CS 3802

Levi D. Smith CS3802 Intro to Software Engineering

CS 3802 - Introduction to Software Engineering

Section A - Fall Semester 1999
Tuesday & Thursday 4:30 - 6:00 PM

OVERVIEW

- General Information
- Syllabus
- Grading Policy
- Project Information
- Project Groups

INSTRUCTOR: Allison Elliott Tew

- Office: 113 College of Computing
- Phone: (404) 385-0595
- Email: allison@cc.gatech.edu
- Office Hours: Open door policy and by appointment

TEACHING ASSISTANT: Shanda Harper

- Email: shanda@cc.gatech.edu
- Office Hours: Monday, Tuesday & Thursday 1 - 2
- Office Hours Location: 104A College of Computing

TEXTBOOKS

- *Software Engineering: A Practitioner's Approach*
  Author: Roger S. Pressman
  Publication Information: Fourth Edition, McGraw-Hill, 1997
- *The Mythical Man-Month: Essays on Software Engineering*
  Author: Frederick P. Brooks
  Publication Information: 20th Anniversary Edition, Addison-Wesley, 1995

GRADING POLICY

Individual grades for the course will be based on the following: homework assignments, exams, group project work, and class participation. Students taking the class on a Pass/Fail basis will be required to earn a final letter grade of C or better to receive a passing grade.

ACADEMIC HONESTY: All students are expected to maintain standards of academic integrity by giving proper credit for all work. All suspected cases of academic dishonesty will be reported and pursued.

- Georgia Tech Academic Honor Code

ATTENDANCE POLICY: Unexcused absences from the midterm exam, final exam, design review, or project presentations will result in an automatic failure (F) of the class.

All assignments are due at the beginning of class on the due date, unless otherwise specified. Assignments will be accepted up to 24 hours late, with a 10 point late deduction. No assignments will be accepted after 24 hours.

CATEGORY	PERCENTAGE	
Homework	**15%**	
Homework #1 - Modeling Exercise	5 %	73
Homework #2 - Design Review Worksheet & Poster	5 %	—
Homework #3 - Design Review Reflection	5 %	—
Midterm Exam	**10%**	78
Project	**50%**	
Preliminary Problem Analysis & Project Plan	5 %	52
Software Requirements Specification (SRS)	10 %	—
Design Document	10 %	—
Prototype	10 %	—
Presentation	5 %	—
Individual Contribution	10 %	—
Class Participation	**10%**	—
Final Exam	**15%**	—

Brooks 1, 2, 3, 4, 7, 10, 11, 15, 16, 17, 19
✓ ✓ ✓ ✓ ✓

SYLLABUS

Week	Date	Topic	Readings	Prepared Lecture		Deliverable
1	09/28	Course Overview Introduction to Software Engineering	● Pressman - 1 & 2 ● Brooks - 1	View	Download	
	09/30	Project Planning Project Introduction & Team Organization	● Pressman - 3 & 5 ● Brooks - 2, 3, & 7	View	Download	
2	10/05	Software Lifecycle Software Process Models	● Pressman - 4 ● Brooks - 16 & 17	View	Download	Team Organization
	10/07	Requirements Engineering	● Pressman - 10 & 11 ● Brooks - 10 & 15	View	Download	
3	10/12	Requirements Analysis: Structured Techniques	● Pressman - 12	View	Download	
	10/14	Requirments Analysis: Object-Oriented Techniques	● Pressman - 19 & 20	View	Download	Preliminary Problem Analysis & Project Plan
4	10/19	HOLIDAY				Homework #1: Modeling Exercise
	10/21	Software Design Concepts	● Pressman - 13	View	Download	
5	10/26	Midterm Exam				Software Requirements Specification (SRS)
	10/28	Return SRS & Midterms				
6	11/02	Software Architecture	● Brooks - 4	View	Download	
	11/04	Structured Design Techniques	● Pressman - 14 ● Brooks - 11	View	Download	
7	11/09	Object-Oriented Design	● Pressman - 21	View	Download	Homework #2A: Design Review Worksheet → Wed noon
	11/11	Design Review	● Design Review Worksheets			Homework #2B: Design Review Poster
8	11/16	Implementation Prototyping	● Brooks - 13	View	Download	Homework #3: Design Reflection
	11/18	Testing Maintenance	● Pressman - 16, 17, & 22	View	Download	Design Document
9	11/23	Forgotten Activities	● Pressman - 5, 6, 7, 8, & 9	View	Download	
	11/25	HOLIDAY				
10	11/30	*The Mythical Man-Month* Discussion	● Brooks - 19	View	Download	
	12/02	Course Wrap-Up		View	Download	
11	12/07	Project Presentations				Prototype
	12/09	Project Presentations				Prototype

Program

CS 38Ø2

Newsgroup: git.cc.class.cs38Ø2.

fosi.hotmail.ru

VC- 8.11/

fosi.home.dhs.org
fosi.xpage.htm
download.at/FOSI

WizFlow Flowcharter

Friday 10 @ 2 - Demo
Next Tuesday - Presentation

October 5, 1999

- Process - foundation - how - key processes - what needs to be focused on
- methods - specific things - HOWTOs
- Tools - automated support - (BOOST)

- Quality Product - flaw free
  - reliable
  - ease of use
  - does what its supposed to do.
  - adaptability
  - maintainable
  - portability (cross platforms)
  - efficiency
- Critical Quality Attributes
- Other Attributes
- HCI - stone story
- Robustness - takes abuse - random inputs

- CMM - Maturity Level - Sophistication of group
- process - constraints

process principles -

→ - requirements - Elicitation & Analysis
- Design - conceptual / detailed
- Coding

- Testing - unit & integration / System Testing
- System Delivery
- Maintenance

Boehm → Bane

- adds ~~risks to~~ thinking about risks to the model - impacts of risks

October 7, 1999

- Unrefrenced pronouns - watch out

Requirements

- Functional - what you must do
- Non-Functional - User Interface, things not set in stone
  - System Requirements
  - Performance Requirements
  - Amount of Information
- Strengths - how much stuff can you do??

  Must / Shall - Jobs the system must do,

  Should - Important Features, but not necessary to have it

  Will - extra features

- Brainstorm

  Taco Bell

  - The system must keep a total for order
  - The system must calculate tax.
  - X. The system must know how much money is in the register
  - The system be able to open the drawer
  - Must - keyed input
  - Must - recognize cupons
  - Must - print receipt
  - will. ~~Must~~ - other cupons
  - Must - print receipts

Ellicitation - Gathering Information

How

Questionaire / Surveys - Only get answers you want
- Reach more people

Brainstorm / Meetings

Observe - You get to see what they do
- people act differently

Interview - Qualitative and Quantative Feedback
- Requires interviewing skills

Ethnography - work at the company
large investment of time

~~Review Software~~

Business Plans -

Review Internal / External Documents - lot of time to look through documents
- May not be Accurate

Review Software

Organization

- business plan
- org chat
- meetings

Existing Systems

- Software
- targeted interviews (techies)
- talk to ther users of the system

Processes

- *Internal Documents
- Ethnography
- Observation
- ★ Interviews / Meetings

Improvements

- Surveys
- interviews / meetings (with key people)

Requirements Analysis

- age range and educational goals

s • include requirements, design etc in schedule

s • include "hardcoded" dates in schedule.

Requirements Specification / Requirements Definitions (verbal shall/must/will)

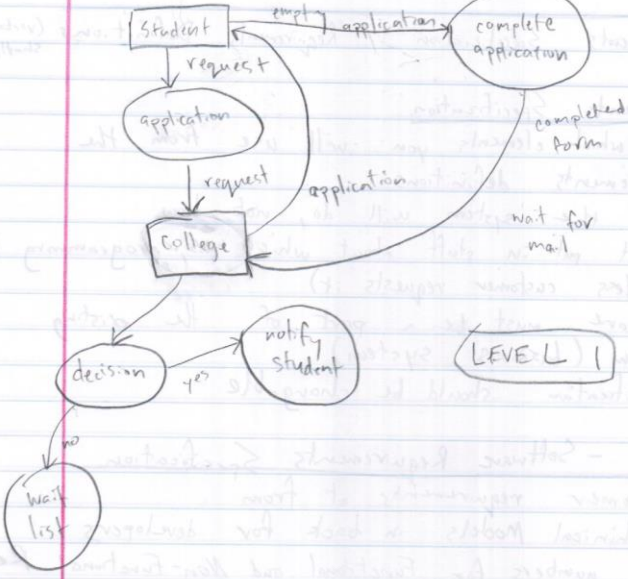
Requirements Specification

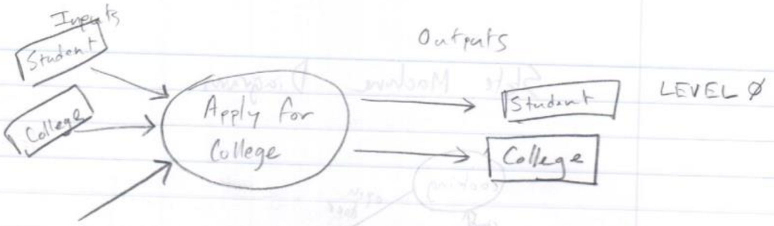
- pick which elements you will use from the requirements definitions
- what the system will do, not how
  - don't put in stuff about which OS, or programming language (unless customer requests it)
- software must be a part of the existing system (business system)
- specification should be changable

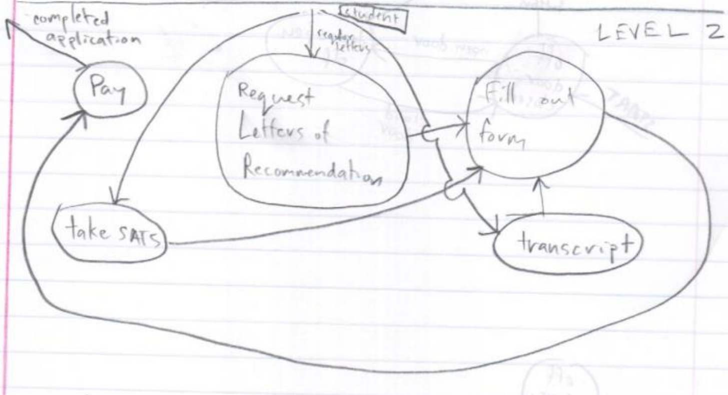
SRS - Software Requirements Specification

- Customer requirements at from
- Techinical Models in back for developers
- Use numbers for Functional and Non-Functional Req
-
- ellicitation ↓ gathering info   analsys ↓ using info
- Appendix - Surveys, Interviews, etc go here
- Viewpoint Oriented Analysis
  - look at everyone's perspective
  - conflict resolutions

Data Flow Diagrams- how data is changed in the model



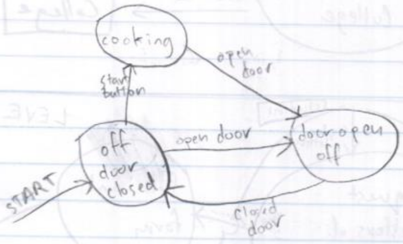


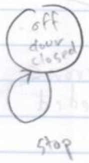


- make sure to label all arrows

λ lambda transitions - not neededed

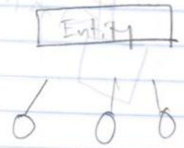
State Machine Diagram





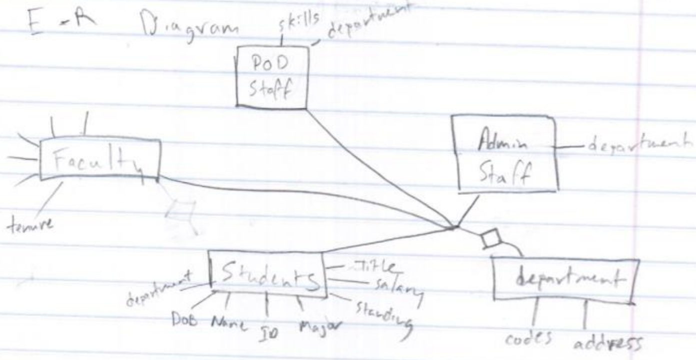
October 14, 1999

Entity Relationship Diagrams



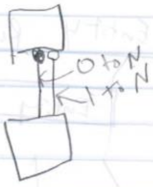
- Data Object

E-R Diagram



Object Oriented Analysis

- Use Cases ⟶ Scenarios



October 21, 1999

Midterm

- Chapters
- Ch 3 + 5 - Project Planning (no chart drawing - know general project planning)
- basic definitions - know how to explain terms
- explain concepts
- 10 points - book stuff
- what modeling / analysis would be best for a situation
- know how to use diagrams, not memorize them

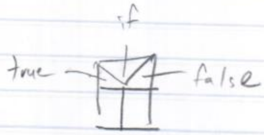
リバイ・ド・スミス

HW#1 Avg 68,08 My grade = 73
Midterm 79,43 My grade = 78

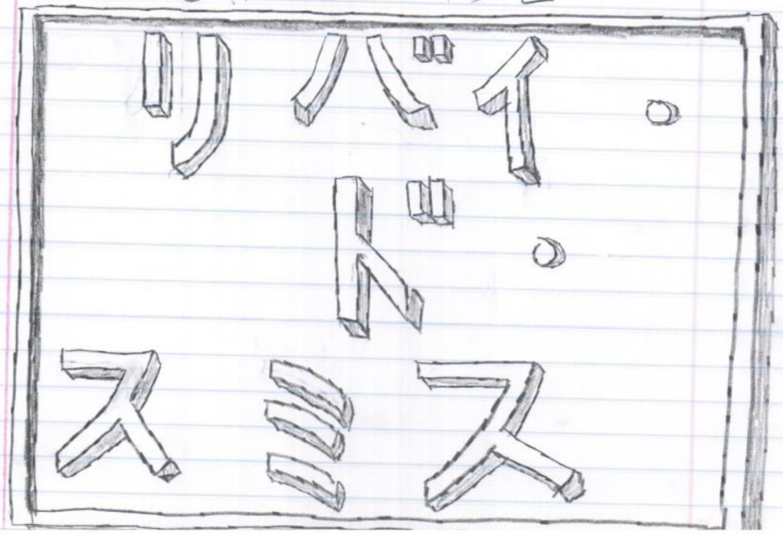
II7) A. Rapid Development
D. Misuse - avoids design, etc,

Software Design -
Engineering or Art?

