

These questions relate to the Toward a Discipline of Software Engineering article (by Anthony I. Wasserman) that was handed out in class. *Please* hand up to a maximum of three pages of typewritten answers according to the format outlined in the syllabus.

1. [20pts] According to Wasserman, would SE principles practiced in the USA be the same as those practiced in Japan? The most important aspect of your answer is the why or why not?

*Universality (pg25): Software engineering has no equivalent to other "established" engineering discipline's set of unambiguous representations for modeling artifacts. Instead there are hundreds of different notations. Many notations have numerous variations. Therefore, based on Wasserman's comments I would predict that the principles practiced here would not necessarily be the same as those used in Japan. In fact I'd think that they were widely divergent. In fact, it seems obvious that the principles practiced from company to company would vary significantly.*

2. [20pts] Define Wassermann's notion of *Modularity and Architecture* and discuss why its important (mention design patterns and how that relates to standardization trends).

*A well-structured system architecture follows basic design principles of software design. It should exhibit a logical and modular structure and support information hiding. Its important because of the role it plays in determining system quality and maintainability. The idea of modularity is rooted in partitioning in some reasonable fashion (modular, data-oriented, event-oriented decomposition, and outside-in or object-oriented design). Design patterns combine the attributes of class libraries and development frameworks in trying to provide reusable classes and frameworks for specific application domains. Standardization of software architecture is coming of age and seeks to establish standard designs and design patterns that can provide an efficient way to incorporate reuse into software design (not reuse of code but reuse of design and important distinction)*

3. [20pts] What are the five partitioning approaches Wasserman mentions in his section on *Modularity and Architecture*? Do you agree or disagree with his stance on these? Why?

*(1) Modular decomposition (assigning functions to modules), (2) Data-oriented decomposition (based on external data structures), (3) Event-oriented decomposition (events the system must handle), (4) Outside-in design (based on user inputs to the system), and (5) Object-oriented design. I do agree that these approaches often complement each other because knowing more than one can often give insights into what, when, where and why (or why not) use one and/or the other (i.e., it is reasonable to combine them in some contexts). If you only have a hammer in your toolbox, I'll bet that most of your problems turn into nails!*

4. [20pts] What are the roles of Reuse and Metrics in the *Life cycle* and *Process* according to Wasserman? Name and define at least two product metrics and two process metrics.

*A broad notion of reuse goes well beyond APIs and class libraries to encompass many types of artifacts associated with a software development project. Software development organizations must also look at domains specific to their own needs and practices. One found, these items should be treated as corporate assets; the investment that went into creating them must be recognized. Such items include not only source and object code, but also design patterns, document templates, test scripts user interface layouts, and software process definitions.*

---

*Its impossible to measure progress toward process and product improvement without a well-defined set of items to be measured and accurate measurements of current practice. There is broad agreement on the need for organizations to measure key aspects of their software development activities, as well as product quality, including usability. Different types of metrics are associated with each software development task (note that metrics are well established in testing and cost estimation).*

*Product metrics include: Defect density (number of defects / LOC discovered up to now); Defect severity categorizes the kinds of defects in terms relative to the user/customer's perception of quality. Process metrics include: Cost underestimation (in dollars) and schedule slippage (in days to months), employee turnover, and a plethora of others that correlate with the quality of the output from the process, namely the product. )*

5. [20pts] What are the five main issues surrounding Software Engineering Environments? Choose the most fundamental issue and discuss why you think its important (i.e., more important than the other four)?

*Platform integration (the ability of tools to interoperate on a [potentially heterogeneous] network; Presentation integration (commonality of the user interface among tools); Process integration (linkage between tool usage and the software development process); Data integration (the ways in which tools share data); Control integration (the ability for tools to notify and initiate actions among one another.*

*Probably the most important two issues are data and control integration. This is evidenced by the commercial ventures that have focused on those aspects. Yet one may argue that platform integration is also very important because of the investment in such. However, by solving the first two there would be a considerable easing on the problem of heterogeneous environments. The big win may eventually come from the development of a open environment that supercedes the platform problem.*