Team Considerations

Another consideration in larger programming tasks is working in teams. Depending on the size of the programming task, teams may range from two or three members to hundreds. Working with teams magnifies the importance of the considerations we have discussed so far. However, it also introduces its own challenges and benefits.

One issue which becomes much more important when programming in a team is the use of a good revision control system (e.g., Git, Subversion, or Mercurial). These systems facilitate multiple people editing the same code base at once, as well as giving the ability to find out who made which changes in the code, and track older versions. The rest of this lesson discusses Git in more detail.

Another significant issue in programming with larger teams is integration—putting the pieces together. Programmers may write and test their own modules, determining that they work in isolation, only to discover that they do not work together with the other programmers' modules when they attempt to integrate. Problems can arise from misunderstood (or imprecisely defined) interfaces and differing expectations. Many programmers underestimate the time required for integration, which can actually be quite significant.

Of course, programming in teams has benefits too—primarily that the team can accomplish more in the same time than any individual can. Of course, the simplest way this works is by the fact that multiple people are programming at the same time. However, there are other benefits to working in teams, arising from the old principle of "two heads are better than one."

One nice way to work in partnerships is *pair programming*. In pair programming, two programmers share a computer with one "driving" and the other "navigating." The driver controls the keyboard and actually writes the code. Meanwhile, the navigator watches the code being written and thinks about what is happening. The navigator is responsible for checking for problems with code—whether they are just local mistakes, or failure to respect global invariants. The programmers may trade roles whenever they feel it is appropriate.

Completed