CS445 / CS645 / ECE451
Fall 2017 — Final Exam

7 December 2017, 9:00am–11:30am

Instructor: Daniel M. Berry

Time allowed: 2.5 hours = 150 minutes

No aids allowed (*i.e.,* closed book).

Answer all of the questions on this exam paper.

There are 10 questions for a total of 150 marks.

Plan your time wisely: 1 minute per mark

Your Name and Student Number

In the immortal words of the yet to be born Worf of Qonos,

Qapla!

Q1		scaled to 15
Q2		scaled to 15
Q3		scaled to 10
Q4		scaled to 5
Q5		scaled to 20
Q6		scaled to 15
Q7		scaled to 20
Q8		scaled to 10
Q9		scaled to 25
Q10		scaled to 15
TOTAL		scaled to 150

In this exam, a short underscore of 1 inch (= 2.54 cm) should be filled with one word. A long underscore of 3 inches (= 7.63 cm) should be filled with a phrase consisting of one to several words. In the either case, if you cannot think of exactly the right number of words, then give the best answer that you can and we'll give it as many marks as we can, possibly even full credit! If you cannot even think of just words to fill in, then write an answer as a sentence, and we'll give it as many marks as we can.

"(complete the sentence)" means that in the space following the current line, you are to complete the sentence that was started and interrupted with the "(complete the sentence)"

If you are asked a question, which ends with a "?", you are to answer that question in the space following the current line.

In this exam, if you are asked for a simple answer, you need not justify it, unless you are also asked explicitly "Why?". However, you may always write down assumptions that can help us give you partial credit.

If an exam question directs you to list, describe, write, explain, draw, mark, put, change, modify, or anything like that, just do so.

In the exam questions,
"CBS" means "computer-based system".
"NFR" means "non-functional requirement", a.k.a. "quality attribute".
"RE" means "requirements engineering".
"SRS" means "software requirements specification, written according to some standard, e.g., IEEE".
"UM" means "user's manual".

Note the difference a serif front like that used for *this* sentence
and a sans serif font like that used for *this* sentence.

1. **[15 total marks]** RE Reference Model and Validation

   A summary of *The Computer Scientist Who Prefers Paper* by Jill Leovy from the *The Atlantic* on 12/01/17 offered by the ACM says (quoting):

   > Former ACM president Barbara Simons' concerns about electronic voting systems' vulnerability to manipulation recently have gained credibility with the public, especially with suspicions of Russian meddling in the last U.S. presidential election coming out. As a co-director of Verified Voting, Simons subscribes to paper-based voting techniques, and she notes many of her computer scientist peers oppose paperless elections "because we understand the vulnerability of voting equipment in a way most election officials don't." Simons thinks mandatory hand-counted audits of randomized ballot samplings are critical for safeguarding against concealed vote theft. She also notes paper voting provides a permanent record, stressing, "There's no malware that can attack paper." Verified Voting supports certain hybrid voting machine models that ink paper ballots and can help people with disabilities to vote, provided the results are audited. "The technical community has a responsibility to inform policymakers of the limitations as well as the benefits of technology," Simons says.

   The summarized article itself says (quoting):

   > What paper [ballots] boasts—and no existing computer system can rival—is a solution to the confounding logic problem at the heart of our electoral system. The secret ballot presents a paradox: How can the validity of each vote be confirmed without being traceable to any individual voter? Ballots must be "anonymous and yet verifiable, secret and yet accountable," says Eric Hodge of CyberScout, a security-services company that advises states and counties.
   >
   > Paper [ballots], Simons said, is the best answer to this riddle. Marked clearly and correctly, it's a portable and transparent record of voter intent, one that voters themselves can verify, at least while the ballot is still in their possession. It's also a permanent record, unlike computer memory, which can always be overwritten. "There's no malware that can attack paper," Simons said.

   Let's frame Simons's argument in terms of $D, S \vdash R$.

   (a) What one sentence of the two quotations above should R be?

   (b) In terms of $D, S \vdash R$, if D includes

   > There are no paper ballots ,

   what is Simons saying about any election system, whose behavior is specified by S?

(c) In writing either an SRS or a user's manual (UM) about some program, occasionally you may need to distinguish between the program being specified as an artifact supplied on a medium and an invocation of the program.

In this case, you are recommended to make two entries into the glossary:

the X program = a copy of the X program on a CD

X = an invocation of the X program running on your your computer

What is the reason that the a copy of the X program on a CD is given the long name, the X program, while an invocation of the X program running on your computer is given the short name, X?

(d) Under what circumstances would it be preferable to put the following entries into the glossary?

X = a copy of the X program on a CD

a run of X = an invocation of the X program running on your computer

(e) What is a potential source of confusion in the two sentences at the beginning of the typical Registering to Vote scenario?

Before registering to vote, a voter must obtain a voter identification number, token, and registration address from the election administrators.

You may begin the registration process by running the pollster module. This is generally done by invoking the sensus command.

2. **[15 total marks]** UML Domain and Use Case Models

Consider the following nonsense sentences that follow English grammar rules, which are derived from nonsense sentences that follow Russian grammar rules.

> The glocky kuzdra shteckly budled the boker and
> kurdyaks the young bokers.
> Each boker of any kind has two shtukovinas and one kaktyetona.

In these sentences, the English words have their normal meanings.

(a) On the next page you will find the skeleton of a UML class diagram of the world described by the nonsense sentences. The diagram shows only the classes. Missing are stereotypes, multiplicities, links, attributes (*i.e.,* variables), and methods (*i.e.,* operations). Please complete the diagram with the missing stereotypes, multiplicities, links, attributes, and methods.

You must insert "≪actor≫" in each class that describes an actor.

