David Lambertson

**Software Engineering Summary**

**Important Points from Article**

* “The software Engineering Institute has developed a comprehensive process meta-model that is predicated on a set of system and software engineering capabilities that should be presented as organizations reach different levels of process capability and maturity.” [15]
* “The role of estimating within the software process often serves as a ‘sanity check’ on the predefined deadlines and budgets that have been established by management.” [16]
* “For our purposes, software quality is defined as conformance to explicitly stated functional and performance requirements, explicitly document development standards, and implicit characteristics that are expected of all professionally developed software.” [17]
* “Computing technology and program architectures have undergone a sea of change, but the characteristics that define high-quality software appear to be invariant.” [17]
* “Unfortunately, testing, even when performed well, is not as effective as we might like for all classes of errors. A much better strategy is to find and correct errors (using FTRs) before getting to testing.” [18]
* “And change increases the level of confusion among software engineers who are worked on a project. Confusion arises when changes are not analyzed before they are made, recorded before they are implemented, reported to those who should be aware that they have occurred, or controlled in a manner that will improve quality and reduce error. [18]
* “All engineering disciplines encompass four major activities: (1) the definition of the problem to be solved, (2) the design of a solution that will meet the customer’s needs, (3) the construction of solution, and (4) the testing of the implemented solution to uncover latent errors and provide an indication that customer requirements have been achieved.” [19]
* “A problem cannot be fully defined and bounded until it is communicated.” [19]
* “Other practitioners believe that it is worthwhile to use a number of different modes of representation to depict the analysis model.” [20]
* “Like analysis modeling, software design has spawned a collection of methods that populate the conventional, object-oriented, and formal regions that were discussed earlier.” [21]
* “The problems that we continue to encounter in the creation of high-quality, computer-based systems have relatively little to do with the means of construction.” [22]
* “Over the past three decades, a rich variety of test case design methods have evolved for software. These methods provide the developer with a systematic approach to testing.” [23]
* “Requirements engineering (and software engineering in general) sometimes reinvents the wheel.” [24]
* “To meet this challenge, software must be constructed from reusable components.” [25]
* “The degree to which the industry embraces software engineering ad works to instantiate it into the culture of software development will have a strong bearing on the final answers to these questions.” [26]

**Things I Didn't Agree With**

“Fortune (we call it risk) is in the back of every software project manager’s mind, and that is often where it stays. And as a result, risk is never adequately addressed. When bad things happen, the manager and the project team are unprepared.” [16]

I don’t agree with this statement because I think any project manager who does not think of risk is not fulfilling his role as project manager. An example is when people use the spiral model risk is at the foremost in their mind.

**Things I Did Not Understand**

I understood the whole article.