CS577A Homework #2

Due Sep. 4, 2002

30 points

Reading: All of Brooks "The Mythical Man Month" or MMM. If you do not have the book yet, we will post the relevant sections on the class web site.

Important Note: This assignment will take considerable time. It is suggested that you start on this assignment as soon as possible.

Problem 1: A Simple Mathematical Model for Brooks Man-Month [*5 points*]

Consider the following mathematical model which is one way of formalizing some of the concepts in the MMM. This formula is not intended to be a predictive model. Rather, it is useful for illustrating and analyzing the concepts expressed in the Brooks book.

$$T = \frac{E}{n^p} + bn + c\frac{n(n-1)}{2} + pS$$

T = calendar effort (in months)

E = total effort (in man-months)

n = number of people

p = degree of partitionability (where 0 = not partitionable, and 1 = totally partitionable)

b = base training time (months per person)

c = average intercommunications overhead between two people (in months/person*person)

S = base sub-task communication overhead (in months)

Explain and derive (briefly!) where each term comes from. Provide explicit references (such as page numbers and quotations) from the MMM book.

Problem 2: Brooks Mythical Man-Month

Show by setting the parameters of E,p,c,b,S appropriately that the mathematical model in problem 1 matches that which is described by the indicated figures contained in the MMM. Either plot a graph that matches the figures indicated or explain qualitative characteristics in such a way that it is clear that the model has the same features. Provide a brief explanation of what the values for the parameters represent. You may find using Mathematica useful in completing the remainder of the questions. Also, when you are having trouble, try creating an explicit example by choosing particular values for the parameters. Avoid writing any programs for this assignment.

a) Fig. 2.1 (hint: perfectly partitionable task, no training, no intercommunications overhead, no subtask communication overhead) [*2 points*]

b) Fig. 2.2 (hint: un-partitionable task, no training, no intercommunications overhead, no subtask communication overhead) [*2 points*]

c) Fig. 2.3 (hint: perfectly partitionable task, no training, no intercommunications overhead, subtask communication overhead) [*2 points*]

d) Fig. 2.4 (hint: semi-partitionable task, training, intercommunications overhead, subtask communication overhead) [*2 points*]

e) Show that the model implies Brooks Law: "*adding people to a late project makes it later*." [7 *points*]

Hint for (e): Consider a project that started with n people which is late, but has completed some fraction $0<d<1$ of the total required effort E. Compare the estimate $T1$ of what it would take to complete the remaining effort $(1-d)E$ using the Brooks formula just with the original n people versus the estimate $T2$ from adding m more people. You may assume that for $T1$, people no longer have to be trained, but for $T2$, the m new people will have to be trained. We may interpret Books law as "*adding late people to a late project may make it later*" so that what needs to be shown is that there is at least one case (given reasonable assumptions) where $T2 > T1$. So either find such a case using actual numerical values, or better, show this by doing some algebra that for m and n sufficiently large $T2 > T1$.

Problem 3: Brooks Man-Month and CS577

For this problem we assume the characteristics of our class projects. Let us say for CS577 we have about 4 man-months to develop a product, so $E=4$ man-months. All the projects must be complete by the end of the semester, so we have a *hard deadline*. There are generally five people on a team, so $n=5$. We will spend approximately 15 days (note that this is likely an overestimate since lectures, recitations, and homework probably do not account for full days) learning MBASE and other "training" needed to do a project, so $b=1/2$ month.

a) **Need for effective communication:** What is the maximum intercommunication overhead that can occur (value of c)? Assume the overhead of assimilating subtasks is fixed and will take about a week, so $S=1/4$ month. Is the condition (value for p) for which this maximum is achieved reasonable (why or why not)? With respect to the maximum value of c you found, what does this say about the importance of efficient communication between your team members? **Warning**: the mathematical model is only a first order approximation and erroneous results may occur for certain values of the parameters. [*3 points*]

b) **Why five people on a team:** Through the use of MBASE in software engineering courses, we have estimates for ***c***, ***S***, and ***p***. Assume that it takes on the average a day to communicate properly to another team member, so ***c*** **= 1/31** month. The overhead of assimilating subtasks will take about a week, so ***S=1/4*** month. Assume that the component/object oriented nature of MBASE gives ***p=9/10***. Justify the choice of 5 people per CS577 MBASE team. [*3 points*]

c) **Optimum team size:** Using the same parameters as in (b), is 5 the optimum team size? If so, why, if not what is the optimum team size and why isn't it used in CS577? [*4 points*]

Extra Credit: Effects of Object/Component Oriented Development

The component/object-oriented approach to product development is used to create a more natural partitioning of the system. As such, it incurs less intercommunication overhead (due to encapsulation) and provides an easier assignment of development subtasks. The tradeoff is there will be more parts and the overhead on putting these parts together in a faithful way.

a) What parameters in the model does this effect and how is it effected?

b) If the task at hand is totally un-partitionable, how does the component/object-oriented approach effect it?

c) If the task is totally partitionable, the component/object-oriented approach will make the task "hyper- partitionable". How does this effect the task effort?

d) Is there a "point of diminishing returns" to the component/object approach? That is, as the partitionability of a task increases, is there a point at which partitioning the task further increases the time needed to complete the task (given a fixed number of people working on it)?