Software engineering

Editor:

I think your article on “The Need for Software Engineering” in the February issue of Computer is one of the best industry-wide surveys yet published ... [Its] publication is going to be a positive motivating source towards adopting the newer methods.

Capers Jones
IBM

Editor:

I have just finished reading your article, “The Need for Software Engineering,” in the February issue of Computer. Your article was quite informative, but at the same time somewhat disturbing. The disturbing element has to do with terminology and perspective.

The article’s title pre-implies a discussion of software engineering, but all I kept reading about was “programming” and “programs” which is only a part of the overall product, namely the system. It is vitally important that we move to the broader perspective of the system rather than the limited concept of programs if we are ever to solve the software development problem. The various structured systems techniques that are now being developed do offer some hope for an overall solution.

There is definitely a glimmer of hope in your article since you seem to be on the verge of realizing that programming is only a part of the development process. Table 1 [in the article], although cast from the programming perspective, does show the all-important concept of design (not program design, but system design). This design stage, in the structured setting, is generally being broken into logical design and then physical design from which programs can be developed. The point is that in the initial stages (problem definition, requirements analysis, and logical design) physical or implementational aspects must be put off as long as practicable. Once the logical design is done (from the point of the all-important users!) the software engineer can go to work on the physical design, implementation, etc.

I realize that ours is a young, struggling discipline, but the broader systems perspective seems to me to be critical. In the various organizations with which I have contact, it appears that the modern programming practices (Table 2) are doing quite well although they take different forms within various organizations. The real problems and bottlenecks appear to be at the systems level. It is the systems analysts who are tearing their hair out trying to perform a multitude of tasks that require varied skills and abilities. As I see it, the problem with the modern programming practices is implementation, while at the systems level there is much yet to be developed.

The main point is not to equate the phrase “software engineering” to that of “programming.” Even using the definition that you quote in your article, the deliverable product and the processes used to develop the deliverables encompass much more than programs and programming.

I would have been much more comfortable in reading your article if it had reported on the progress and problems within the programming phase of software engineering and had gone on to indicate how much more needs to be done with the other phases of the process.

Duane C. Abbey
Office of the State Comptroller
Des Moines, Iowa

About the cover

The cover this month illustrates a typical transformation used in program analysis: conversion from a multiple-entry/multiple-exit structure to a single-entry/single-exit structure. The latter are important for testing because resulting test-cases are simpler, more reliable, and easier to repeat. Cover design: Jay Simpson.