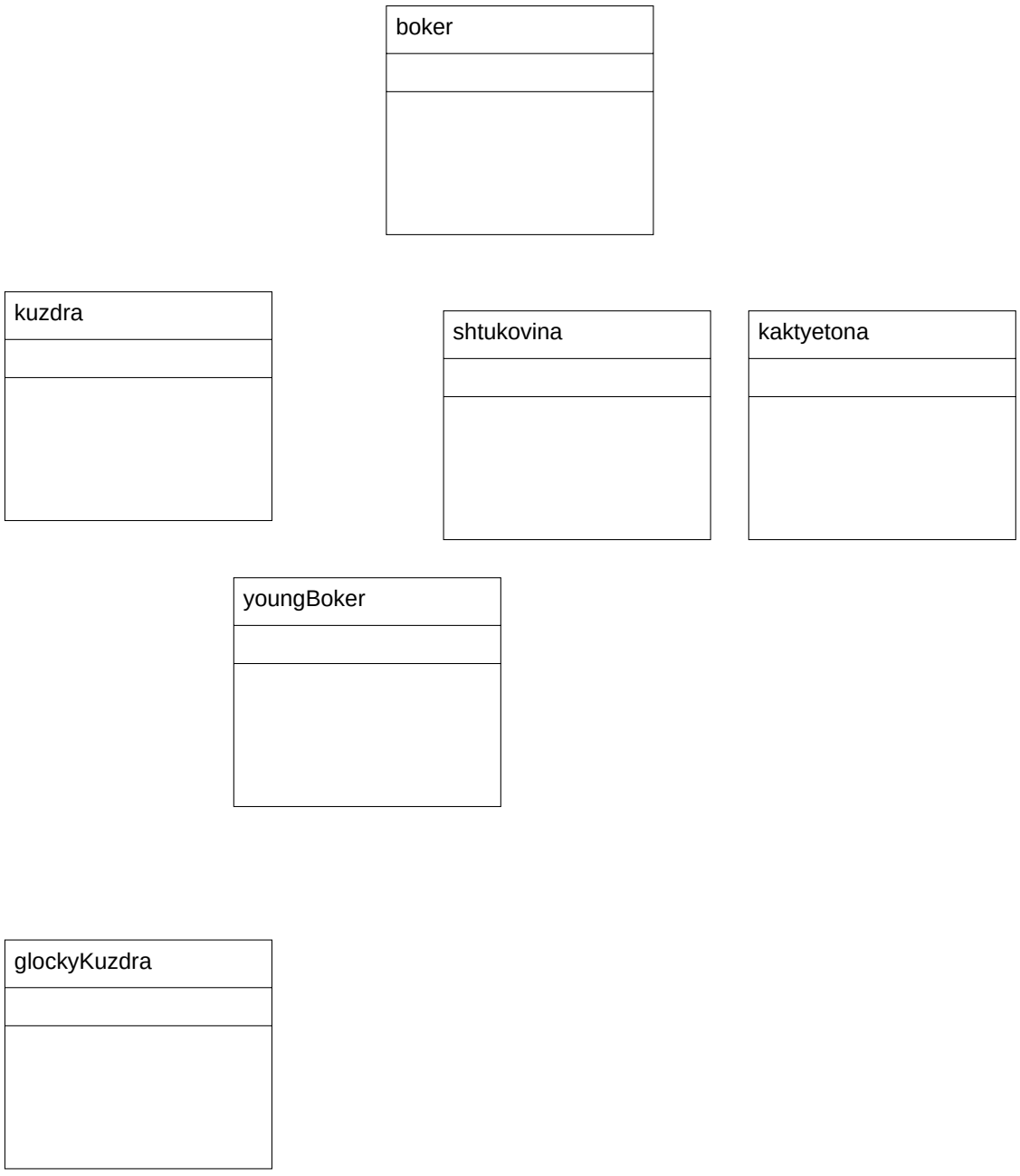
You must insert all and only links that are needed, including those describing aggregation (componentry) and inheritance (subclassing).

You must place each of the following verbs in only the right classes: budle and kurdyak. It is possible that some appear in more than one class.

An adverb (*i.e.,* a nonsense word ending with ly) is to be treated as an argument value passed to a parameter called how to the method corresponding to the verb modified by the adjective.

An adjective (*i.e.,* either a regular English word or a nonsense word ending with y but not with ly) induces a subclass.

You must provide the right multiplicity to each class (but not to the links), using the exact number of instances if there is a specific number of instances of the class and using "*" if there are an unknown, possibly multiple number of instances of the class.

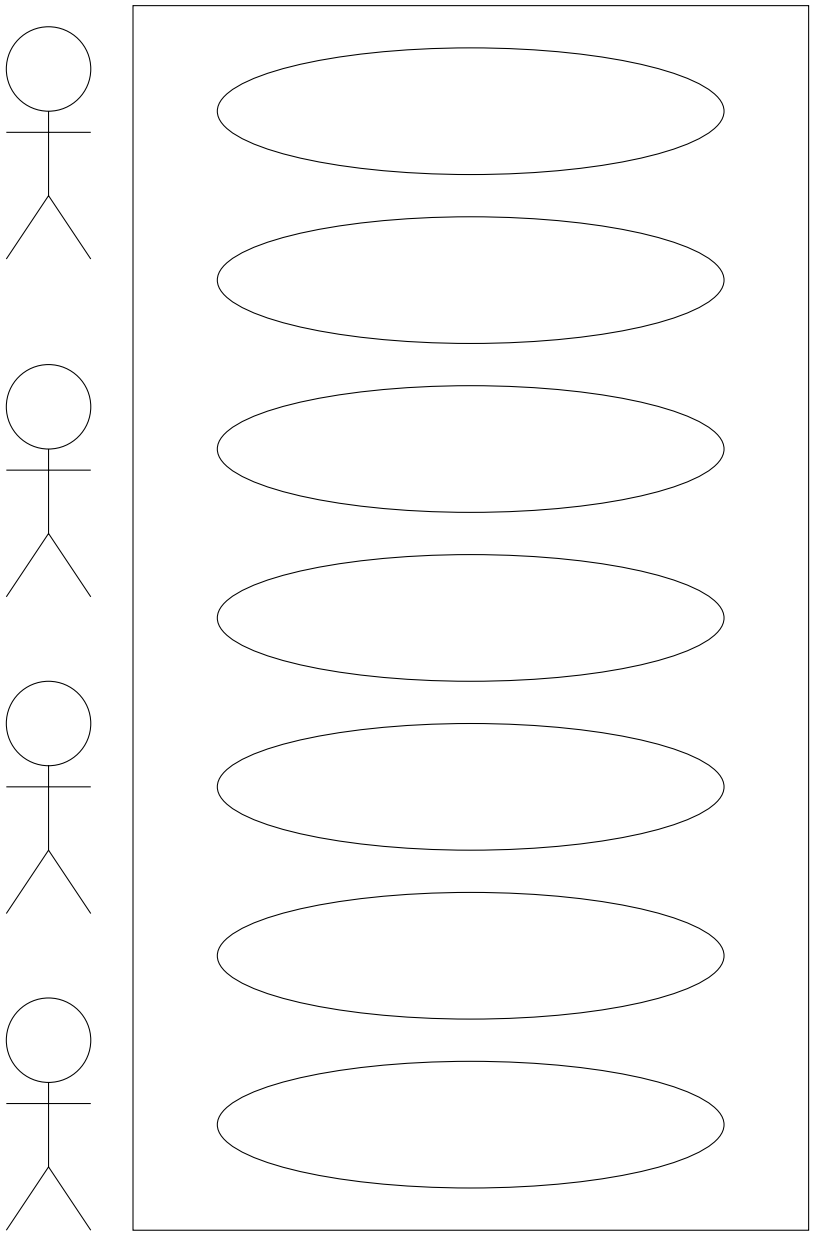

boker
kuzdra
shtukovina
kaktyetona
youngBoker
glockyKuzdra

(b) Mark the link that permits the doing of the verb budle with "1". Mark the link that permits the doing of the verb kurdyak with "2". Based on tenses of the verbs in the original sentence, which link is performed first?