Software Design Model
General Design Guide line



We will probably use PDL for design
Software Procedure - PDL



Nov 9, 1999

Due tomorrow
- What game is about
- Design Decisions
  - already made
  - not OO decision

already made
- modification to requirements (errors/omissions)
- new system model
- how game ends

pending
- modules/functions
- how will the game be implemented
- speed vs time

Conceptual Design
- functions of the game
- customer's language
- introductions
- how it will work
- requirements revisited
- present revised requirements

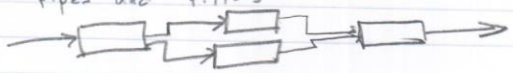
Technical Design
- how to implement requirements
- data structures
- modules

architectual design - ~~information~~ data - process design

modular decompositions - break down the "big picture"

event oriented decomposition - events that could happen

architectual styles
- pipes and filters

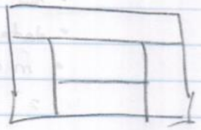


- Classic Pipe and Filter is similar to a compiler
- Layering - functions mainly interact with functions above and below it,
  - may be unefficient
- Repository - database
  - repository has control of all sources
- Interpreter
- Process control -

  Feedback loop -

  Feedforward loop - many variables could act on one observed variable

Architecture control Diagram



→ orders models

eg 259

CS 3802

Levi D. Smith

November 18, 1999

- Overview - what you're about to read
  - what you're going to learn - how
- Revised Req - things we've changed
  - explain why we changed/added requirements
- System Model

  Data, Funct/Process, Control behavior Model

  OO - Obj Oriented Model + CBM

- Conceptual Design
  - how game is put together
  - diagram
  - functional allocation of methods

- Detailed Design

  Data - what data types will be used

  Procedural - what do the functions use.

  Psuedocode - "sort", "First of List"

  - AI and complicated stuff needs to be more detailed - ★★★ Russcal ★★★★★★

- User Interface - does interface support all functionality

- Validation

  mapping between req and modules

  check off requirements

  use a table to connect methods to requirements

Testing - white box - black box??
- overview

Sources -

Appendix - Original Requirements

- Diagrams and Explanations
Book Boxes -

Black box - can't see what's inside
- puts out the correct answer

W.B. - does everything inside work correctly

Avg 60.25 Std 11.6

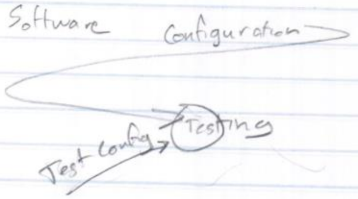
77 ↔ 41.5

Next
Transform vs transaction

- Questions

- Presentation - Slides (PowerPoint) (Tuesday)

December 2, 1999



impossible to exhaustively test a system

- Black Box
  - Equivanence partitioning
  - Boundary value analysis
  - error guessing
  - Cause - effect graphing - mapping errors

{ put in input and see if it gives the right answer

- White Box
  - Logic tests - examine code segments are executing
  - Mathematical proofs - formal proofs (for important stuff)
  - Cleanroom testing - work in a perfect environment
    - only OS and only tools you need are present

must have correct semantics

Processing Strategies
- Top Down - Logic has more attention
- Bottom Up - Good Fundamental pieces of code - so top logic *should* be right

CS 3802 - Introduction to Software Engineering

Modeling Exercise - Homework #1
Fall Semester 1999

DUE: 21 October 1999

DESCRIPTION:

Construct a Data Flow Diagram and Entity-Relationship diagram for the following system.

- The Data Flow Diagram should contain at least 2 levels. You must include Level 0 and Level 1 diagrams. You should include as many additional levels as needed to depict all of the functionality.
- The E-R diagram should include entities, attributes, associations, and attributes of the associations if appropriate.

Bonus points are available for modeling significant additions to the features of the system description listed below. The professor and TA have discretion as to what constitutes a significant addition.

Allison's Humane Society

Allison's Humane Society information system should support all of the following activities of the Allison's Humane Society. The humane society does not participate in euthanasia of pets. All pets will be sheltered until adopted.

Membership & Fund Raising

- Maintain "Member List", which includes name, address, phone number, length of membership, dues, total contributions.
- Process new Membership Applications
- Send a newsletter to all members on the first of each month highlighting the shelter's success stories and upcoming events. The newsletter should also include volunteer opportunities as well as any special needs of the shelter.
- Advertise by sending brochures to prospective members from a "Mailing List", which includes names and addresses.
- Generate Fund Raising Drive mailings, which should be sent to both current and prospective members
- Maintain "Donations List" - name, address, amount of donation, date of donation
- Prints advertising brochures, applications, fund raising letters, and donation forms.

Pet Adoption

- Maintain a "Pet Inventory" - all pets in the shelter (name, ID#, breed, color, age, size, health information, behavioral history, previous owner)
- Maintain a "Current Pet List" - pets in the shelter available for adoption. This is not the same as the Pet Inventory, but it includes the same type of information.
- Process Pet Adoptions - receive pets for adoption, track pets stay in the shelter, store Adoption

Record when pet is placed in a new home

- Maintain an "Adoption Record" for each pet - pet ID#, new owner information (name, address, phone number, references)
- Maintain an "Adoption List" - history of all pets placed in homes through the shelter

Note: Maintain means adding, deleting, and updating records.

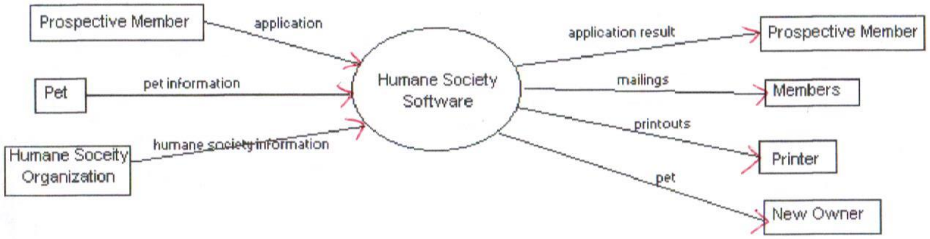
Note: The lists of attribute information are intended to be representative of the data you need to store Additional fields may be needed.

