

1.264 Midterm Exam Solutions
Fall, 2004

1. Software process. (33 points)

a. List at least 5 errors made by this development team in their execution of this project. (11 points)

- i. No resource estimates (function points, lines of code, schedule months, person months) were made at the start of the project. Chris gave an off-the-cuff estimate when asked at the first meeting.
- ii. No uncertainty was communicated to the steering committee. Chris gave a point estimate, and he didn't state any assumptions on the size of the project, complexity of requirements, etc.
- iii. No discussions of tradeoffs in functions or team size to meet the 6-month date were held. Chris should have done this as requirements became complex, and even at the start of the project. No principled negotiation occurred, and requirements gathering was not managed aggressively, as the short timeframe required.
- iv. Chris did not communicate the true schedule as the project progressed. The real estimates were not given, and no proactive information was given before there was a problem. If requirements took twice as long as expected, so would every other part of the project, and this was not addressed. The project had poor visibility with its management.
- v. The team did not use the spiral model (or other appropriate model such as evolutionary prototyping, etc.). They used a waterfall, which is inappropriate for rapid development, which this project clearly required.
- vi. The design was not coordinated properly since the pieces did not fit together. This is poor management and possibly poor use of tools. The data model should have been frozen early.
- vii. Integration was not begun until too late in design and coding. Leaving the hard parts until the end is a mistake; they won't integrate properly.
- viii. QA time was very inadequate, leading to many bugs.

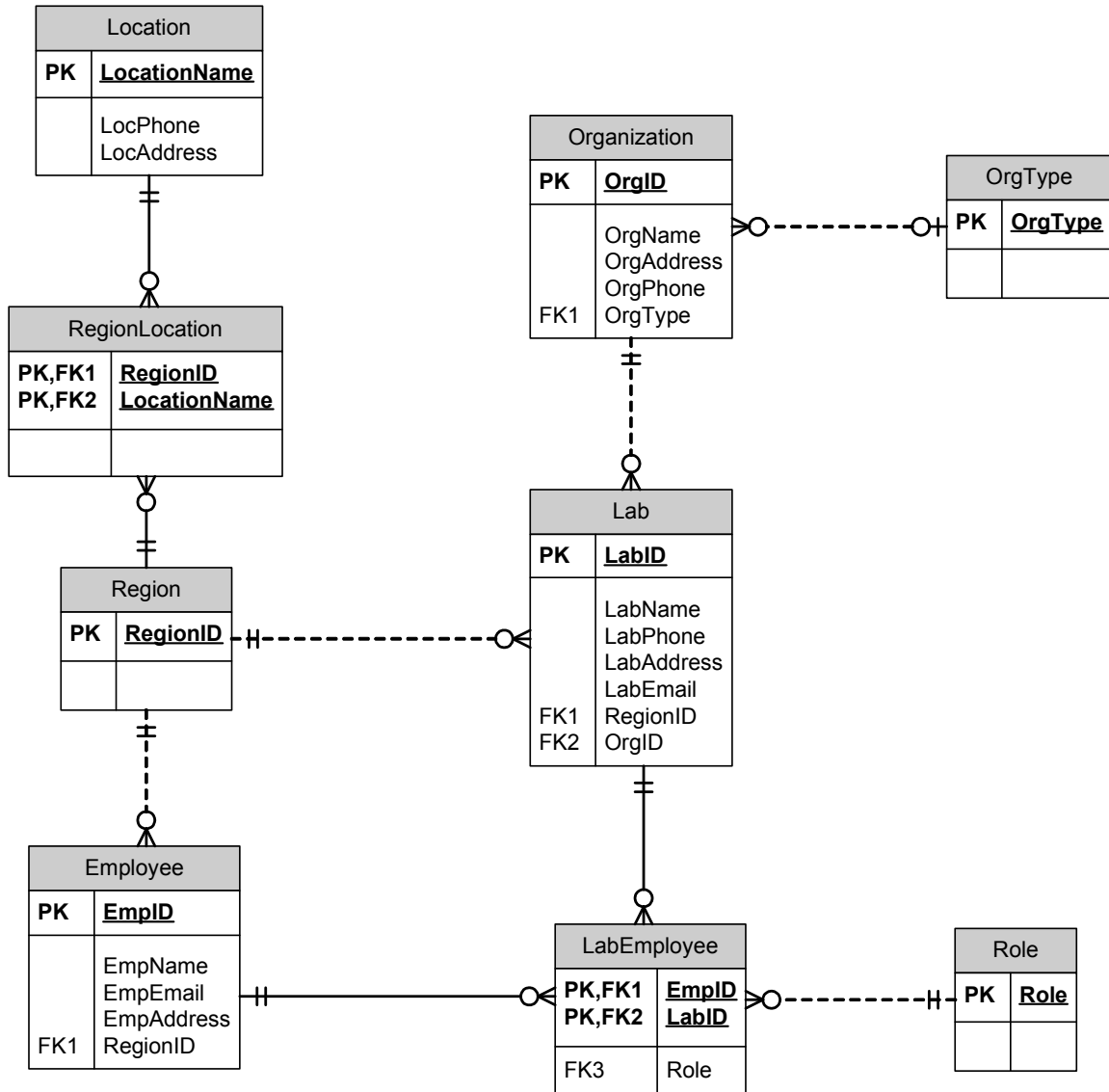
b. Outline the key steps in the resource estimation that the team should have used to avoid these errors/problems. List at least 5 steps; describe each in 1-2 sentences or phrases. (11 points)

