1.264 MIDTERM EXAM

FALL 2013

NAME ____

Exam guidelines:

1) 80 minutes are allowed to complete the exam.
2) Open notes; open book.
3) There are 7 questions (100 points) and 10 pages (including this one) in the exam booklet.
4) No laptop computers, calculators, cell phones or messaging devices are allowed. Please turn off any that you have brought.
5) Please write legibly – you are welcome to use both sides of the paper; we can provide additional paper if necessary.

PART I: SOFTWARE PROCESS (35 POINTS; SUGGESTED TIME: 25 MINUTES)

1. **Read the case study below and identify five errors in the execution of the project: circle a phrase from the case to support each error you identify, write a number next to each circled phrase, and describe each error with a phrase or two next to the corresponding numbers on the following page. If the error is an activity that the team did not do and thus is not mentioned, also include it.**

RailTrak is a corporation delivering commuter rail passenger service. They have received complaints from passengers about not being able to find a seat for their daily commute during the peak hours and that they're willing to pay more to get a reserved seat. The RailTrak management has decided to launch an online reservation system for their commuter rail services.

Although there are many off-the-shelf reservation systems for airlines, the RailTrak management has decided to implement an in-house reservation system because rail services are different from flights in many respects. Trains make many stops along a route, and seats can be reused by many different passengers on different route segments. The system will have both reserved seats and unreserved seats. To avoid the hassle of a reserved seat being taken by someone without a reservation, LED lights will be installed on every seat in the car to indicate the status of a seat, as well as an on-train server that will turn on/off the lights automatically based on the reservation status. The LEDs and server will be installed by a vendor within 8 months but the online reservation system will need to exchange information with it.

The executive board meets with their software manager, Paul, and proposes that he manage the design and implementation of the online reservation system within 8 months so it will be ready to work along with the vendor's control system. Upper-level management has given him a team of three software developers. Paul estimates that the project will take ten months with a team of four, based on his experience and intuition.[1] He tries to negotiate a more reasonable development schedule but management insists that the new system be launched along with the vendor's control system, and promises to give him the top developers in the company to meet the schedule. Paul agrees to the 8 month schedule.[2]

Paul's team consists of Lisa, Tom and Jerry, the top software developers in the company. While the team's past project experience is mainly in C++, Lisa suggests that the system be written in Python, a language that she just learned and is supposedly better suited to developing

[1] **Estimate schedule based on "experience and intuition" (Should use a model to estimate the size and effort)**

[2] **Overly optimistic schedule/ No effective negotiation (Should negotiate with board for a realistic schedule. If schedule cannot bend, product size and resources must.)**

distributed systems. Paul decides to give it a try in order to meet the compressed schedule. [3]He also plans to use the spiral model. With the eight month schedule, he plans to do two spirals of four months each.[4]

Since the data model and design seem appropriate after two weeks of requirements and high-level design, Paul decides to skip detailed design and move to implementation to speed up the software development.[5] Meanwhile, the vendor is busy installing LEDs and servers on commuter trains, and repeatedly postpones design meetings with Paul's team on the integration of two systems. Paul decides to go on with his first spiral of development anyway.[6]

At the end of month 3, the vendor finally has a substantive meeting with Paul's team and they note that a fundamental part of the software's design is incompatible with the way the on-train server works. The vendor states that Paul's team will need to change the design of the reservation system. Paul decides to release his software in spiral 1 without compatibility with the vendor's system anyway and make the changes in spiral 2.[7]

As Paul begins the development process for spiral 2, the schedule begins to slip because of the significant changes in system requirement and designs. By month 5, Paul is 1 month behind schedule and he decides that more developers will expedite the process. [8]He goes to upper-level management to ask for additional developers, but with recent layoffs there are none available. Paul decides to hire three contracted software developers to supplement his team. By month 6, the team is 6 weeks behind schedule and Paul decides to reduce the scope of the system to meet the 8 month deadline. By month 8, Paul's team only has 2 weeks to integrate with vendor's control system and they discover more incompatibilities between the two systems.[9]

At the end of moth 8, Paul's team delivers a system with limited functionality which fails to integrate with vendor's control system. The executive board is disappointed.

