Answers to Guide 12

* CMMI: (capability maturity model integration)
  + Who uses CMMI most frequently?
    - Department of Defense and U.S. Government contracts relating to software development.
  + Compare and contrast the five maturity levels.
    - Level 1 – Initial: processes unpredictable, poorly controlled, and reactive
    - Level 2 – Managed: processes characterized for projects and is often reactive.
    - Level 3 – Defined: processes characterized for the organization and is proactive. (project tailor their processes from organization’s standards)
    - Level 4 – Quantitatively Managed: processes measured and controlled.
    - Level 5 – Optimizing: focus on process improvement.
* Process area (CMMI): (a cluster of relative practices in an area, that, when implemented collectively, satisfies a set of goals considered important for making improvement in that area.)
  + Compare and contrast the following process areas (skimming the others):
    - Configuration Management (CM):
      * Support process area at maturity level 2
      * Purpose is to establish and maintain the integrity of work products using configuration identification, configuration control, configuration status accounting, and configuration audits.
    - Project Planning (PP):
      * Process area at maturity level 2
      * Purpose is to establish and maintain plans that define project activities.
    - Requirements Development (RD):
      * Engineering process area at maturity level 3
      * Purpose is to elicit, analyze, and establish customer, product, and product component requirements.
    - Risk Management(RSKM):
      * Process area at maturity level 3
      * Purpose is to identify potential problems before they occur so that risk handling activities can be planned and invoked as needed across the life of the product or project to mitigate adverse impacts on achieving objectives.
    - Technical Solution (TS):
      * Engineering process area at maturity level 3
      * Purpose is to select, design, and implement solutions to requirements.
      * Encompasses products, product components, and product related lifecycle processes either singly or in combination as appropriate.
* Software Metrics: (a standard of measure of a degree to which a software system or process possesses some property.)
  + Which of the metrics discussed in Section 1 are used to measure the following things? Pay particular attention to “size” metrics.
    - Software product: Mean time to failure, defect density, customer problems, customer satisfaction.
    - Software process: Cost of quality, defect density, review efficiency, testing efficiency, defect removal efficiency, residual defect density
    - Software quality: reliability, performance – load testing, stress testing, soak testing, security – # of vulnerabilities, maintainability and code quality – lines of code, static code analysis, rate of delivery - # of software releases
  + W.E. Deming said that “the most important things cannot be measured.” DeMarco articulated a similar idea (see his quote above). Does this apply to software measurement?
    - Most things that really matter – honor, dignity, discipline, personality, grace under pressure, values, ethics, resourcefulness, loyalty, humor, kindness – aren’t measurable.
    - From a Christian perspective, it could. Does this program make the world a better place?
  + Section 3 states that metrics can sometimes do more harm than good. Do you agree with this? If so, give an example; if not, explain why not.
    - Yes, a simple example would be students cheating to get an A because the academic system rates how well a student does purely on the basis of those letters and numbers on the transcript.
* Project Management Tools:
  + Martin Fowler’s Continuous Integration
    - Why is software integration so hard?
      * People don’t believe continuous integration will work with their project or that it won’t make any difference, so they don’t try to utilize it.
      * Requires daily, multiple commits, and automated integration testing on those commits as well as merge conflict resolution between different builds by different developers before merging into master.
    - What is continuous integration?
      * Software development practice where members of a team integrate their work frequently, with each integration verified by an automated build (tests included) to detected integration errors as quickly as possible.