Back to CS 3802 Home Page

Levi D. Smith
October 21, 1999
CS 3802

73

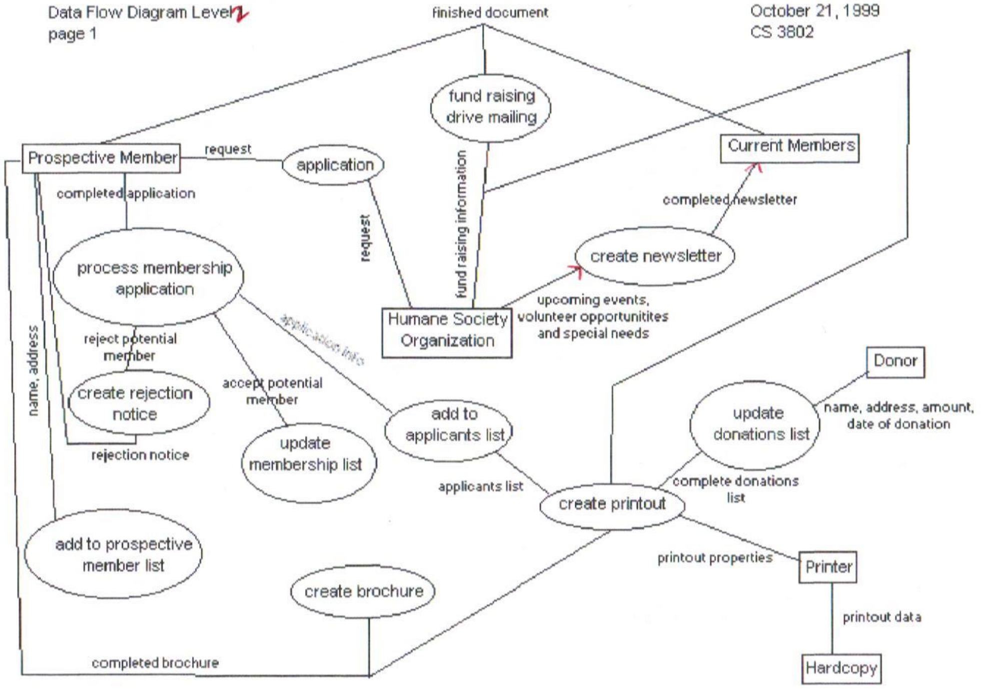
Data Flow Diagram Level 0



13/15

Data Flow Diagram Level 2
page 1

Levi D. Smith
October 21, 1999
CS 3802



This is level 2, not 1

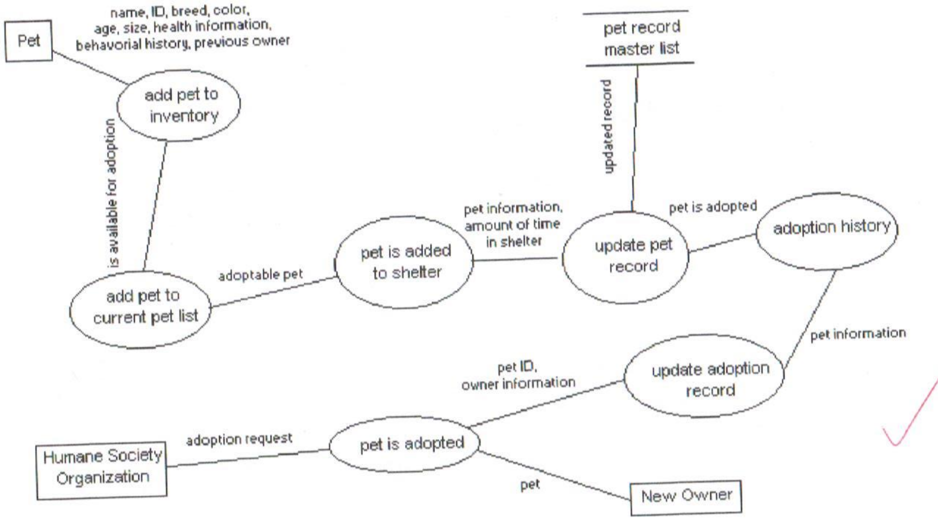
No Arrows -2

No level 1 diagram
-20

22

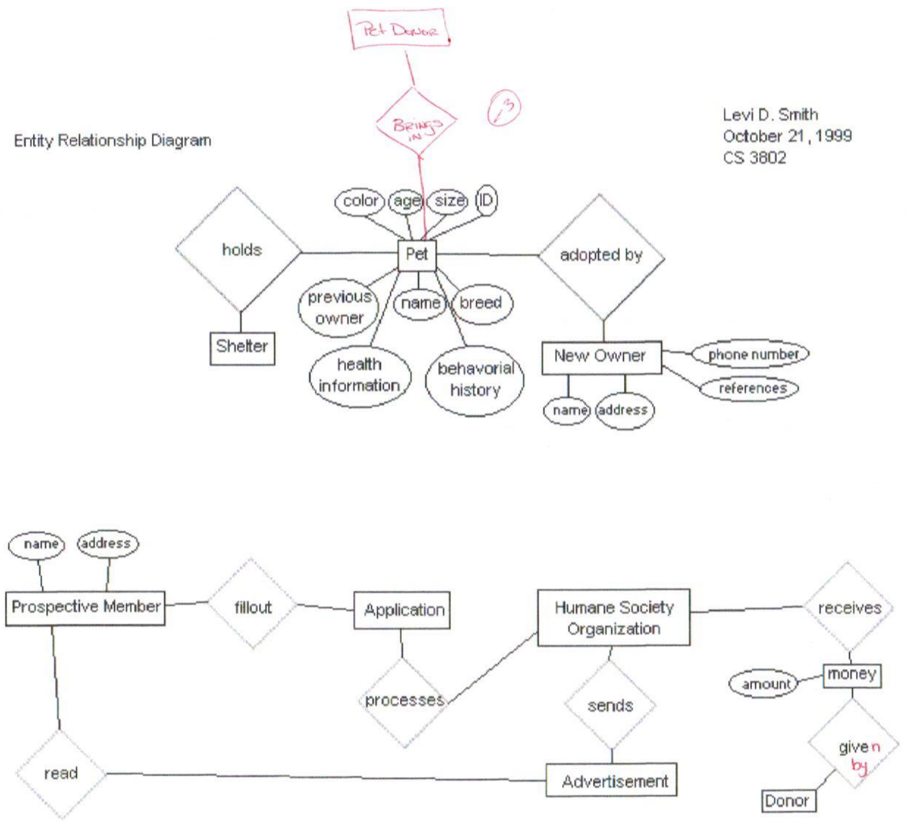
Data Flow Diagram Level 2
page 2

Levi D. Smith
October 21, 1999
CS 3802



Entity Relationship Diagram

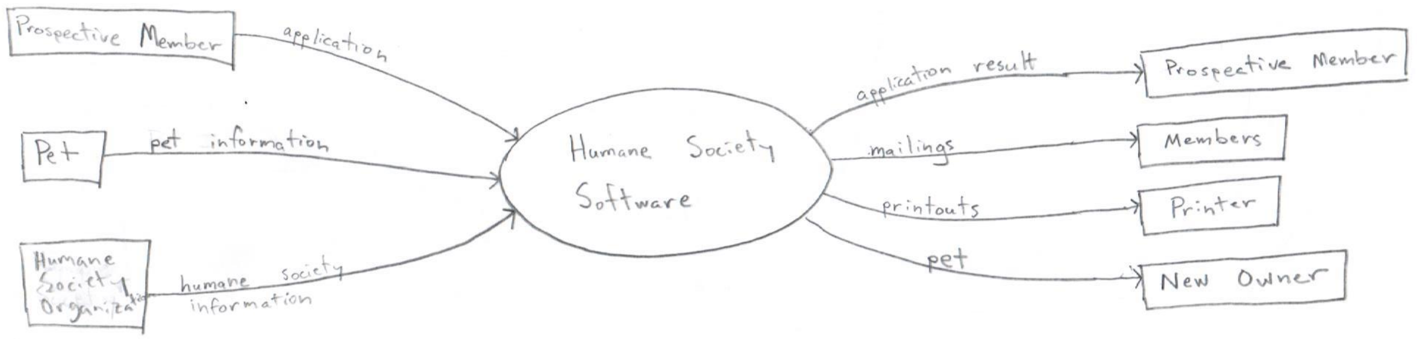
Levi D. Smith
October 21, 1999
CS 3802



-3

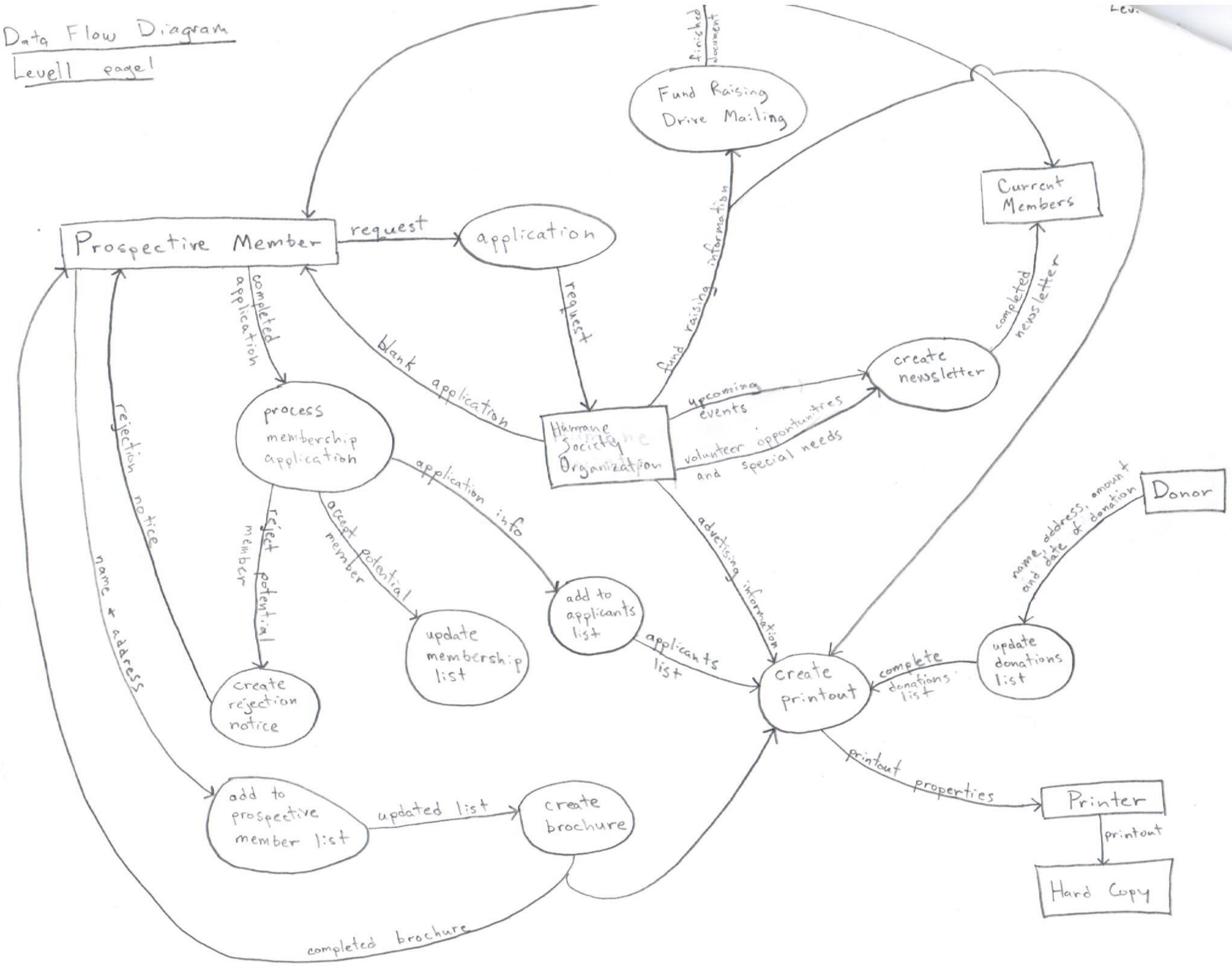
Levi D. Smith
October 21, 1999
CS 3802

Data Flow Diagram Level Ø



Data Flow Diagram

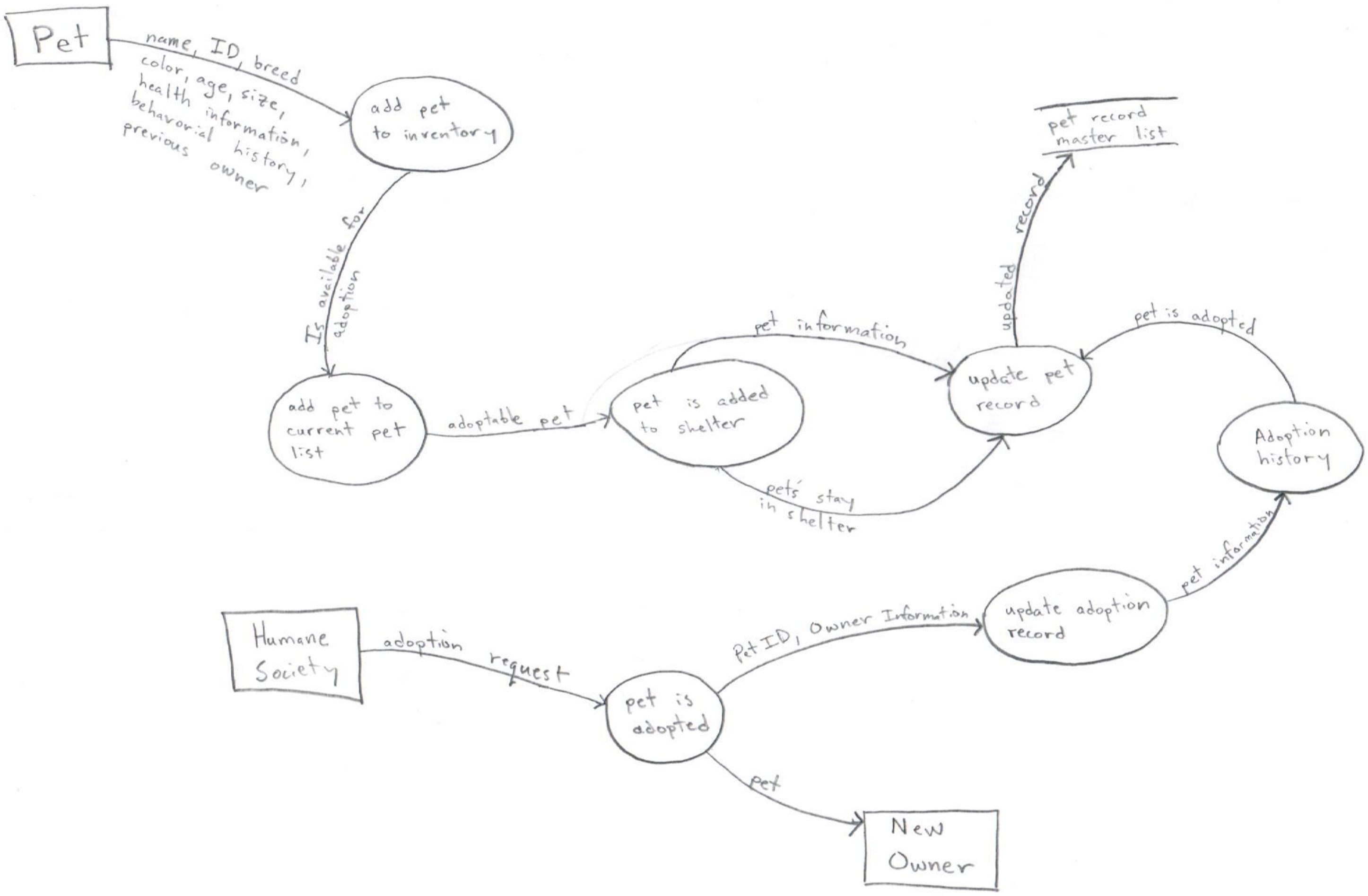
Level1 page1



Data Flow Diagram

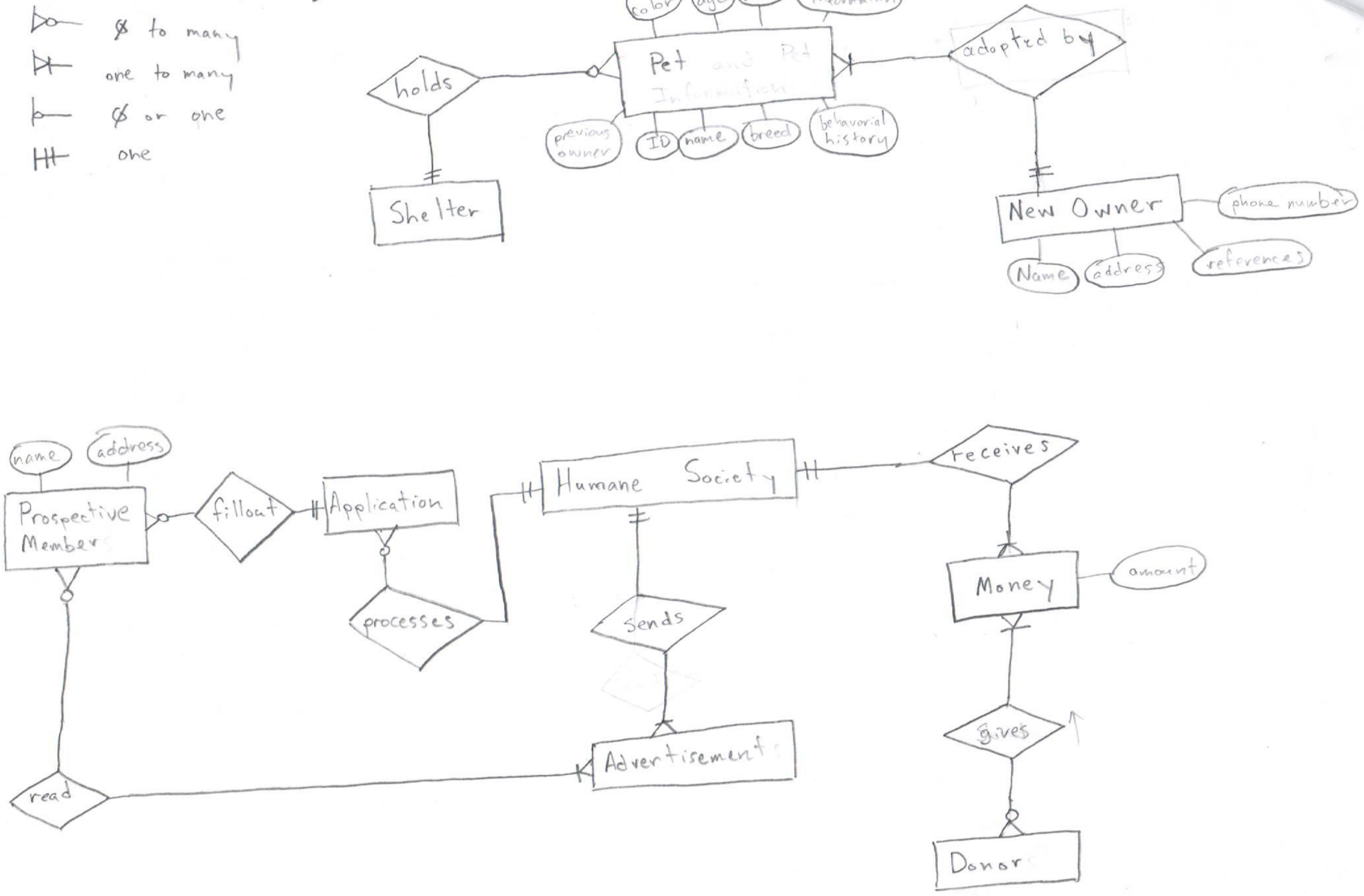
Level 1 page 2

Levi



Levi D.

Entity Relationship Diagram



CS 3802 - Introduction to Software Engineering

Design Review - Homework #2 & 3
Fall Semester 1999

DUE:

Homework #2 - Design Review Worksheet: 10 November 1999 at noon

Design Review: 11 November 1999

Homework #3 - Design Review Reflection: 16 November 1999

GOAL:

The purpose of this assignment is to prepare for and participate in a Design Review. A professional software engineer must be comfortable giving and receiving critiques of their designs. The Design Review process is designed to help you develop and hone these skills. The design review also allows you a chance to ask questions about and receive feedback on your design so that you can make adjustments, corrections, and changes before the Design Document is due.

Homework #2 - Design Review Worksheet

DUE: 10 November 1999 before 12:00 PM (Noon)

GOAL:

For a Software Design Review to be successful, all participants must come to the review meeting informed and knowledgeable about the system they will be commenting on. The goal of the Design Review Worksheet is to provide a synopsis of your project so that the Design Review participants can become familiar with your project (and you with theirs) before the review session.

DESCRIPTION:

Group Component:

Write a 1-2 page synopsis of your design project. Be sure to include the following information:

- Design Project Overview: A short (~ 5 sentence) description of the educational game you are designing.
- Completed Design Decisions: List 3 design decisions you have already made, and provide the rationale (i.e. reasons) for each decision
- Future Design Decisions: List 3 design decisions that you are still struggling with/working on

TURNIN: E-mail a HTML version of the design project synopsis to allison@cc by 12:00 PM

Wednesday, 11 November, 1999.

Design Review

DUE: 11 November 1999

GOAL: The goal of the Design Review is to give and receive feedback on the educational game you are designing. The review serves two purposes: to provide experience in giving and receiving technical designcritiques and to provide feedback on individual design projects that can be incorporated into the final DesignDocument deliverable.

DESCRIPTION:

Individual Component:

- Read all of the Design Review Worksheets, which will be posted to the class web-site by Wednesday, November 10, 1999.
- Bring to class a list of 5 comments about the design projects described by the Design Review worksheets. (Be sure to include comments about projects other than those that you have worked on or are currently working on.)
- In class, circulate among the design projects, asking questions and providing critiques and other information about your classmates design projects.

Group Component:

Prepare a poster about your design project that displays information that you would like to talk about. It would be reasonable to include some set of the following information, although feel free to include other kinds of information that you would like to discuss with your peers and colleagues:

- General design project overview, probably best displayed with a storyboard/screen shot of the general game, or a brief statement (i.e. a couple of sentences) about the game.
- Some representation about the design decisions you have already made, with accompanying logic, where appropriate;
- Some representation that easily conveys the design decisions that you are still struggling with

  Note: It is important that you spend some time thinking about which aspects of your design project you wish to discuss, and figure out a way to present that information so that it best encourages discussion. I am expecting more than 3 PowerPoint slides with the information turned in for Homework #2. However, I will allow a good deal of flexibility when it comes to how that information is presented. Do keep in mind that you are producing artifacts for public viewing and discussion and that the information should be large enough and clear enough for a group of people to see and discuss it.

- Each group should provide a notebook where individuals can leave design comments
- Each group should make sure someone from their design team is with their project at all times, althoughthat person should change through the design review session so that everyone has a chance to visit other design projects.

TURNIN:

Individual Component: Typed list of comments about design projects, due at the beginning of class. (You need to bring two copies of this, one to turn in to me, and the other to keep with you as you circulate around the room.)

Group Component: Design Review Poster to be displayed in class

---

Homework #3 - Design Review Reflection

DUE: 16 November 1999

GOAL: Part of the overall educational experience is making sure that we take a moment to look back at what we have done and see what we have learned from our experience. The goal of the Design Review Reflection is for each member of the design team to inventory of what they have learned in the Design Review process.

DESCRIPTION:

Individual Component:

- Read all of the comments that we offered to your group during the Design Review
- Write a paper about what you learned through the design review process. You should include:
  - What you learned about your design
  - How the comments and feedback effected your design (i.e. were design decisions reinforced, changed, was a different perspective examined that you hadn't looked at before, etc.)
  - You may also include a discussion of the design review process itself - not a critique of the process alone, but relate the experience of participating in the design to things you learned. (i.e. "I realized that having to prepare a poster helped us finalize some design decisions that we had postponed". Or "Talking to other teams about their design decisions helped me realize that we had forgotten to think about how XYZ relates to our design project.")

TURNIN: Typed discussion paper due at the beginning of class

Back to CS 3802 Home Page

B

Levi D. Smith
November 11, 1999
CS3802
Design Review Comments

Kid's Chess: Buttons do indeed have properties, such as a label, and foreground and background colors. The Button class is essential since it will notify the other classes when an action (such as a click) is performed on the button area. The Cell class is a good idea, but the Base class could just simply be a boolean variable in the Cell class.