(c) What is the coordination ambiguity (an ambiguity involving a conjunction) in the nonsense sentences?

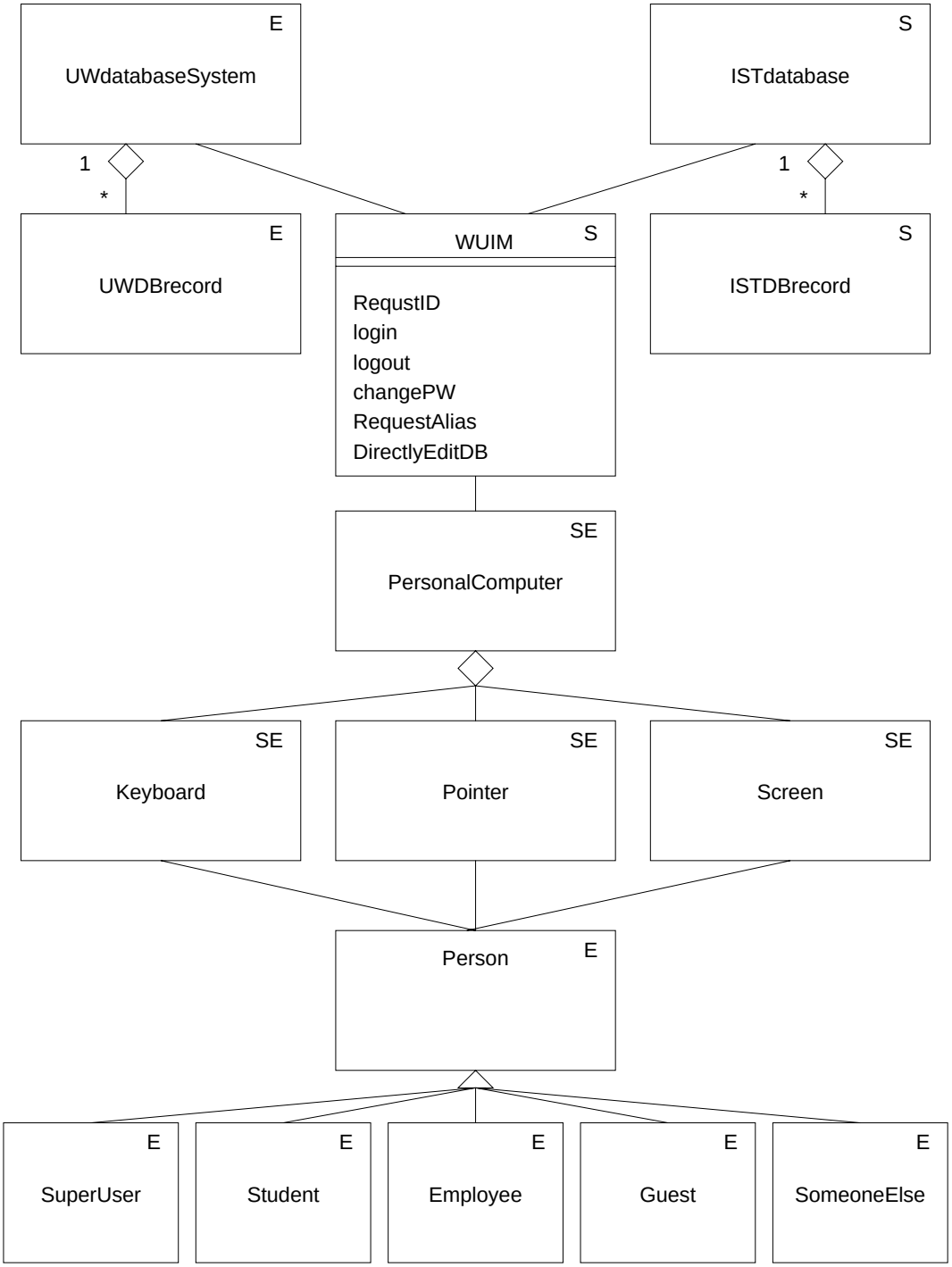
(d) What way does your diagram disambiguate this ambiguity?

(e) Below is a skeleton for a use case diagram for the same world of nonsense sentences. It has more actors and blank use cases than are needed. Label each actor that is needed with the name of an actor from the class diagram, and fill in each blank use case that is needed with the name of an operation from the class diagram that is visible to an actor by virtue of the links in the class diagram. Draw a link from each actor to only each use case that it uses by virtue of the nonsense sentences. It is OK to have a use case that is not linked to any actor and to have an actor that is not linked to any use case. Finally, put an "X" through each actor and blank use case that is not needed.



3. **[10 total marks]** WUIM

A variation of the published world diagram for WUIM is below.



(a) We are given the rule that the vocabulary for the specification of a CBS must come from the entities in the interface part, the intersection betweeen the CBS's environment and the CBS's system.

Most likely, your Deliverable 6, whether a UM or an SRS, *mentions* the ISTdatabase as the place keeping the permanent list of all accounts ever created. Most likely, your specification of each of the RequestID and RequestAlias use cases *mentions* the ISTdatabase and its components as being interrogated for clashes and being updated during the use case's operation. In this case, is the rule being followed?

(b) Describe two different changes, each to a different part of the Deliverable 6, that would lead to the vocabulary rule's being followed:

i.

ii.

(c) We choose to change the world diagram. On the diagram in the previous page, somehow indicate the change that must be made. That "somehow" can be a short textual description of the change, it can be the crossing off of removed entities and drawing in of added entities, it can be an arrow pointing to where an entity is moved, or a combination of the above. Alternatively, in the space below, you may draw a new diagram.