- i. Quickly estimate the number of Web pages, database tables and external interfaces to be built. Estimate the complexity of each Web page, table, interface.
- ii. Estimate the function points for each.
- iii. Based on the type of software to be used (database, Java, etc.) for each part of the system, estimate the lines of code.
- iv. Look up the schedule and person months for the appropriate project type (business) and software maturity (nominal, at best).
- v. Apply a range of uncertainty based on the stage of the project.

c. Outline the key stages in the software development process that the team should have used to avoid these errors/problems, and roughly how long each stage should take. List at least 5 actions; describe each in 1-2 sentences or phrases. Give a rough timeline to reach a shippable system at the 6 month point. (11 points)

- i. Use the spiral model (or other evolutionary model)
- ii. Use 3 month spirals to ensure that a system can be shipped at the 3 month point (a very simple one, but this ensures integration) and at the 6 month point (also likely to be very simple)
- iii. Do requirements in the first 3 weeks of spiral 1, instead of letting them go for 3 months. You can start coding with a rough set of requirements for the first spiral, since you'll be finding all sorts of other problems with the data, pages, interfaces, etc. Requirements should be roughly $\frac{1}{4}$ of the effort in one spiral
- iv. Do design in the next 3 weeks. It should also be $\frac{1}{4}$ of each spiral. Design went on for 6 months (!) in this project, as long as the total project was supposed to take.
- v. Build an initial prototype, similar to what you did in 1.264, in about 4-5 weeks. Make sure everything integrates. You can do better than in 1.264 because you have more time and more knowledge.
- vi. Test, review and debug for 2 weeks or so in spiral 1 and show an initial product to the steering committee at the 3-month mark. If it takes until the 4-month mark, that's not a major problem, though it will tell you that the product won't be complete until 7 months, since 3 month spirals are usually the shortest possible.
- vii. The committee, if it sees a working prototype at 3-4 months, will be more supportive, can help limit requirements changes, etc.

2. Data model (34 points)



3. Database. (33 points)

- a. List the laboratory names that each employee supports, for all employees. Display the employee ID, employee role and lab name. Order by employee ID.**

Implicit:

```
SELECT LabEmployee.EmpID, LabEmployee.Role, Lab.LabName
FROM Lab, LabEmployee
WHERE Lab.LabID= LabEmployee.LabID
ORDER BY LabEmployee.EmpID;
```

SQL-92:

```
SELECT LabEmployee.EmpID, LabEmployee.Role, Lab.LabName
FROM Lab INNER JOIN LabEmployee ON Lab.LabID = LabEmployee.LabID
ORDER BY LabEmployee.EmpID;
```

- b. Delete employees that are supporting no laboratories (i.e., whose employee IDs are not present in the laboratory-employee table).**

Implicit or SQL-92:

```
DELETE Employee.*
FROM Employee
WHERE Employee.EmpID NOT IN (SELECT LabEmployee.EmpID FROM
LabEmployee);
```

- c. List the storage location names and phone numbers that can supply each lab. List the lab name being supplied.**

Implicit:

```
SELECT Location.LocationName, Location.Phone, Lab.LabName
FROM Lab, Region, RegionLocation, Location WHERE
(Region.RegionID = Lab.RegionID AND
Region.RegionID = RegionLocation.RegionID AND
Location.LocationName = RegionLocation.LocationName)
```

SQL-92:

```
SELECT Location.LocationName, Location.Phone, Lab.LabName
FROM Lab
INNER JOIN Region ON Region.RegionID = Lab.RegionID
INNER JOIN RegionLocation ON Region.RegionID = RegionLocation.RegionID
INNER JOIN Location ON Location.LocationName = RegionLocation.LocationName
```