Math Invasion: Instead of one linked list to hold all objects on the screen, maybe there should be a linked list of each type of object. For example, a linked list of aliens and another linked list of bombs. ✓ How are getting the same old problems over and over again going to be beneficial to a child? The child would eventually memorize all of the answers that would leave to replay value to the game. The randomized questions would allow the child to learn more and give the game much more replay value.

Mind Trial: Why should the GamePiece and Player objects be combined? A Player has a name, score, color, and a GamePiece. On the other hand, a GamePiece should keep track of its position and how many pie pieces have been collected. The game should run at least 640x480, but with accelerator cards the game may be able to be run in higher resolutions with added effects. The graphics accelerator card would only be an optional feature, and not required to play the game.

Monster in Shapeworld: This game could be expanded so that the player could "eat" numbers, words, and other things besides just shapes and colors. Once the child learns all of the shapes and colors, the child will have no reason to play the game anymore.

good, but outside scope of game

The Stock Market Game: First, will all players be given a set amount of money at the beginning of the game? Will the amount of money the player has at the beginning of the game depend on the difficulty level? Also, will this game read data from an Internet database to get stock quotes on real companies, or will the stock quotes be randomly generated by the program?

B-

Levi D. Smith
November 16, 1999
CS3802 Homework Assignment #3

The review process allowed our team participate with other groups to find the strengths and weaknesses of each team's design. The comments offered during the design review allowed our team to realize which aspects of our design need to be strengthened and which elements of the design do not require as much additional attention.

First, the design for our program needs to be improved so that the player has the ability to move about the galaxy freely, instead of moving in one straight path through the planets. The linear path would become dull and repeated after playing the game a few times. Therefore, since the exploration factor will add much more replay value for the child playing the game, our design will be modified to include this new way of traveling throughout the game. Other people commented on the fact that the presentation of a children's game, which includes factors such as graphics, multimedia, and the user interface, is much more important than the actual speed of the game. More media, such as movies of the planets as they are visited, should be added to our design to give the game a more interactive feel. Finally, I realized that the game must have a randomization algorithm intelligent enough so the same questions will not be asked to the child repeatedly. One idea was to have a tag on each question in the bank which would be marked true if the question has already been asked. If a question is tagged, then it will not be asked again until all other questions have been asked. Another approach we had considered is have a list of unasked questions where a question would be randomly picked from the unasked list. After a question is asked, it is then removed from the unasked question list and added to an asked question list. No questions from the asked question list will be used until the unasked question list is exhausted. At that time, all questions in the asked question list will be transferred back to the unasked question list.

The design review process has been helpful in many ways. I gained experience in presenting a software design to other people who are not a part of my team. Constructing the poster for our design made our team realize what the team members felt were the most important parts of the game. Making screen shots allowed the team members to come to a consensus on how the end product should look and feel. Visiting other teams allowed us to find which aspects of the game other teams felt were the most important. In almost all cases, the other teams stressed that gameplay and simplistic controls were the most important factor when creating children's game.

CS 3802 - Introduction to Software Engineering

Group Project - Fall Semester 1999

PROJECT GOAL:

As a team, develop a software application through the entire development life cycle. This means performing requirements analysis, design, implementing a functioning prototype, testing and documenting the system

TEAM ORGANIZATION:

Sun 5pm
Student Center 2nd Floor

Each team will consist of four (4) students. You may choose your own teams, although teams may be adjusted to accommodate all the students in the class. One team member should be chosen as project lead for organizational and contact purposes, but all team members are expected to contribute to all components of the project.

PROJECT SWAP:

In the real world, rarely are the same individuals involved in the entire process of product development from start to finish. To make your projects more realistic and to demonstrate the important roles that quality, communication, and documentation play in the life of a software engineer, each team will swap projects with another team after the Requirements Specification phase and then swap back to implement the project for which you analyzed the requirements.

DELIVERABLES:

DELIVERABLE	**DUE DATE**
Team Organization	5 October
Preliminary Problem Analysis & Project Plan	14 October
Software Requirements Specification	28 October
Design Document	18 November
Prototype	7 December
User Documentation	7 December

IMPLEMENTATION:

The prototype can be developed in any environment using any development or prototyping language. Although I would recommend using a standard prototyping development environment, such as Visual Basic, which the TA will be able to support.

PROJECTS:

Your assignment will be to write a simple, educational game for children. You may choose to implement your version of a popular children's game or you may invent a game of your own. (You may not select a game that has been previously developed as a project for this class: Concentration, States & Capitals, Early Years Math Tutor, Right from Wrong, GeoRacer, You Don't Know Math, Basketball Typing Tutor, Dragon Slayer, Kiddie Scrabble, Hangman.) The game should entail a moderate degree of complexity and should be sure to support a few clearly articulated educational goals.

CS 3802 - Introduction to Software Engineering

Team Organization - Project Deliverable #1
Fall Semester 1999

DUE: 5 October 1999

TEAM ORGANIZATION:

Each team will consist of four (4) students. You may choose your own teams, although teams may be adjusted to accommodate all the students in the class. One team member should be chosen as project lead for organizational and contact purposes, but all team members are expected to contribute to all components of the project.