(d) Consider an application of $D, S \vdash R$ to the WUIM second updated project vision. That is, the WUIM second updated project vision is taken as R. For your project, you spent a lot of time identifying exceptions and alternatives to the so-called typical scenario of each use case, and to *try* to find ways within S to deal with all of these exceptions and alternatives, so as to minimize the assumptions D about the environment. What is the one element r of R, concerning WatIam user IDs that cannot be mitigated (fixed, solved) by any addition to S, although S can specify (1) detecting that r is not going to hold and (2) then reporting an unrecoverable problem if r *does* hold?

Thus, basically, D is forced to assume that r holds.

4. **[5 total marks]** Use Cases and Scenarios

   (a) Consider the two main use cases for the Sensor voting system:

   > Registering to Vote

   and

   > Marking Your Ballot

   The beginning of the scenario for Registering to Vote says

   > Before registering to vote, a voter must obtain a voter identification number, token, and registration address from the election administrators.
   >
   > You may begin the registration process by running the pollster module. This is generally done by invoking the sensus command.

   You may assume that a voter and you refer to the same user, you the voter.

   The beginning of the scenario for Marking Your Ballot says

   > Before you can mark a ballot, you must obtain the unvoted ballot for the election and place it in your .sensus directory. You must also be registered to vote in that election.
   >
   > Start by running the pollster module as you did when you registered to vote.

   Note that subsequent steps of the scenario for Marking Your Ballot include the Pollster module's asking you to supply the name of the ballot and the Pollster module's supplying you with ballot questions on which you will vote, presumably because they are as yet **un**voted.

   What is the difference between the list of activities that must be done before running the Pollster module for Registering to Vote and the list of activities that must be done before running the Pollster module for Marking Your Ballot?

5. **[20 total marks]** Graduate Student Lectures

Frédéric Bouchard: Declarative Specifications for Software Code Base

(a) What is Bouchard's key concept?

(b) In what step of the software development lifecycle does Bouchard hope to achieve the biggest cost savings?

(c) According to Bouchard, what is wrong with the code of many imperative programs?

(d) When Bouchard asked "Is it still intuitive that the list I pass in [as a] parameter will be modified?", for what kind of system did Berry say that it *very intuitive* that an input parameter would be modified?

Shuchita Singh: Prototyping in RE

(a) According to Singh, what is the main goal of prototyping in software development?

(b) According to Singh, what are the three main kinds of prototyping?

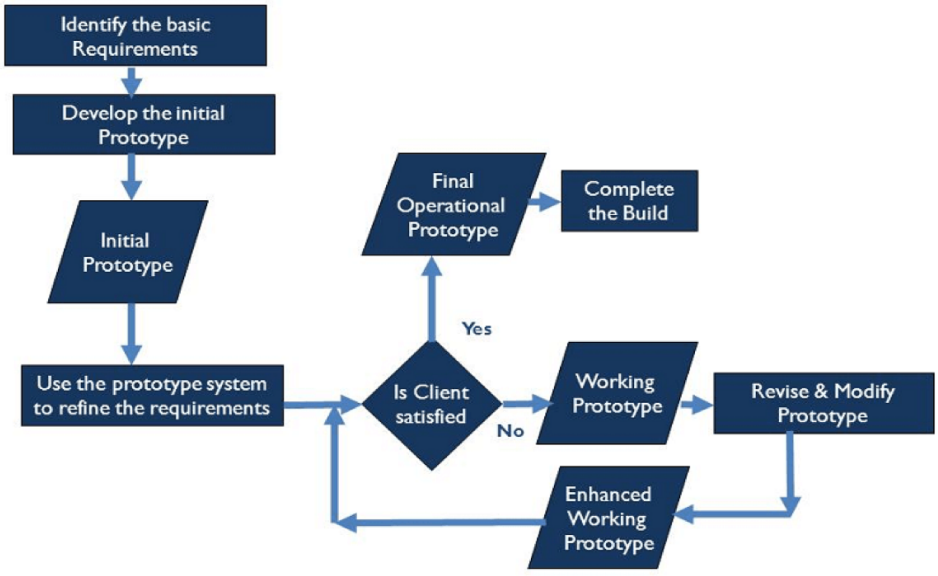
i.

ii.

iii.

(c) According to Singh, what is the main difference between throwaway prototyping and the other two, in terms of the production software? (Your answer must not use the word "throwaway" or any synonym for "throwaway"!)

(d) Below is a diagram from Singh's slides showing the lifecycle of prototyping.



Given the way the lifecycle ends, which of the three types of prototyping can it *not* be?

(e) Modify the diagram above to make it correct for the one type of prototyping it cannot be.

Johan Sjöberg: The Pit Stop 20 Project: How SAS Became the World's Most Punctual Airline

(a) According to Sjöberg, what historical event created the opportunity to do the analysis that SAS did?

(b) According to Sjöberg, what was SAS's original goal driving the analysis that it did?

(c) However, what SAS goal is *actually* helped by reducing the turn around time of an aircraft to 20 minutes?

(d) What are the main drawbacks of reducing the turn around time of an aircraft to 20 minutes with respect to the goal of improving punctuality of SAS's flights?

JunHao Lu: Water Demand Data Analysis Automation: Requirements and Specifications

(a) Lu's talk described a requirements engineering experience of his in which he had to determine the requirements for a program to analyze water demand in the City of Waterloo. The requirements specification given by the client to Lu gave an algorithm for doing the analysis that the client wanted Lu to implement. What document served as the requirements specification from the client for this software?

(b) What requirement by the City of Waterloo conflicted with the requirement to gain sufficient revenues from selling tap water?

Wow!!! What irony!!!

(c) Lu's starting the development from Matlab with its built-in functions for plotting and least square fitting is basically an example of _____ _____ing.

(d) Each system built has functional requirements, *i.e.,* the calculations it must do, and nonfunctional requirements, *i.e.,* the required efficiency. For most systems, functional requirements are _____ to test than are nonfunctional requirements. Evidently, for the *system built by Lu*, because it was very hard to _____ which of multiple possible outputs were _____, the functional requirements are _____ to test than are nonfunctional requirements. Wow!!!

6. **[15 total marks]** NFRs

(a) Even when you do know what the actual response time of The CBS is, it is still difficult to decide *whether* The CBS has fast response time for two different reasons (complete the sentences):

i.

ii.