[3] **Silver bullet syndrome. Overestimated the benefits of a new and unfamiliar technology (Should use the technology the team is familiar with, or should use a different schedule estimation level, e.g. use nominal instead of efficient)**

[4] **2 Spirals may not be enough (2 is fine, but 3 may be better)**

[5] **Not enough design and shortchanged upstream activities (Should have more detailed design before rushing into implementation.)**

[6] **Lack of input and integration (Should collect enough vendor input and make appropriate integration design between two systems.)**

[7] **Plan to catch up later (Should have better risk management.)**

[8] **Request to add more developers, which would make the project later (Rescope the project to meet the schedule or negotiate with the board for an extended schedule.)**

[9] **Late integration (Should have included integration as early as possible)**

Write a phrase or two describing each error, making sure that the numbers below correspond to the numbers that you wrote next to the circled text above. (20 points) (Suggested time: 15 minutes)

ERROR 1:

ERROR 2:

ERROR 3:

ERROR 4:

ERROR 5:

2. **Briefly describe how to correct each of the 5 errors. (15 points) (Suggested time: 10 minutes)**

ERROR 1:

ERROR 2:

ERROR 3:

ERROR 4:

ERROR 5:

PART II: DATA MODEL AND UML (38 POINTS; SUGGESTED TIME: 30 MINUTES)

3. Draw a normalized entity-relationship diagram that corresponds to the specification and business rules in the box below. Follow these five steps while drawing your model on the next page. (25 points. Suggested time: 20 minutes)

a. Draw a single box for each entity: give each an appropriate name. (5 points)

b. List the attributes in the box for each entity (data types are not necessary). (5 points)

c. Indicate the primary key for each entity by writing "PK" next to the appropriate attribute(s). (5 points)

d. Draw all relationships between the entities in the model. Indicate foreign keys by writing "FK" next to attributes that are foreign keys. For many-to-many relationships, show the associative/intermediate entity explicitly; don't use the many-to-many notation. (5 points)

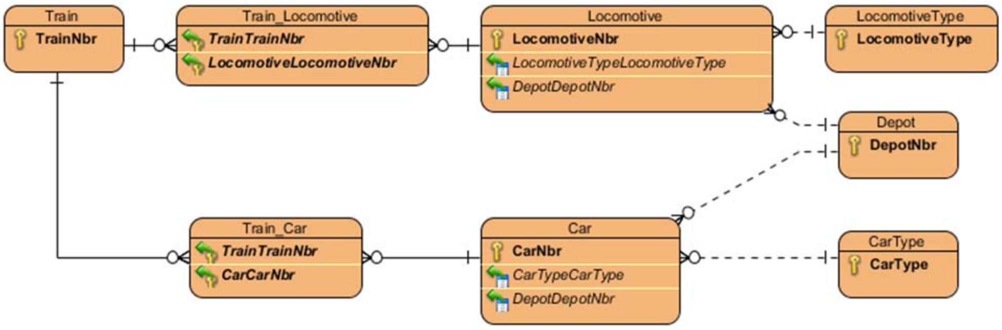
e. Indicate the cardinality of each relationship (one-to-many, many-to-one, etc.) Any time a cardinality of "many" occurs at either end of a relationship, you do not have to specify whether that cardinality is "zero-or-more" versus "one-or-more." Use crow's-foot notation; if you use any other notation, define it. (5 points)

You are hired by RailTrak to design their asset management system. The system will manage the locomotives and cars which are put together into trains. The system will also manage the maintenance of locomotives and cars at depots.

- Each train is pulled by one or more locomotives and has one or more cars. Train numbers are unique.
- Each locomotive has a unique number and can either be electric or diesel.
- Each car has a unique number and can either be single-level or double-deck.
- A locomotive or car will usually be assigned to multiple trains, since trips are short. Cars and locomotives are assigned to the same trains every day.
- Locomotives and cars are serviced at depots. There are multiple depots.
  a. Each depot has a unique number and can serve one or more locomotives/cars.
  b. Each locomotive is serviced at one depot.
  c. Each car is serviced at one depot.