DESCRIPTION:

Write a 1 - 2 page description of your team providing the following information:

- Team Name
- Names of all team members with preferred e-mail addresses
- Description of the roles of each member
  - You should include a description of each role
  - The name of the team member who is serving in each role
  - An explanation of how and why you developed this set of roles
- Three choices (in order of preference) for the educational game you would like to develop.
  - You may choose to implement your version of a popular children's game or you may invent a game of your own.
  - The game should entail a moderate degree of complexity and should be sure to support a few clearly articulated educational goals.
  - Be sure to include the name of the game, what you intend to be teaching (i.e. your clearly articulated educationl goals) and the target audience.

Back to Project Information Page | **Back to CS 3802 Home Page**

✓

Team Name
GUIS (Guys Under Intense Stress)

Team Members

Jamie Hobbs	gte435j@prism.gatech.edu
Duy Pham	duy@cc.gatech.edu
Chris Ingram	ing@cc.gatech.edu
Levi Smith	command@cc.gatech.edu

Team Roles - how did you come up with these?

Leader - He organizes group meetings and ensures that the project is finished by the deadline. (Levi)

Resource Manager - Keeping track of code and sending updated code to all group members is the job of the resource manager. Each team member must submit their code to him when their code is complete so he has the most recent version of the code. (Jamie)

Project Creators - They make initial design and requirements for the project. They decide which features will be in the project and how the project will be ~~implemented~~ designed. (All members)

Project Designers - The role of the project designers is to make the design documents for the program. They must decide the basic algorithm(s) to be implemented by the program. (All members)

Documentation Specialists - They write the user documentation. The user documentation will be an overview of the game, rules, and how to play the game. (Duy and Chris)

Program Debuggers - Testing all of the code is the job of the program debuggers. They must review all code and make sure there are no glitches or errors in the code before the deadline. (Jamie and Chris)

Game Engine Programmers - The engine programmers will code the core of the program. Their code should be written so it will easily work with the graphical user interface written by the GUI programmers. (Levi and Duy)

GUI Programmers - Making the graphical user interface that will be used with the program engine will be the job of the GUI programmers. The GUI programmers must meet with the game engine programmers to ensure that their GUI will operate with the program engine. (Jamie and Chris)

Game Choices

WordTris - A game similar to the classic Tetris, except the player must use falling three letter blocks to create words. When a word is created the blocks from the word are removed from the board. The game ends when the blocks reach the top of the game board. This game helps spelling capabilities of the user. A screen at the end of the game will display all words that the user correctly constructed and also display the definitions of the words, which will increase the user's vocabulary. This game would be aimed at children ages 8 and up.

Who Wants to be a Millionaire- This game will be similar to the ABC game show "Who wants to be a Millionaire." The player is asked various multiple choice questions, then as they answer the questions correctly they move up a ladder of money values, until they finally reach one million dollars. There are a few other options to the game, such as being able to eliminate two choices (50/50) once during the game and another option is asking the audience for help. The game would broaden the players scope of knowledge in a wide variety of subjects by answering the questions. The questions at the beginning of the game will be much easier than the questions at the end of the game, so the game can be played by people ages 8 and up.

Need to be
careful that
you aren't
only re-
inforcing the
memorization
of trivia

Trivial Pursuit - Players move a marker around a board and answer questions in various subjects. When the player answers a question correctly, he/she gets a slice of a circle with a color representing that subject. The winner is the first player who gets all slices and completes the circle. This game will teach players various facts in a wide variety of subjects. Trivial Pursuit would have questions at at least a sixth grade level, so the game would be for people ages 10 and up.

CS 3802 - Introduction to Software Engineering

Preliminary Problem Analysis & Project Plan - Project Deliverable #2
Fall Semester 1999

Student Center
4pm Sunday

DUE: 14 October 1999

GOAL:

The purpose of this assignment is to evaluate your progress during the requirements phase of the project before the Software Requirements Specification document is due. The paper you write should provide the instructor with an in depth view of where your team is in the problem understanding and requirements analysis activities. It also should outline your plan for continued progress throughout the quarter.

DESCRIPTION:

Write a 5 - 10 page description of your project providing the following information:

- Preliminary Problem Analysis
  - C Detailed Problem Statement (what is the problem that you are solving? what do you now understand about the problem that you didn't know when you were given the 2 - 3 sentence initial description)
  - J Complete definition of the Educational Game you are designing. Be sure to include and clearly articulate your educational objectives and intended audience.
  - Summary of Findings to Date (what information have you gathered? what issues have you addressed? what issues/items do you still need to gather more information about?)
- Project Plan
  - L Outline of future work, with schedule
  - L Work assignments (i.e. who on your team is doing what?)
  - L Description of the roles of each member, accompanied by a description of how and why those roles were chosen
- A Sources & Bibliography

GRADING CRITERIA

Back to Project Information Page | Back to CS 3802 Home Page

CS 3802 - Introduction to Software Engineering

Preliminary Problem Analysis & Project Plan - Grading Criteria
Fall Semester 1999

DUE: 14 October 1999

GOAL:

The purpose of this assignment is to evaluate your progress during the requirements phase of the project before the Software Requirements Specification document is due. The paper you write should provide the instructor with an in depth view of where your team is in the problem understanding and requirements analysis activities. It also should outline your plan for continued progress throughout the quarter.

GRADING CRITERIA:

Preliminary Problem Analysis (60)	
Detailed Problem Statement (20)	
Game Description (20)	
Summary of Findings to Date (20)	
Project Plan (25)	
Schedule (15)	
Division of Labor (5)	
Roles Revisited (5)	
Sources & Bibliography (5)	
Overall Document Quality (10)	
TOTAL (100)	

Back to Project Assignment | Back to Project Information Page | Back to CS 3802 Home Page

Mind Trial -- GUIs

Problem Analysis & Project Plan

Trivial Pursuit® is a game of trivial facts that is intended to educate its players on their knowledge of different categories of trivia. Our problem is to alter the game-play of Trivial Pursuit® to present itself as an effective educational tool. In order for our game to be able to accomplish this feat, it would have to provide both a fun atmosphere as well as a game that runs smoothly without any problems "popping up" during the course of playing the game.

The game:

-must display a game board.

-must provide tokens to represent each player.

-must present questions to the player.

-must keep track of the progress of each player.

-must be able to determine a winner.

-must be able to ask a question that is in the proper category.

-must be able to determine if the player has gotten the question correct.

-must be able to display a digital die that randomly determines a number to allow the player to move around the board.

-must be able to quit at any time.

-should highlight the possible moves for the player.

-should have proper difficulty settings.

-will have an option that will keep track of the top ten winners.

-will allow the players to alter the colors of the playing pieces.

While Trivial Pursuit® is designed to teach trivial facts, this game - Mind Trial - is a fun, efficient way to teach children of all ages pertinent information on a variety of subjects. The chosen topics span a wide field of knowledge and include History, Mathematics, Science, Geography, Grammar, and Current Events. The majority of these subjects are more or less constant, but a few, namely Current Events and Geography, can change. Therefore, updates could be sent giving the most current questions.

Different levels of difficulty are also planned. A novice level for those in grades 3-5, intermediate for 6-8, and an advanced level for high schoolers. The questions for each level will be of appropriate difficulty based on the setting. Mind Trial will be designed for 2-6 players. Having a one player mode would mean programming a computer player that would simply pick a random answer, and would therefore defeat the purpose. Each question will be a multiple choice problem with 5 possibilities. Mathematics however could have questions where the player would enter in the answer, since there is only one possible correct answer.

Summary of Findings to Date

Thus far we have gathered, mostly from the Internet, that Trivial Pursuit® is widely played, both on the web and offline, including but not limited to, the classic board game and computer based game. People of many varying educational levels play the game. Naturally, we cannot encompass such a wide range. Therefore, we have limited our "educational pursuit" to the middle and high school levels.

The original game itself is "trivial" so in order to make it educational, we have decided to emphasize our game on a few specialized topics taught in schools such as history, geography, grammar, and so forth. By doing so, our game can help students study for their quizzes, tests, exams, and what not by merely playing. It has been shown that kids learn faster when they are enjoying themselves. Thus, this game will help them do just that. Since this game is based upon the original Parker Brother's ever-so-famous Trivial Pursuit®, the rules and regulations are very much the same. Also, a copy of the original rules is attached.

Most of the questions on the original game are, how can we say, trivial? That is the essence of the game itself. Therefore, we will have to acquire a different set of questions and answers that would be suitable for a given subject rather than just using the ones from the game.

Work Schedule

October 14 - October 21 (8 days)

The team will begin to do background research the game and gather the tools necessary to create the game.

October 21 - October 28 (8 days)

During this period the software requirements specification will be created.

November 11 - November 18 (8 days)

The design documents will be written during this period.

November 23 - November 30 (8 days)

The game engine and graphical user interface will be coded during these eight days.

November 30 - December 2 (3 days)

During these days, the game engine will be integrated with the user interface. The debugging process will also take place at this time to ensure that everything operates properly.

November 30 - December 7 (8 days)

Construction of the presentation for project will be done these days. The user documentation will also be written during this time period.

Division of Labor

The background research will be handled by all team members. Levi and Chris will interview people to find out what people really want in an educational game. Jamie and Duy will search for books and documents relating to the creation of educational computer games.

All team members will take a part in the creation of the requirements specification documents.

Duy and Chris will write the user documentation. Duy will write the rules and regulations for the game. The technical documentation, such as hardware and software requirements, will be written by Chris.

Levi and Duy will program the engine for the program. Levi will be responsible for programming the game board, dice randomization and movement of game pieces. Duy's job will be to create a database of questions along with correct and incorrect answers.

The user interface will be coded by Jamie and Chris. Jamie will program the menu bars, buttons, and other interactive components. Chris will create the graphics for the game. Images of game pieces, the game board, and dice will be needed.

Jamie and Chris will review all code before the deadline to ensure the program is error free before submission.

Team Roles

Leader - He organizes group meetings and ensures that the project is finished by the deadline. Levi was chosen for this role because he has had previous experience of being a team leader.

Resource Manager - Keeping track of code and sending updated code to all group members is the job of the resource manager. Each team member must submit their code to him when their code is complete so he has the most recent version of the code. This role was given to Jamie because he spends the most time online which will allow him to receive and send code updates more often than anyone else in the team.

Project Creators - They make initial design and requirements for the project. They decide which features will be in the project and how the project will be designed. All members will need to give input on how the project will be created.

Project Designers - The role of the project designers is to make the design documents for the program. They must decide the basic algorithms to be implemented by the program. Every team member will need to share his opinion on how the project should be designed because. a design written by only a fraction of the team would be biased towards their personal preferences

Documentation Specialists - They write the user documentation. The user documentation will be an overview of the game, rules, and how to play the game. Duy and Chris chose to do the job of documentation because they are the best writers of the team.

Program Debuggers - Testing all of the code is the job of the program debuggers. They must review all code and make sure there are no glitches or errors in the code before the deadline. This job will be filled by Jamie and Chris because they can quickly read through code and find errors.

Game Engine Programmers - The engine programmers will code the core of the program. Their code should be written so it will easily work with the graphical user interface written by the GUI programmers. Levi and Duy have this role because they have the most game programming experience.

GUI Programmers - Making the graphical user interface that will be used with the program engine will be the job of the GUI programmers. The GUI programmers must meet with the game engine programmers to ensure that their GUI will operate with the program engine. Jamie and Chris chose to do this job because they are skilled in laying out interfaces and they have the most experience in making computer graphics.

Bibliography

www.cccvette.com/trivia-4.htm

www.chatgames.com

www.genusnet.com/friends/trivia.htm

www.newswire.ca/releases/November1997/06/c0721.html

www.ntn.com

www.trivialpursuit.com

CS 3802 - Introduction to Software Engineering

Software Requirements Specification - Project Deliverable #3
Fall Semester 1999

DUE: 28 October 1999

TURN-IN: 2 copies of your document should be turned in. One copy will be graded and the other will be given to your design team.

GOAL:

The purpose of this assignment is to prepare a Software Requirements Specification. The document should present the requirements and a system model for the educational game you are developing.