(b) An NFR, R, has a *perception problem* when different people will have different perceptions as to whether a CBS satisfies R. An example of such an NFR is

The new CBS shall be highly usable.

This particular NFR suffers also from a *vague boundary problem*.

When you replace the NFR

The new CBS shall be highly usable.

with the fitness criterion that requires an *experiment testing whether* At least 75% of the users shall judge the new CBS to be at least as usable as the existing CBS., how has the perception problem been solved?

With this fitness criterion, you may have solved the perception problem, but you have replaced one vague boundary problem with another.

(c) What is the replaced vague boundary?

(d) What is the replacing boundary?

(e) Describe a result of the fitness criteria experiment in which the vague boundary problem persists.

7. **[20 total marks]** Ambiguity

(a) Why is an ambiguity that happens while someone is speaking less of a problem than an ambiguity that occurs in writing?

(b) Consider the 12 variations of the sentence WUIM assigns IDs to UW persons., obtained by distributing only and also to different places in the sentence.

_____ 1. Only WUIM assigns IDs to UW persons.

_____ 2. WUIM only assigns IDs to UW persons.

_____ 3. WUIM assigns only IDs to UW persons.

_____ 4. WUIM assigns IDs only to UW persons.

_____ 5. WUIM assigns IDs to only UW persons.

_____ 6. WUIM assigns IDs to UW only persons.

_____ 7. Also WUIM assigns IDs to UW persons.

_____ 8. WUIM also assigns IDs to UW persons.

_____ 9. WUIM assigns also IDs to UW persons.

_____ 10. WUIM assigns IDs also to UW persons.

_____ 11. WUIM assigns IDs to also UW persons.

_____ 12. WUIM assigns IDs to UW also persons.

Do not worry about the truth of any of these sentences. The exercise concerns the *meanings* of these sentences. Some sentences may share a common meaning.

Items B through M below are sentences, each of which might clarify, interpret, explain, or exemplify the meaning of one of 12 sentences above. Item A is simply the claim that a sentence above is classified as meaningless. Please, in the underscore next to each sentence S above, write the letter labeling every of the 13 items below that clarifies, interprets, explains, or exemplifies S. Each letter other than possibly "A" appears in the underscores at most once, an underscore may contain more than one letter other than "A", and some letters may not be used at all. Yes, E and M are the same and H and L are the same.

A. Meaningless
B. Each of WUIM and another sytem assigns IDs to UW persons.
C. WUIM does not assign e-mail aliases to UW persons.
D. WUIM assigns e-mail aliases to UW persons.
E. WUIM assigns IDs to WLU persons.
F. WUIM cooks IDs for UW persons.
G. WUIM does not withdraw IDs from UW persons.
H. WUIM does not assign IDs to persons with no connection to UW.
I. WUIM withdraws IDs from UW persons.
J. No other system assigns IDs to UW persons.
K. WUIM removes IDs from WLU persons.
L. WUIM does not assign IDs to persons with no connection to UW.
M. WUIM assigns IDs to WLU persons.

(c) Not all of the sentences 1 through 12 would be true of the real world W after WUIM were implemented and deployed in place of the ID assignment system. Mark with a check mark (✓) all those sentences that would be true in W.

(d) Write a sentence that would be true in W that uses no also and at least two onlys, placed in the correct positions.

(e) Consider the highly ambiguous plural sentence.

All UW persons have their own unique Watlam IDs.

Write a less ambiguous, singular sentence that means what this ambiguous sentence is intended to mean, given what should be true in the real world at UW.

(f) In the real world as it should be at UW, what is wrong with the following sentence?

Each UW person has his or her own unique Watlam IDs.

(g) In the real world as it should be at UW, what is wrong with the following sentence?

All UW persons have their own unique Watlam ID.

8. **[10 total marks]** Cost Estimation

(a) What is the main reason that programmers underestimate the costs of *making a change* to software?

Recall the number of all possible connections between nodes for each number of nodes varying from 1 through 5,

nodes	connections
1	0
2	1
3	3
4	6
5	10

The number of connections c between nodes grows quadratically with the number, n, of nodes:

(1) $\frac{n^2}{2} - \frac{n}{2}$.

Between every pair of lines in a program there is a potential interaction between them. Therefore, the *potential* interaction in a program grows quadratically with the number of lines.

Recall that the COCOMO formula for effort is:

(2) $E = a \times L^b \times X$,

where L is the length of the program under development in KLOC (thousand lines of code), and where the exponent b ranges from 1.05 to 1.20.

(b) On the basis of Formula 1, what is the maximum that we can expect the exponent b to be?

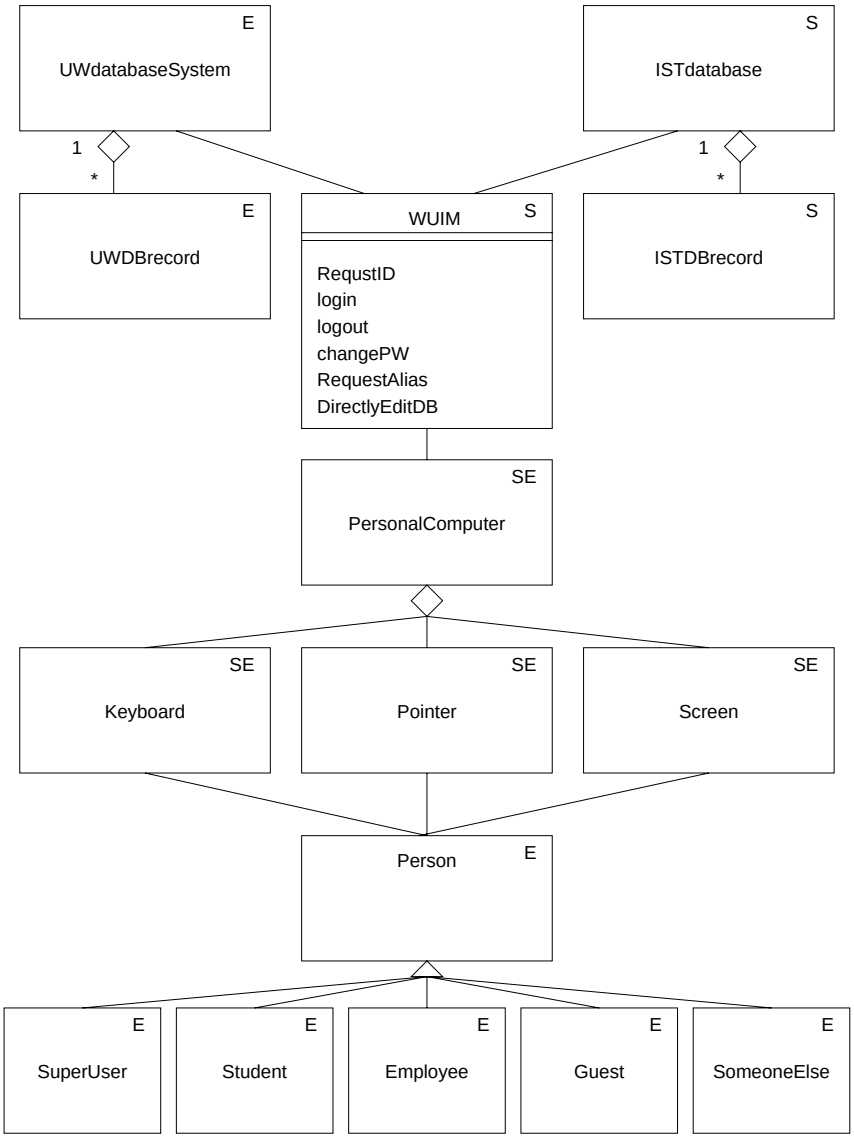
(c) What is the minimum that the exponent can be?

