code from elearning

March 22, 2011

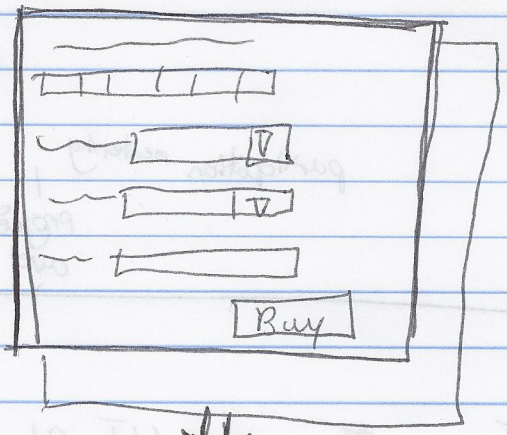
participation mentoring, midterm project cvg

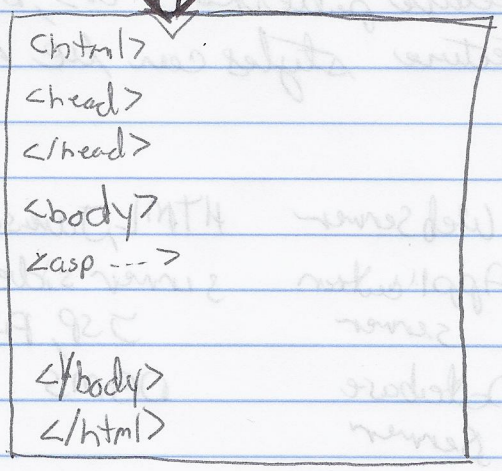
- 3 architecture patterns - UI, BL, DA (Persistence)
- architecture styles can be mixed

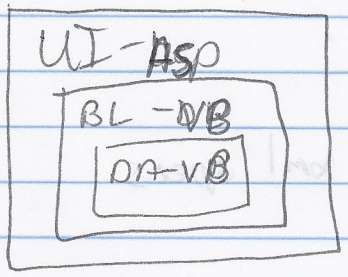
Web Server — HTML, Javascript, etc
Application server — server side scripting JSP, ASP, PHP
Database Server — DBMS

index.html
index.aspx — asp with xml specs

index.aspx







- could have multiple BL layers within UI

Final Project

- due April 29, 2011 @ 11:59pm
- charter and ERD due March 29 @ 11 AM
- no late submissions

- Shane's Classic Car Imperium
- website
- community around classic cars
- auction or sale / bid or buy
- create an account
- post car for sale or auction
- info about car = make, model, condition, mileage, etc
- anybody should be able to look
- forum, required to login
- sortable catalog / searchable
- payment information
- no credit card info in db
- receipt for payment
- notification about car being sold
- rating system for seller/buyer (optional)
- no special treatment for dealers
- many cars as ~~possible~~ wanted
- confirmation emails for ~~buy~~ buyer and seller
- JSP app using Spring Framework - Spring Source
- will design database,
- JDK, Spring ~~Source~~ Board, tool kit
- IE7+, Firefox 3+, Chrome, Safari compatible
- no auction longer than 10 days from start date
- low percentage for operating costs
- ~~negotiable~~ reserve price

Hervé's Classic Car ~~Emporium~~ Emporium

HLRs
- View Catalog of ~~cars~~ car $
- Register Users
- Bid or Purchase
- Test cars for sale or auction/Buy now
- search
- forum
- login
- notifys system
- Reporting

- Due Tues
- Team's Charter
- " ERD

- Charter
  - Project background
  - HLB
  - deliverables - what will be done
  - timeline
  - communication plan

- ERD
  - only submit final draft
  - listings, users, cars, relays,
  - include assumptions

---

user form
- name
- email
- password
- birthday (13 Days)
  18 sell or buy

business (include form)
- bill addr
- shipping "
- phone cell/home

bid
- who
- when
- what

listing
- car
- owner
- auction / sale
- keep track of bids
- start time
- end time

cars
- color
- make
- model
- miles
- VIN

- carfax
- description
- transmission/engine
- rebuild state
- price
- owner