DESCRIPTION:

Write a Software Requirements Specification including the following information:

- Introduction
  - Project Overview/ Problem Statement
  - Document Overview

- System Model
  - Choose a model that is most appropriate for the educational game you are developing; your most likely options are Data Flow Diagrams, an Object Model, or State Machine diagrams. (Be sure that the model you choose is consistent with the analysis method you have chosen)
  - Provide a clean, electronic drawing of your model
  - Include explanatory text as needed

- Functional Requirements
  - Be sure to state your requirements clearly.
  - Include a numbering scheme that will allow traceability.
  - You may provide a brief design rationale for any requirement which you feel requires explanation for how and/or why the requirement was derived.
- Non-functional Requirements
  - Be sure to state your requirements clearly.
  - Include a numbering scheme that will allow traceability.
  - You may provide a brief design rationale for any requirement which you feel requires explanation for how and/or why the requirement was derived.
  - Also, remember to include system, performance, and usability requirements in this section.
- Validation Criteria
  - Requirements Validation: How do you know that you have the right requirements? What steps did you take to make sure that you had enough information? What steps did you take to make sure that you had correct information?
  - Testing Plan Overview: How will you make sure that your system is developed according

to the requirements stated? How will you make sure that your requirements are validated and verified throughout the development process? You do not need to go into specific testing plan strategies (i.e. white box, black box, etc.).

- Appendix
  - Methodology Analysis: What analysis methodology did you chose and why?
  - Elicitation Techniques: What elicitation techniques did you use? What information were you seeking from each technique? Also, include in this section any sample surveys, interview questions, etc. that you used.
- Sources & Bibliography

GRADING CRITERIA

Back to Project Information Page | Back to CS 3802 Home Page

MIND TRIAL
Software Requirements Specification

Submitted by GUIS
(Jamie Hobbs, Duy Pham, Chris Ingram, Levi Smith)

29 October 1999

Project Deliverable #3
for CS3802 Section A
Fall Semester 1999
Georgia Institute of Technology

Table of Contents

While some games entertain, Mind Trial is designed to both entertain and educate students of all ages. Mind Trial is modeled after Trivial Pursuit ®, the popular trivial game that is suitable for both the young and old alike. While the format and rules are the same, the questions will be quite different. The problem to be solved throughout the course of this project is how to make a classically "trivial" game into something that educates children on pertinent subjects while maintaining an amusing and/or diversionary atmosphere. Mind Trial, in its ideal form, would teach students Math, Science, Geography, History, Grammar, and Current Events while instilling them with a fun-filled experience.

The requirements that are necessary to the functionality of the game are listed below, along with a few details that would be nice to include in order to improve several aspects of the game. Some of these requirements include: game play, the questions asked during the course of the game, the options that are available to the user of the software, graphics and sound effects, the artificial intelligence of the game, the actual code, the marketing aspects, and the performance of the software.

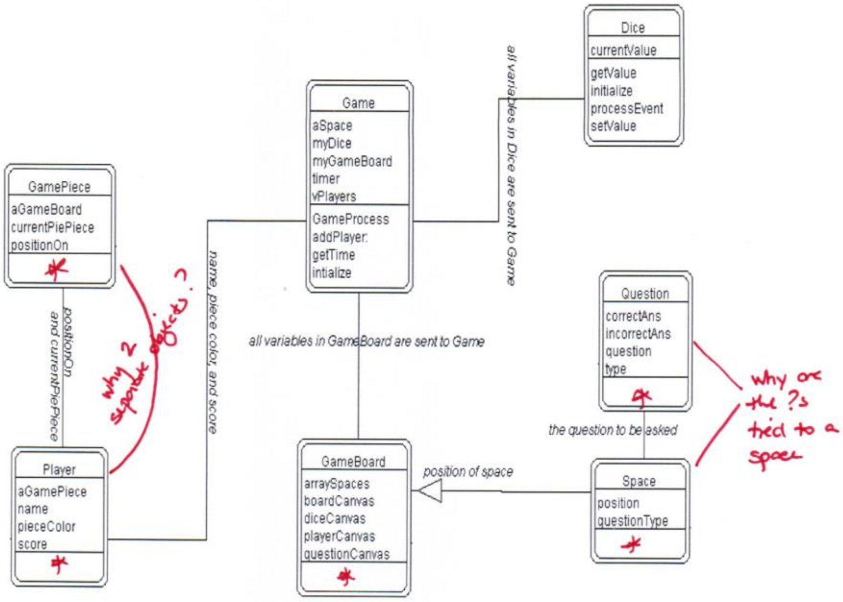
All of the data used in constructing the software, as well as its marketing strategy was checked for validity. This includes extensive background checks on all people interviewed, a double check on the pertinent data that was presented during the course of research, and a realization of the limitations provided by the surveys conducted.

An appendix is attached which includes some of our information that was found during the conduction of the research. A sample survey that was distributed locally is included, as well as a sample interview with an educator of small children. On the official website (http://www.prism.gatech.edu/~gte187k/mindtrial/) is a survey that several people have partaken in. This information, although with its limitations, has proven to be an invaluable source in providing information to possible expansion options that could be pursued in the future. The main option for upgradability discussed at this time is that there could be additional categories of questions available via download. This download could be an expansion pack or update to posted to the World Wide Web site, or a disk could be mailed to the person by simply mailing the parent company that is responsible for production of the game.

fragment

Due to the fact that an object-oriented language is easier to upgrade, translates into less time and money spent in upgrading the software. Due to its ability to be platform independent, Java was chosen as the most suitable language for Mind Trial. There are seven major classes (Game, GamePiece, Player, GameBoard, Dice, Question, and Space), each with their own attributes and services. The Game class will serve as the main program and hub of all of the communication between the classes. It will also hold most of the instances of each of the other classes. The only types of communication that occurs between each of the classes is a send/receive relationship, except for the communication between the Space and GameBoard classes. This particular communication is a part/whole relationship. For more details on these types of communication, see the glossary.

implementation decision



why 2 separate objects?

why are the ?s tied to a space

Figure 1. Above is the system model of the Mind Trial software. Notice the communication links and the class hierarchy structure. This layout will prove to be very useful if it is decided that Mind Trial should be expanded to include a large variety of questions.

* No methods?!?

still not sure how this is educational? are you explaining the right answer? is that reference material available?

Functional Requirements:

Game play Requirements:

F.GP.001 The game must display a game board to the user.

F.GP.002 The game must provide tokens to represent each player.

F.GP.003 The progress of each player must be recorded.

F.GP.004 A digital die must randomly select a number between and. including one and six.

F.GP.005 The user must have the ability to quit the game at any time.

F.GP.006 A game token must move to the space the user has selected, unless the user has selected an illegal move.

F.GP.007 The game must allow two tokens to land on one space.

F.GP.008 Available moves should be highlighted for the player.

then why not prevent illegal moves

?? **F.GP.009** The game board should be a circle of spaces with three lines of spaces spanning the diameter of the circle.

F.GP.010 A piece must be able to move to any adjacent space. - ??

doesn't the roll of the die determine where they can move?

F.GP.011 The player should be able to start a new game at any time.

F.GP.012 T~~he software must have the ability to determine the winner~~ when a player collects six pie pieces, they win the game

F.GP.013 A congratulatory message should be displayed to the winning player.

redundant w/ F.M.002

F.GP.014 The game should allow multiple players. how many

F.GP.015 The game should have an "About Box" which would contain the names of the creators of the game, the program version number, and other important information.

why? Nonfunctional at best

F.GP.016 The program must be able to display the rules of the game to the user.

F.GP.017 Help should be provided if the user presses the "F1" key. implementation

F.GP.018 The player will have the ability to save the current progress of the game.

F.GP.019 The player will have the ability to load previous game states.

F.GP.020 The player should be able to enter his name for future records, such as game scores.

F.GP.021 A list of correct answers will be displayed to the user when the game is over.

?not as they miss the ? - they have to wait until the end??

F.GP.022 A game demonstration should be available to show users how the game is played.

F.GP.023 When the game ends, the program should ask the player if a new game should be started.

F.GP.024 The game should have pull-down menus from the menu bar for many of the basic functions of the game. implementation

Question Requirements:

F.Q.001 Questions must be presented to the player after the player moves the token.

F.Q.002 Questions must be multiple choice.
F.Q.003 The player must have the ability to select an answer to the question presented.
F.Q.004 The type of question presented to the player must be in the same category as the space where the token lands.
F.Q.005 The software must be able to determine if the player has selected the correct answer.
F.Q.006 A timer should limit the amount of time a player can take to answer a question.
F.Q.007 Expansion packs should be available on the official website to add questions to the game database

Nonfunctional

F.Q.008 The question categories will include Grammar, Mathematics, Sciences, Current Events, and Geography.
F.Q.009 One hundred (100) unique questions should be available in each category.

- Not very many

F.Q.010 Answer choices for a question should be randomized each time a question is displayed.
F.Q.011 Players should have at least one minute to answer a question.
F.Q.012 A player must receive one pie piece when a question is answered correctly from a space which is an intersection between the circle and a diameter.

this would mean nothing if I hadn't played Trivial Pursuit

F.Q.013 The player must receive an extra turn when correctly answering a question from any non-intersection space.

what happens if they answer it wrong?

Options Requirements:

F.O.001 The player should have a choice from three levels of difficulty.
F.O.002 All winners should be recorded into a hall of fame list.
F.O.003 The player should have the ability to change the color of the token.

Graphics and Sound Effects Requirements:

F.GS.001 A bell will sound if the player answers a question correctly.
F.GS.002 A buzzing sound will play if a question is answered incorrectly.
F.GS.003 The game will use textured polygon models for the game tokens and die.

implementation

F.GS.004 The game should have a program icon.
F.GS.005 The default token colors will be green, blue, yellow, orange, red, and magenta.
F.GS.006 The game will have soft background music. - why?
F.GS.007 The user should have the option to turn music and sound effects off.

Artificial Intelligence Requirements:

F.AI.001 The game should have intelligent computer opponents who will make reasonable decisions depending on the state of the game..

What does this mean?

F.AI.002 Computer opponents should select an answer based on a percentage, which may be varied according to the difficulty setting.
F.AI.003 On "Novice" difficulty setting the computer opponent should select the correct answer 20 percent of the time.
F.AI.004 On "Normal" difficulty setting the computer opponent should select the correct answer 40 percent of the time.
F.AI.005 On "Difficult" difficulty setting the computer opponent should select the correct answer 60 percent of the time.

Multiplayer Requirements:

F.M.001 Players will have the ability to challenge other players across the Internet.
F.M.002 Up to six players should be able to play against each other locally.
F.M.003 Players will be able to electronically chat with each other during the game.
F.M.004 In multiplayer mode, the game must display the name of the player who currently has control of the board.

- why? over net? Locally?

Non-Functional Requirements:

System Requirements:

NF.S.001 The game must operate on the Windows 95 and Windows 98 operating systems.
NF.S.002 The game should be compatible with the Linux operating system.
NF.S.003 The game will operate on the Macintosh Operating System.

why not say platform independent

NF.S.004 The software must be able to run as a stand alone application.
NF.S.005 The software should have the ability to run as an applet on the World Wide Web.
NF.S.006 The game should run on computers with at least 16 MB of RAM.
NF.S.007 The size of all program files should not be larger than 16 MB.

why?

NF.S.008 The game should run in a window that is no larger than 640 pixels wide by 480 pixels high, which will allow the greatest number of users to display the program correctly.
NF.S.009 The program should use at least 256 colors
NF.S.010 The game will use graphics acceleration cards to improve the quality of the graphics.
NF.S.011 The game should require at least a 28.8 Kbps connection for multi-player games via the Internet.

all implementation

Coding Requirements:

NF.C.001 The game should be coded in the Java Development Kit version 1.1.8

NF.C.002 The code will be written with Hungarian notation.
NF.C.003 The code must compile with no errors
NF.C.004 The game will be coded on the UNIX operating system using the VI editor.

Marketing Requirements:

NF.M.001 Expenses must remain within the budget (see Figure A).
NF.M.002 The final product should have a suggested retail price of nineteen dollars and ninety-five cents ($19.95).
NF.M.003 The game will have an official website to promote the product.
NF.M.004 A demonstration version, with many features disabled, will be released.

Performance Requirements:

NF.P.001 The program should have no loading times longer than ten seconds.
NF.P.002 The game must not "lock up" or "freeze". – even ??
NF.P.003 The user must be able to interact with the program using a mouse, or joystick.
NF.P.004 The GIF and JPEG format should be used to display the die and tokens which will make file sizes as small as possible. imp.
NF.P.005 Audio files should be created in the AU format which is most compatible with Java.
~~**NF.P.006** The game will be able to be played using a joys~~tick.