(d) Why (is the answer you gave the minimum that the exponent can be)?

(e) Why is the actual value of the exponent b neither the maximum nor the minimum?

(f) That the value of the exponent b ranges from 1.05 to 1.20 says what about the amount of actual interaction between lines of a program?

(g) In the domain model for WUIM shown below, mark with a "*" every entity of the diagram that would be subjected to function point counting to determine the total number of function points for WUIM.



(h) Unfortunately, unlike for the Turnstile domain model, there is not enough information in the domain model to do effective function point counting based solely on the domain model. What other artifacts about WUIM must be examined to do effective function point counting?

9. **[25 total marks]** State Machines and Linear Temporal Logic

   (a) "A state in a computation", hereinafter called "état", is defined as (complete the sentence)

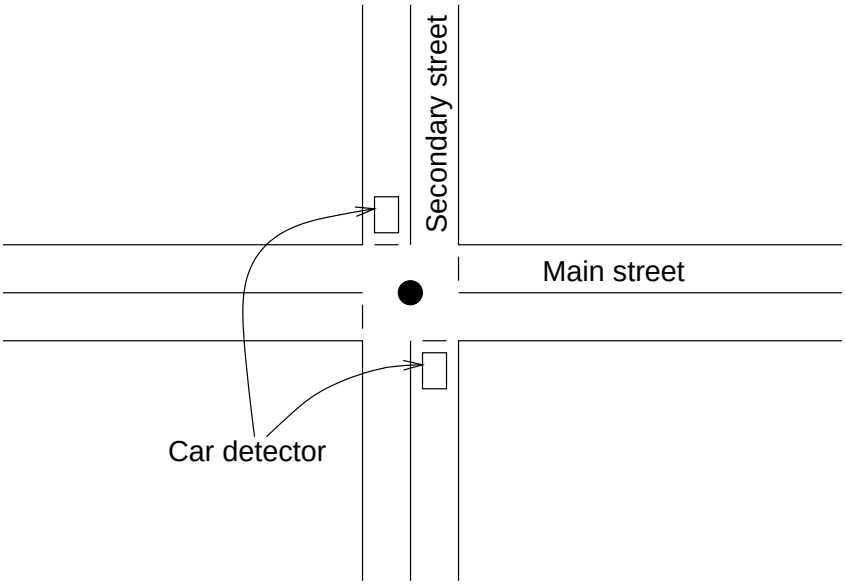
   "A state of a state machine model" is defined as (complete the sentence)

   (b) The key difference between an état in a computation and a state of a state machine model is in the numbers. For the computation of any substantial software system the number of états in a computation is potentially _____ while the number of states in a state machine model specification of the software system is _____.

   (c) Describe an état of a state machine model, M, in terms of states of M and whatever else is needed:

   An état of a computation of M is (complete the sentence)

   (d) You will be playing with a linear temporal logic and a state machine specification of a simple traffic light. The traffic light is at the intersection of a main street and a secondary street shown below:



The secondary street has behind its two stop lines car detectors each of which senses the presence of any car idling above it while stopped for a red light. The traffic light ensures that at any time, either only its two main street faces show green or only its two secondary street faces show green. In each face, the cycle is red, green, yellow, and back to red. As long as there are no cars idling above the two secondary street car detectors, the main street faces continue to show green. As soon as at least one car is idling above the two secondary street car detectors, the traffic light begins the cycles to end up showing green in the secondary street faces and showing red in the main street faces. The traffic light continues to show green in the secondary street faces for a time period known as the long time out (LTO), at which time the light cycles back to continuously showing green on its main street faces until such time as at least one car is idling above the two secondary street car detectors. In the cycling, each transition, e.g., such as from showing yellow to showing red, that is not governed by the long time out or the car detectors, is for a time period known as the short time out (STO). You will be considering a linear temporal logic and a finite state machine specification of this behavior. Each of the specifications has states (in the sense of a "state of a state machine") and events. In these specifications, the two time outs are caused by an external machine not described in the specifications.

In the names of the states,

M means "main street faces show",
S means "secondary street faces show",
R means "red",
G means "green",
Y means "yellow",
1 and 2 distinguish two otherwise identical states,
T means "timed", and
NT means "not timed".

Among the events,

D>=1C means "car detectors detect at least one car",
D0C means "car detectors detect zero cars",
STO means "short time out", and
LTO means "long time out".