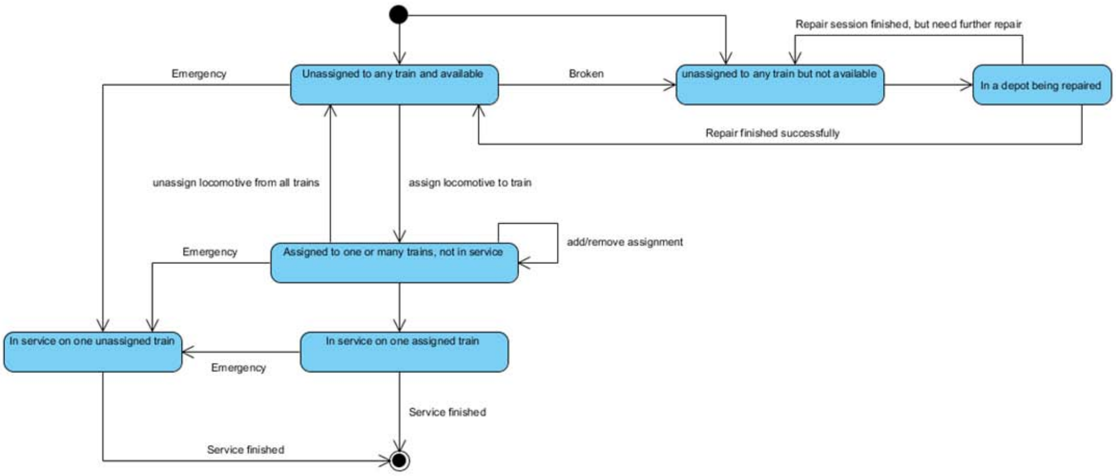
To repeat: You only need to turn in one drawing that includes all the elements listed in steps a through e above.

(**DRAW YOUR DATA MODEL HERE)**



4. **Draw a state diagram for a locomotive. (13 points. Suggested time: 10 minutes)**
   A locomotive can be:
   a. **Unassigned to any trains, but available to be assigned**
   b. **Unassigned to any trains, but unavailable to be assigned because it needs repair**
   c. **Assigned to one or more trains**
   d. **In service on one of its assigned trains**
   e. **In service on a different train than assigned due to emergency or unforeseen operational change**
   f. **In a depot being repaired**

Draw a diagram with a set of states and transitions that models the possibilities listed above. You may combine or split states. Make reasonable assumptions about transitions.


Repair session finished, but need further repair
Emergency
Unassigned to any train and available
Broken
unassigned to any train but not available
In a depot being repaired
Repair finished successfully
unassign locomotive from all trains
assign locomotive to train
add/remove assignment
Emergency
Assigned to one or many trains, not in service
In service on one unassigned train
In service on one assigned train
Emergency
Service finished
Service finished

PART III: DATABASE AND SQL (27 POINTS; SUGGESTED TIME: 25 MINUTES)

Suppose that you have implemented the database described in Part II: the tables, attributes, and relationships in the database correspond exactly to the model that you have defined. You are now asked to write the following SQL queries against your database. Briefly define any variable you use that isn't obvious.

5. **List the train number and number of cars assigned to each train, showing the trains with the most cars first. (9 points)**

```
SELECT TrainNbr, COUNT(*) AS NumOfCars
FROM Train_Car
GROUP BY TrainNbr
ORDER BY NumOfCars DESC
```

6. **List the train number of all trains that have one or more double deck cars that are serviced at depot 19. List each such train number only once. (9 points)**

```
SELECT DISTINCT TrainNbr
FROM Train_Car
INNER JOIN Car
ON Car.CarNbr = Train_Car.CarNbr
WHERE DepotNbr = 19 AND CarType = 'Double-Deck'
```

7. **List the locomotive numbers, and number of assigned trains, of locomotives serviced at depot 22 that are assigned to more than 3 trains. (9 points)**

```
SELECT Locomotive.LocomotiveNbr, COUNT(*) AS NumOfTrains
FROM Locomotive
INNER JOIN Train_Locomotive
ON Locomotive.LocomotiveNbr = Train_Locomotive.LocomotiveNbr
WHERE DepotNbr =22
GROUP BY Locomotive.LocomotiveNbr
HAVING COUNT(*) > 3
```

1.264J / ESD.264J Database, Internet, and Systems Integration Technologies
Fall 2013