Validation Criteria

The team knows that the correct requirements have been produced because the requirements specification has included: a precise and accurate problem statement, a detailed system model, and all necessary functional and non-functional requirements. The problem statement tells the team what to do for the project and gives the purpose for doing the project. After each phase in the software lifecyle, the problem statement will be reviewed to validate the work that is being done and to test if the work being done is really headed toward solving the problem statement. The system model for the project gives the team a visual image so the team can visualize the processes and entities that will be implemented by the program. The objects that will be coded in the program will follow the layout of the system model. Without the system model, programmers will most likely begin coding aimlessly, which will result in the final code lacking strong object oriented properties. By using a system model, the programmers can code abstractly, which will allow all of the pieces of the program to communicate properly with each other, without the use of coding shortcuts. Finally, the functional and nonfunctional requirements allow the team to know when they are finished with the design and coding activities. When all of the necessary requirements (also known as 'musts' and 'shoulds') are reached, only then will the team know that the project is complete. good! After all necessary requirements are accomplished, the team may be begin to work on many of the extra features (also known as 'wills') that should have been added to the code, but were not required by the specification. By following these guidelines, the team has a clear view on all the work that must be completed to finish the project, therefore the team has concluded that the software requirements are correct.

Finding information for the project was a long process. The team knew that in order to obtain enough information to allow the production of a well-defined program the major elicitation techniques must be used. First the team had to find sources with professional knowledge on the subjects of computer entertainment and child psychology. Talking with a teacher, who spends time everyday working with children, allowed the team to get enough information about a child's mentality to tailor the software for a child's greatest enjoyment. – what about education? The team received information about the things children like to see in electronic games, the best way to hold the child's attention, and most importantly the most effective way children learn. → 1 way for all kids? Members of the team also visited a local electronics gaming stored named Gamer's First. The team members inquired the sales representatives about the important factors that must be present to make an entertaining game. The employees shared the secrets of the video game industry and let the team know which areas of the game would require the most attention. Information about games found on the World Wide Web gave the team much needed knowledge that could be used in creating the game. Since the team also used both hard copy~~ed~~ and online surveys, data was gathered on what common people felt was the most important factors in making the game. The official rules for the official Trivial Pursuit® game were also reviewed, allowing the team to think of ideas on how to improve the existing game. After gathering all of the information, the team felt that all of the information necessary for making the game was present, and further researching would not greatly benefit the creation of the game. Therefore the team has concluded that enough information has been gathered for the requirements specification.

good

In order to ensure that our data was correct, the team tried its best to check each source for validity. The team made sure that all people interviewed had some knowledge of the electronic entertainment industry or had professional experience with child psychology. The team safely assumed that all employees of electronics and gaming stores had professional knowledge of the gaming industry. The teacher who was interviewed, Ruth Jackson, had obviously spent much time working with children due to her profession. The on-line survey's validity has to be questioned due to the fact that anyone who has access to the world wide web can select their opinion on the questions presented. Further crippling the validity of the source, a person could vote multiple times, allowing one person's opinion to outweigh the opinion of another. The on-line data will not be interpreted as a professional source, but instead the data will be used as an indication of what most ordinary people think about which factors should be emphasized when creating an educational computer game.

The team must follow the requirements stated when developing the program, else the software produced will not result in the product which was expected which will lead to the initial problem statement remaining unsolved. When a requirement is completed, a team member will mark the requirement as completed on the team's master requirements list. The team will review the current project progress to ensure that the requirement was successfully completed and that the team has not lost focus on the task to be accomplished. Each team member will give his/her opinion on the state of the progress made by

in coding
design a both?

the team. If a team member feels that a requirement was not given enough attention or a requirement was not fully completed, then the team will decide if development time should be spent on the area of the project in question.

Methodology Analysis

The object oriented analysis technique was chosen since each entity in the software acted almost independently of each other and required minimal coupling. A more structured approach was undertaken, but the team found the approach did not yield the most appropriate results. In the game, there were not many distinct states and were not intensively data driven, therefore the structured analysis was insufficient for the team's purposes.

Need more explanation

Elicitation Techniques

The team's elicitation techniques consisted of interviews, surveys, document reviews, and observation. The interview with Mrs. Ruth Jackson was chosen because the team needed information children's mentality. Since Mrs. Jackson is an educator and a parent she was the optimal choice for the interview. The team chose to do surveys since the online surveys were accessible from people of many cultures around the world. Since the existing game documents gave information about how the game should be played, the team analyzed the documents. Members of the team visited Gamer's First and local arcades to observe what aspects of an electronic game were most appealing to players. An interview was also conducted at Gamer's First giving the team information about the processes required for making an educational game. If time had permitted, our team was going to visit a large gaming corporation (such as Parker Brothers or Nintendo of America) to obtain information from the people who actually play the largest role in creating the games that children play. Due to time and constraints and lack of funds our team was not able to partake on the external activity. These sources could have greatly added to our collection of resources.

- discussed this more in the previous section

Sample Interview Questions

Name

Company/Organization

What do you feel is the most important factor when creating an entertaining videogame?

Do you believe it is possible to make a game that is educational but still hold a child's attention? If possible, explain how you think this might be accomplished.

How many hours do you believe a child plays videogames a week?

On average, how much time do you feel a child would play a Video game before taking a break?

Interview with Ruth Jackson, Educator

(**Note:** This document has not been edited for grammar in an attempt to remain unbiased during the course of the interview. JHatGA is a member of our team, in this case Jamie Hobbs, and Jaxdrac is Ruth Jackson.)

JHatGA: I have to program a game in my Comp Sci class. It has to be an educational game for children. My question is what do kids like in a game? It has to be both educational and fun.

Jaxdrac: they like repetition, bright colors, and catchy music. The content can be anything as lopng as those elemants are the delivery vehicle!

JHatGA: repetition, bright colors, and catchy music

JHatGA: got it

Jaxdrac: yep. Kathleen knows every song on her "squirrel" program

JHatGA: and when they get older? like middle school range?

JHatGA: squirrel?

Jaxdrac: violence

JHatGA: hard to introduce violence into an educational game though

Jaxdrac: The squirrel is actually the narrator-- giggles the gopher is its true identity. But my 3 year old has herself convinced it is a squirrel

Jaxdrac: I am not sure about hte older kids. Try short puzzle type things that require the use of specific knowledgfe to solve

JHatGA: alright, thanks. I needed an interview with a teacher

JHatGA: that was close to an interview

Jaxdrac: like answering a question about geography reveals a piece of a silly picture that they have to figure out what it is. And have several pictures that are traded out so that the puzzle is not always the same

JHatGA: classic concentration style

Jaxdrac: yeah, kinda like that.

Jaxdrac: with the littleones, it is easy, sort of, because they know so little. Any subject is really cool to them. KAthleen likes the part that lets her hear different instruments playing different songs. SHe learns the names of the instruments, what they look like, and what they sound

Preliminary Instructions for Mind Trial

Object:
To move around the board answering questions, and to collect the 6 colored pie pieces by answering the question correctly at each of the "headquarters". To win, a player must acquire the pieces and return to the central starting point where he/she will be asked the final question from a random category.

Game-play:
The first player "rolls" the die and moves the allotted number of spaces in any direction. A player cannot move both forward and back in the opposite direction in the same move. A question of the appropriate category will be asked. If answered correctly, the player gets to go again. If the question was from one of the category "headquarters" the player is awarded a pie piece of that category and gets to go again. The player continues in this way until a question is missed. Then the second player goes in the same way, and play continues until a winner is determined.

Around the board are two kinds of spaces: normal questions and scoring questions at the "headquarters". The normal questions are the most ample spaces. They make up the "spokes" from the central starting point and the majority of the spaces between the scoring corners. At each corner is a category "headquarter". A correct answer here earns a scoring wedge. More than one playing piece can occupy a space.

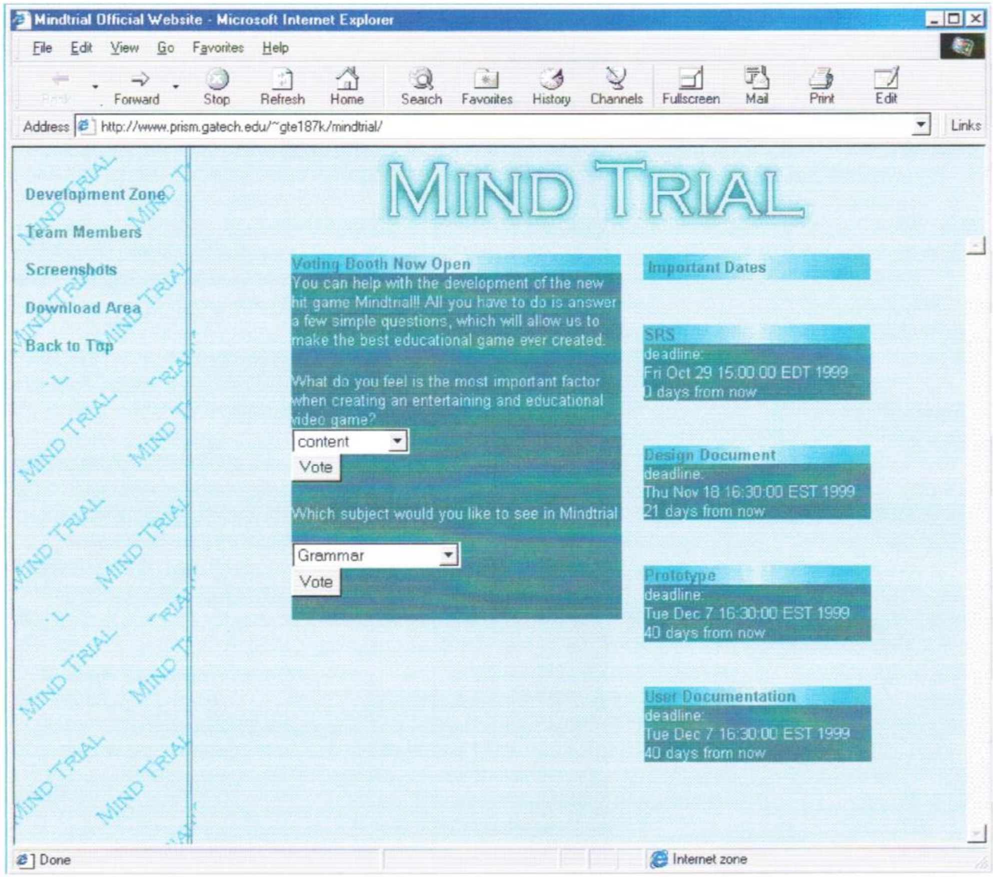
Winning the Game
Once each category's piece has been acquired, the player must return to the center. Once he/she lands on this spot *by exact count*, a question from a random category will be asked. If answered correctly, that player wins. If incorrect, that player must move off of the center and return later, again *by exact count*, to try to answer another winning question.

For information on updates and questions, please visit the official website at:
http://www.prism.gatech.edu/~gte187k/mindtrial/

Figure A : Software Budget

Item	Description	Amount
Java compiler	jdk1.1.8	$0.00
graphics tools	JASC Paint Shop Pro Demo	$0.00
	Adobe Photoshop*	$0.00
textbook	"Software Engineering - A Practitioner's Approach" 4th Ed	$90.00
Internet access*		$0.00
travel expenses for interviews, research, and other outside activites		$20.00
report covers		$3.99
computer paper	Great White 500 sheets	$5.99
writing utensils	6 pack of BIC No. 2 mechanical pencils	$1.59
presentation costs	posterboard, markers, transperancy sheets, etc.	$10.00
personal expenses		$20.00
Total		$151.57

* Provided by College of Computing


Mindtrial Official Website - Microsoft Internet Explorer
File Edit View Go Favorites Help
Back Forward Stop Refresh Home Search Favorites History Channels Fullscreen Mail Print Edit
Address http://www.prism.gatech.edu/~gte187k/mindtrial/
Links
Development Zone
Team Members
Screenshots
Download Area
Back to Top
MIND TRIAL
Done
Internet zone

Voting Booth Now Open

You can help with the development of the new hit game Mindtrial!! All you have to do is answer a few simple questions, which will allow us to make the best educational game ever created.

What do you feel is the most important factor when creating an entertaining and educational video game?

content

Vote

Which subject would you like to see in Mindtrial

Grammar

Vote

Important Dates

SRS
deadline:
Fri Oct 29 15:00:00 EDT 1999
0 days from now

Design Document
deadline:
Thu Nov 18 16:30:00 EST 1999
21 days from now

Prototype
deadline:
Tue Dec 7 16:30:00 EST 1999
40 days from now

User Documentation
deadline:
Tue Dec 7 16:30:00 EST 1999
40 days from now

For the Voting Booth survey - ***What do you feel is the most important factor when creating an entertaining and educational video game?***

The vote results are:

#	Answer	Count	%
1	content	34	43.6%
2	graphics	13	16.7%
3	replay value	11	14.1%
4	control	11	14.1%
5	sound	6	7.7%
6	complexity	3	3.8%

Total Votes: 78

OK88 Internet Services (www.ok88.com)

For the Voting Booth survey - ***Which subject would you like to see in Mindtrial***

The vote results are:

#	Answer	Count	%
1	History	7	30.4%
2	Computer Science	6	26.1%
3	Current Events	4	17.4%
4	Music	2	8.7%
5	Mathematics	1	4.3%
6	Grammar	1	4.3%
7	Geography	1	4.3%
8	Foreign Languages	1	4.3%

Total Votes: 23

OK88 Internet Services (www.ok88.com)

Name

Steven B Cribb

Company/Organization

Gamer's 1st

What do you feel is the most important factor when creating an entertaining videogame?