Consider the following specification written in linear temporal logic:

$\Box$(MGSR-NT $\Rightarrow$ (MGSR-NT $\mathcal{W}$ (D>=1C)))
$\Box$((MGSR-NT $\wedge$ D>=1C) $\Rightarrow$ $\bigcirc$ MYSR)

$\Box$(MYSR $\Rightarrow$ (MYSR $\mathcal{W}$ (STO)))
$\Box$((MYSR $\wedge$ STO) $\Rightarrow$ $\bigcirc$ MRSR1)

$\Box$(MRSR1 $\Rightarrow$ (MRSR1 $\mathcal{W}$ (STO)))
$\Box$((MRSR1 $\wedge$ STO) $\Rightarrow$ $\bigcirc$ MRSG)

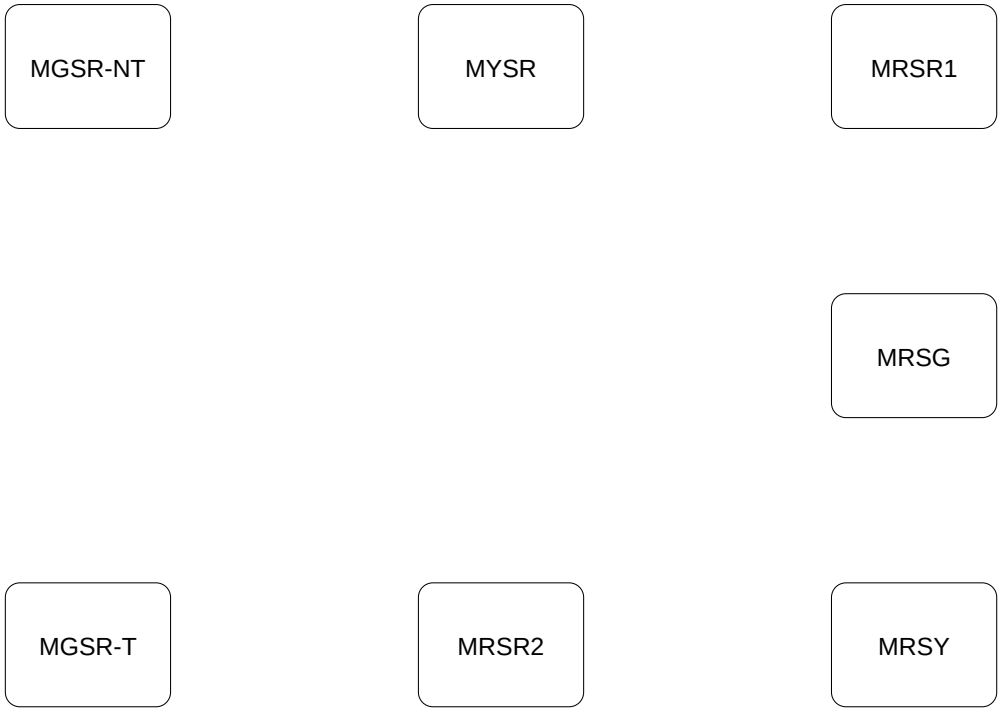
$\Box$(MRSG $\Rightarrow$ (MRSG $\mathcal{W}$ (D0C $\vee$ LTO)))
$\Box$((MRSG $\wedge$ (D0C $\vee$ LTO)) $\Rightarrow$ $\bigcirc$ MRSY)

$\Box$(MRSY $\Rightarrow$ (MRSY $\mathcal{W}$ (STO)))
$\Box$((MRSY $\wedge$ STO) $\Rightarrow$ $\bigcirc$ MRSR2)

$\Box$(MRSR2 $\Rightarrow$ (MRSR2 $\mathcal{W}$ (STO)))
$\Box$((MRSR2 $\wedge$ STO) $\Rightarrow$ $\bigcirc$ MGSR-T)

$\Box$(MGSR-T $\Rightarrow$ (MGSR-T $\mathcal{W}$ (LTO)))
$\Box$((MGSR-T $\wedge$ LTO) $\Rightarrow$ $\bigcirc$ MGSR-NT)

i. In the skeletal state machine diagram below, draw the finite state machine that is specified by the above linear temporal logic specification. Indicate that MGSR-NT is the starting state, and show all transitions specified by the temporal logic specification. If it turns out that this specification does not specify what is claimed, just carry on, making the state machine equivalent to the temporal logic specification above.



The state machine, as you have drawn it, is not complete. It is necessary after each time out of *any* duration to reset the external timer to begin the count down to the next two time outs. That is, there is a *single* timer that is reset by the operation, ST (*set timer*). Setting the timer begins the count down to both time outs, STO and LTO. After any setting of the timer with ST, at the end of the short-time-out duration, the event STO occurs and if that event is not detected by any state, then at the end of the long-time-out duration, the event LTO occurs. The timer continues to run after that.
There are two ways to have the timer setting operation, ST, done when it needs to be done: (1) as an action in transitions and (2) as an action to be perform on entry to a state.

ii. In your completed state machine diagram, mark with an asterisk ("*"), each transition that needs the action, /ST, in order to achieve the first way to have the ST done when it needs to be done.

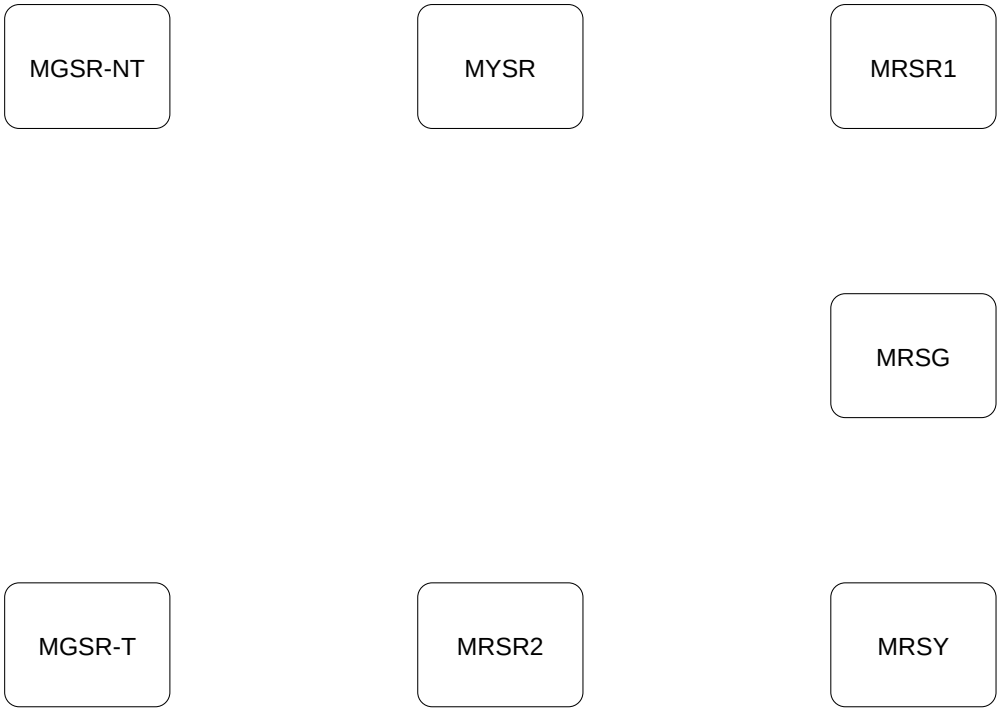
iii. In your completed state machine diagram, mark with a plus sign ("+"), each state that needs the entry action, entry/ST, in order to achieve the second way to have the ST done when it needs to be done.

