26.2 Develop a list of software characteristics (e.g., concurrent operation, graphical output) that affect the complexity of aproject. Prioritize the list.

Sometimes, complexity arises from a poorly established interface between the customer and the software developer. Discounting that, the following technical characteristics should be considered:

- real-time attributes
- multiprocessing requirement (concurrency)
- nature of the algorithm
- requirement for recursion
- nature of input
- determinacy of input
- nature of output
- language characteristics
- knowledge/experience of staff on application.

26.4 Assume that you are the project manager for a company that builds software for household robots. You have been contracted to build the software for a robot that mows the lawn for a homeowner. Write a statement of scope that describes the software. Be sure your statement of scope is bounded. If you're unfamiliar with robots, do a bit of research before you begin writing.Also, state your assumptions about the hardware that will be required. Alternate Do a functional decomposition of the robot software. Estimate the size of each function in LOC. Assuming that your organization produces 450 LOC/ pm with a burdened labor rate of $7000 per person-month, estimate the effort and cost required to build the software using the LOC-based estimation technique

For our purposes here, we do a simple functional decomposition:

- user interaction (2400)
- sensor monitoring (1100)
- message display (850)
- system configuration (1200)
- system control [activation/deactivation] (900)

LOC estimates for each function are noted in parentheses. A total of 6450 LOC are estimated. Using the data noted in the problem:

6450 LOC / 450 LOC/pm = 14.3 pm

Cost: 14.3 pm * $7,000/pm = $100,000 (approximate)

26.5 Use the COCOMO II model to estimate the effort required to build software for a simple
ATM that produces 12 screens, 10 reports, and will require approximately 80 software components. Assume average complexity and average developer/environment maturity. Use the application composition model with object points.

Using the weightings (From Table 26.6 of Pressman) the unadjusted object points are

Object type	**Complexity weight**		
	Simple	**Medium**	**Difficult**
Screen	1	2	3
Report	2	5	8
3GL component			10

object point = 12 * 2 + 10 * 5 + 80 * 10 = 874

If we assume 80% reuse

NOP = (object points) * [(100 - %reuse)/100]
= (874) * [(100 – 80)/100] = 874 * 0.2 = 174.8

Using the weightings from (Table 23.7 of Pressman) for nominal developer experience

PROD = 13

The estimated effort in person months is

estimated effort = NOP/PROD = 174.8 / 13 = 13.45

26.6 Use the software equation to estimate the lawn mowing robot software. Assume that Equation (26.4) is applicable and that P = 8000.

assuming B=0.16 and P = 3,000

$t.min = 8.14 * (LOC / P)^{0.43}$ =
$= 8.14 * (6450/3000)^{0.43} = 8.14 * 1.38 = 11.31$ months

t = 11.31 month / 12 months/year = 0.94 years

$E = 180 * B * t^3$
$= 180 * 0.16 * (0.94)^3 = 24$ person months

26.7 Compare the effort estimates derived in Problems 26.4 and 26.6. What is the standard deviation, and how does it affect your degree of certainty about the estimate?

The estimates are all within a reasonable range with the software equation representing the most conservative estimate. Note however, that the project is small for software equations in Section 26.7.3, therefore putting this result in doubt.Given the values computed in the above problems, it likely that an estimate in the range 13 - 18 person-months would be appropriate for this project.

26.8 Using the results obtained in Problem 26.7, determine whether it's reasonable to expect that the software can be built within the next six months and how many people would have to be used to get the job done.

The software equation predicts "no" but we may be outside its bounds. Using the original COCOMO model (not COCOMO II),

D = 2.5 E ^ 0.35
= 2.5 * 16.92 ^ 0.35
= 6.7 person-months

It appears that 6 months is aggressive, but probably possible (with the equivalent of 3 people available to work on project) given the results of COCOMO and the size/complexity of the project.

26.11 It seems odd that cost and schedule estimates are developed during software project planning—before detailed software requirements analysis or design has been conducted. Why do you think this is done? Are there circumstances when it should not be done?

Costs and schedule are estimated early because such information is demanded (by upper management) as early as possible. If a project is extremely complex with high technological risk and a fixed price proposal is to be submitted, costing of the project should (if possible) be delayed until after requirements analysis. Note: the cost of requirements analysis alone can be estimated early.

26.12 Recompute the expected values noted for the decision tree in Figure 26.8 assuming that every branch has a 50–50 probability. Would this change your final decision?

No it will still be cheapest to buy the system.

expected build cost = 0.5 * $380K + 0.5 * $450K = $415K
expected reuse cost =
0.5 * $275K + 0.5 * [0.5 * $310K + 0.5 * $490K] = $337.5K
expected buy cost = 0.5 * $210 + 0.5 * $400K = $305K
expected contract cost = 0.5 * $350K + 0.5 * $500K = $425K

Extra

Consider a complex, real-time software project estimated at 33,000 LOC, 12 person-years of effort. If eight people are assigned to the project team, the project can be completed in approximately 1.3 years. If, however, we extend the end date to 1.75 years, What will be the effort?

Consider Software equation derived from PNR curve

$E = L^3 / p^3 t^4 = \sim 3.8$ person-years

By extending the end date by 6 months we can reduce the number of people from 8 to 4!