Ingenuity, originality, and fun play mechanics, which should also be original, yet not too complex. Intriguing presentation and premise are also important. Good grap are almost a necessar plus.

Do you believe it is possible to make a game that is educational but still hold a child's attention? If possible, explain how you think this might be accomplished.

Yes. This would have to be a delicate balance of fun gameplay, learning/subject matter that is presented in an original & fun manner, and reward. "Learn & Reward" would be the overall objective for the game designers.

How many hours do you believe a child plays videogames a week?

15+

On the average, how much time do you feel a child would play a videogame before taking a break?

1hr

Glossary

Hungarian Notation – prefacing a variable's name with a representation of its type.

Object Oriented language – a type of programming language that uses different classes to represent "real-world" objects that interact during the course of implementing the software.

Part/Whole – type of class communication that results in one class representing a whole, while another one is a piece of that whole class. For example, the relationship between a Car class and a Door class would be a Part/Whole communication link, because a door is part of a car.

Send/receive – type of inter-class communication that results in one class sending information to another.

Sources & Bibliography

The rules provided with the Trivial Pursuit ® board game.

The following web sites have contributed information to our project:
Trivia Game news headlines
www.chatgames.com
Trivial Pursuit Competition
www.genusnet.com/friends/trivia.htm
-dedicated to trivia competitions. Though the actual competitions are held off-line, this page keeps an accurate update of the actual event.
Interesting Trivia on the Trivial Pursuit Game
www.newswire.ca/releases/November1997/06/c0721.html
-provides a history and various trivia facts about Trivial Pursuit ®
Live Interactive Trivia Games
www.ntn.com
-provides information on trivia competitions in general
Trivial Pursuit Home Page
www.trivialpursuit.com
-provides information on the trivial pursuit game.

Various interviews conducted locally at a few retail stores including:
Steven Cribb at Gamer's First

Need better. documentation of who you interviewed

Interviews with school teachers, like the one with Ruth Jackson.

Online survey data that can be found at the official website. - URL?

Submissions via e-mail of what visitors to the official website would like to see in an education software program.

Electronic Gaming Monthly vol. 123 p. 213-224.

GUIS

CS 3802 - Introduction to Software Engineering

Software Requirements Specification - Grading Criteria
Fall Semester 1999

DUE: 28 October 1999

TURN-IN: 2 copies of your document should be turned in. One copy will be graded and the other will be given to your design team.

GOAL:

The purpose of this assignment is to prepare a Software Requirements Specification. The document should present the requirements and a system model for the educational game you are developing.

GRADING CRITERIA:

System Model (20)	10
Requirements (30)	
Functional Requirements (20)	12.5
Non-functional Requirements (10)	5
Validation Criteria (10)	9
Method (15)	
Methodology Analysis (5)	2.5
Elicitation Techniques (10)	7.5
Quality of Requirements (15)	
Completeness (5)	4
Conciseness (5)	4
Correctness (5)	2
Document Quality (10)	
Organization (5)	5
Style & Grammar (5)	4
Total (100)	65.5

Back to Project Assignment | Back to Project Information Page | Back to CS 3802 Home Page

Name: SMITH,LEVI

CS 3802 – Introduction to Software Engineering

Fall Semester 1999

Midterm Exam – 26 October 1999

Instructions: The time for the test is 1 hour and 10 minutes. All the answers are to be written on this paper. One of the secrets to success on this exam is to be concise and neat; the other is to be sure to ask for clarification if you think you need it. The test is closed book and closed notes, but open mind.

78

1. (5 points) What is the most important phase in the software development lifecycle? Why?

Software requirements is the most important phase in the software development lifecycle because all other phases depend on the software requirements (since it is the first one, as illustrated in the waterfall model). If the software requirements are incorrect, then all other phases will have to be corrected. (Errors propagate through the rest of the lifecycle)

2. (5 points) Define 3 software quality attributes.

✓Maintainability - how hard is upgrading (adding new features) to the code

✓Usability - how difficult is it to train/learn how to use the software

✓Durability - how error prone is the software; will it crash? (Will it survive Y2K? :))

3. (5 points) Explain the Rapid Application Development (RAD) model.

If an application is created very quickly then it is more error prone than an an application developed slowly using a well, thoughtout process.

-5

4. (5 points) Do you agree with the following statement? Why or why not?

"The only important deliverable for a successful project is the working program."

I do not agree with the statement.

The working program is not the only important part of a project. Data models, requirements, and code comments, are all as equally (if not more) important than the working program itself, since maintenence has been proven to be the most costly phase of the software lifecycle. If only a working program was delivered then it would be much more costly for another team to maintain the software.

-5

Name: SMITH,LEVI

5. (10 points) List 3 functional and 3 non-functional requirements for an on-line ticket ordering system (i.e. Ticket Master).

Functional

1) Must know if an event is sold out or if seats are available.

2.) Must know the price for each seat (or group of seats) at an event

3.) Must know the location (street address, zip, state, etc) of all events in database.

Non Function

1) Should run on both Windows and Linux ~~boxes~~ operating system,

2) Should be written in Java

3.) Should be able to get the event information in less than two minutes,

6. (15 points) Your consulting company has been contracted to develop a custom word processing application for Documents 'R Us. What elicitation techniques would be most appropriate for this project and why?

✓I
✓S
✓O
✗B
✓E
✓R
✗R

Review Existing Documents - you will need to know what type of documents the company will be using with the software.

Ethnography, you might want to see how the company currently does work so you will know the most effictive way your program can be of use.

not really

Interviews / Surveys - you must know what the employees want in a documents processing program.

Observe - observing the employees work would also benefit since you would see know the employees actually work, instead of a "higher up" telling you how things are done, which will probably be biased towards his perspective.

CONT on BACK

10

I'm assuming this company does not have existing word processing softwares but if it did, the reviewing the existing software would probably be an appropriate ellicitation technique

Name: SMITH,LEVI

7. (15 points) Discuss the advantages and disadvantages of developing a software system using a prototyping software process model.

Ad-vantage The prototyping software process model would allow one to get instant feedback from the customer, so you could know what the customer wants instead of waiting till the program is complete to show the project to the customer, which will likely result with many more revisions, according to the customer's request.

Advantage - Some customers like to know how the project is going - they like to be updated frequently. ~~same~~ Same as above.

Disadvantage - Some customers don't like to be a part of the software development process. ✗

9

8. (15 points) Compare and contrast Structured and Object-Oriented Requirements Analysis. Explain the advantages and disadvantages of each method.

Object Oriented Requirements analysis relies on objects which interact with each other as the way to model the requirements. There is only one type of data model which is the object oriented data model, which illustrates the properties and services of an object, and how object communicate with each other.

Structured Requirements Analysis uses ✓ software processes and data transfer as the analysis technique. Structured analysis offers more types of data models - Entity Relationship, State Transition Diagram, ✓ and Data Flow.

Entity relationship shows the properties of each entity and how each entity relates to the other ✓ entitys.

State Transition Diagram shows the state of the program and how it moves from one state to another

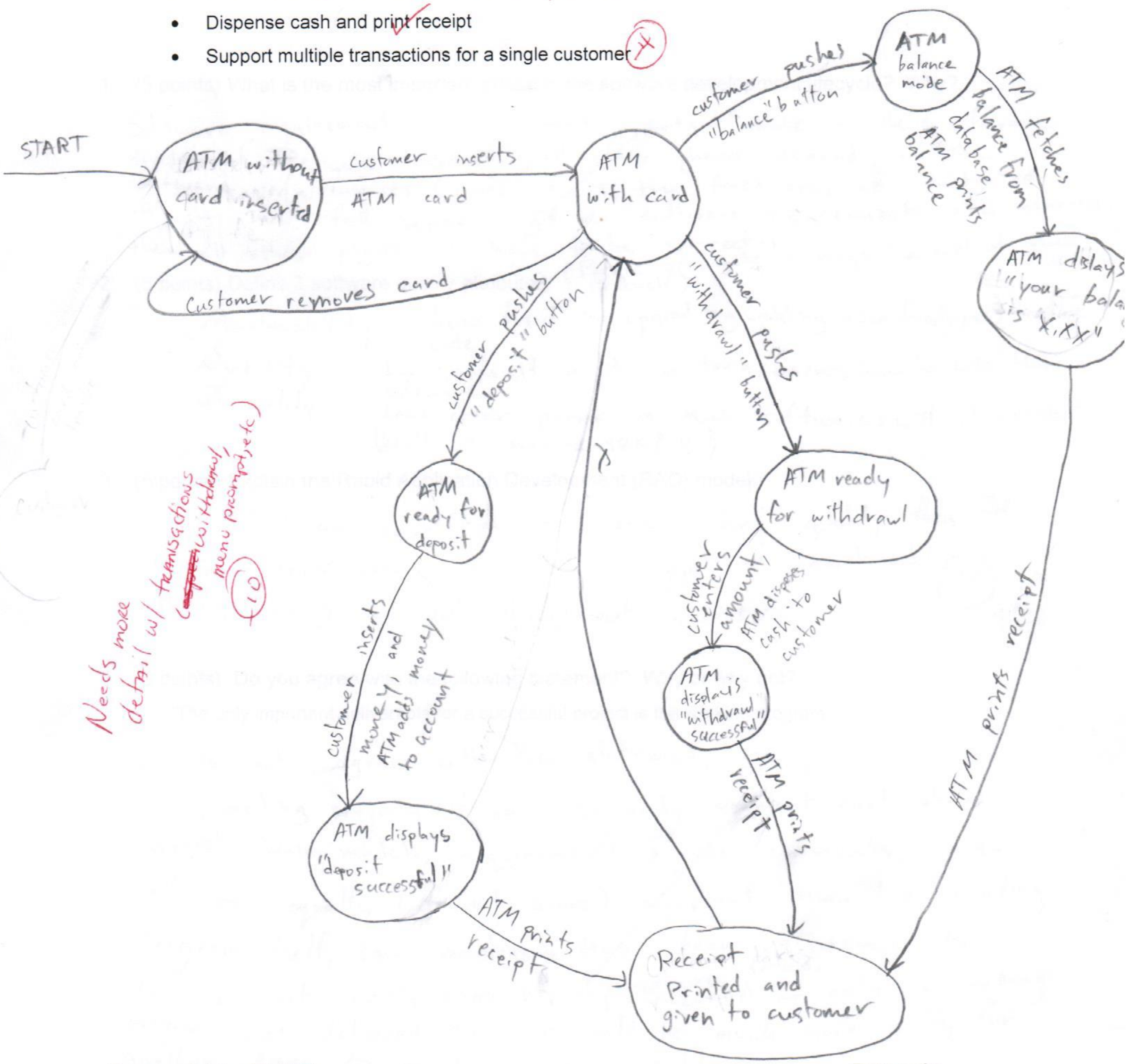
CONT on BACK ⇓

The data flow diagram illustrates how data goes from one process to another process,

Name: SMITH,LEVI

9. (25 points) Remember the ATM that we have analyzed repeatedly in class. As you recall, we continue to make simplifying assumptions. This time draw a state machine diagram for a "real" ATM. Be sure to state any assumptions you make. It should have, at minimum, the following: (Bonus points may be earned by increasing the complexity of your ATM beyond the standards listed below.)

- Secure access via ATM Card -4
- Process deposits ✓, withdrawals, and balance ✓ inquiries
- Dispense cash and print ✓ receipt
- Support multiple transactions for a single customer -4


START
ATM without card inserted
customer inserts ATM card
ATM with card
customer removes card
customer pushes "balance" button
ATM balance mode
ATM fetches balance from database ATM prints balance
ATM dislays "your balance is X.XX"
ATM prints receipt
customer pushes "deposit" button
ATM ready for deposit
customer inserts money, and ATM adds money to account
ATM displays "deposit successful"
ATM prints receipt
customer pushes "withdrawl" button
ATM ready for withdrawl
customer enters amount, ATM dispenses cash to customer
ATM displays "withdrawl successful"
ATM prints receipt
Receipt Printed and given to customer

Needs more detail w/ transactions (withdrawl, menu prompts, etc) -10

-18

0 43100 34026 3