(e) It is possible to add another transition leaving MGSR-T, going directly to MYSR, and bypassing MGSR-NT (but keeping MGSR-NT in the machine as the initial state). **On the state machine fragment given below**, in the standard notation, write the (event, condition, and action) that must be on this transition for the behavior of the whole state machine to be unchanged. Assume here that the ST action is done in the transition.

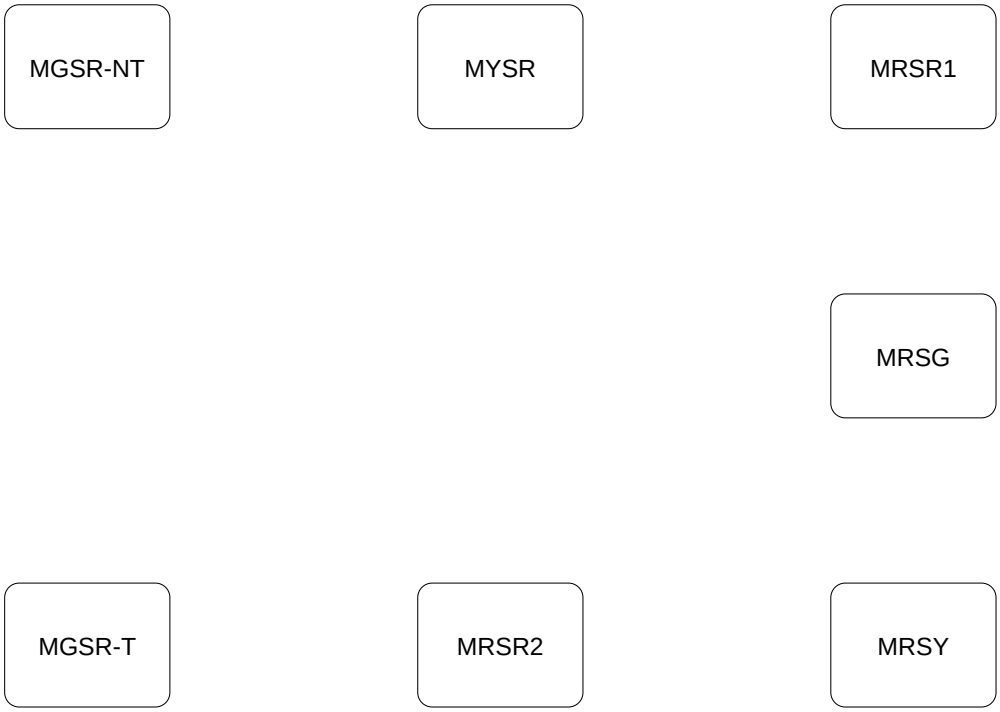
(f) Since each time out, LTO and STO, is guaranteed to happen, some but not all of the occurrences of $\mathcal{W}$ can be changed to $\mathcal{U}$. In the specification above, put an asterisk ("*") above each $\mathcal{W}$ that can be changed to $\mathcal{U}$ and still be correct.

We are going to be making two changes to the state machine specification of the traffic light. For each change, use the skeletal state machine diagram below it to indicate the *change from the original state machine that you drew*. That is fill in only the states and transitions that are affected by the change and show what they are now. *You do not have to draw the whole machine again; in fact it will make grading easier if you don't!* The two changes are independent from each other and should be enacted from the same original state machine that you drew.

(g) The first change is to add a variable, ss1 or ss2 to each of the two street sensors in the secondary street. Each street sensor variable, ss1 or ss2, shows the current number of cars above the street sensor. The change in behavior is that the duration of a green light facing the secondary street is equal to the short time out, STO, multiplied by the maximum number of cars above the two sensors at the time the light changes to green.



(h) The second change is to make it so that when a light changes to red in a direction, the light sends a focused signal stopAllCarsOnThisStreet in the direction of the red light, that tells every e-car that receives it to do a safe stop for the red light.



(i) List two real-world difficulties with actually implementing the second change.

i.

ii.

10. **[15 total marks]** The Requirements Iceberg

(a) Everyone complains about how insecure the Internet and e-mail are and are trying to add security to the Internet, but are failing. This is because the original requirements, developed in the late 1960s and early 1970s, for the ARPAnet, which later became the Internet, was that it be completely open. Anyone sitting anywhere on the net was to be able to use any other site on the net as if he or she were logged in at the other site. In other words, the ARPAnet was required to be open and essentially insecure, and the implementers of the initial, research version of the ARPAnet did a damn good job of implementing the requirements!

Adding security to the Internet ultimately fails because there is always a way around the add on security through the inherently open Internet.

To get a secure Internet, we have to rebuild the whole thing from requirements up.

Explain what happened between the early 1970s and now that caused the Internet to have a security requirement that it did not have originally. Your explanation must be in terms of the following concepts:

i. the feedback loop from the real world to the software that occurs with E-type systems,
ii. that security has to be required from the start and cannot be added to an existing system,
iii. the recommended treatment of throw-away prottoypes, and
iv. the relation between the cost to fix a defect and the lifecycle stage in which the defect is detected.

(b) On this basis what needs to be true about lack of security for it to be cost effective to rebuild the Internet from the ground up with security being required from the beginning?

(c) It has been said that both mathematicians and programmers build formal models of real-world phenomena. For example, the mathematical theory of aerodynamics describes how an aircraft flies, and also the avionics software on an aircraft describes how an aircraft flies. Nevertheless, there is a fundamental difference between these two formal models. What is it?

(d) Identify the customer and the client for the following items of software:

Software	**Customer**	**Client**
Microsoft Office		
Amazon's Inventory SW developed in house		
Amazon's Inventory SW bought off the shelf from SAP		
Opensource Linux		

(e) In one "if — then" sentence, what is the main take away lesson of the lecture titled *Requirements Determination is Unstoppable*?

(f) List three problems with the project to develop the Bradley tank that contributed to the huge cost overrun.

i.

ii.

iii.

(g) In what ways did the fact that the people involved were in the military affect what happened?