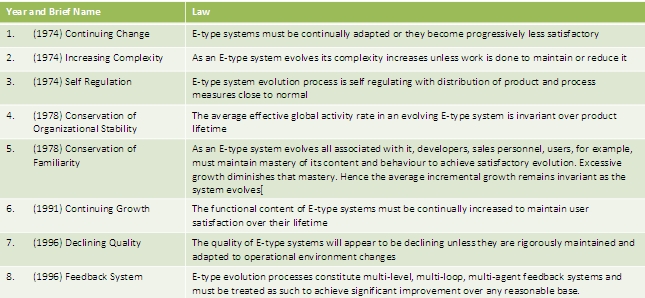
Written Assignment 7

Student

University of the People

**1. Under which circumstances might the Lehman's laws break down?**

According to Sommerville (2011), Lehman’s laws encompass characteristics of software evolution and hold true for large organizational software systems, or E-type systems, where new releases of a system are needed as business needs change. Lehman’s Laws are:

****

(<http://1.bp.blogspot.com/-GPTPCnZrVII/Ti78uP5VFtI/AAAAAAAAAkY/aA4cCZOZmDQ/s1600/cats3.jpg>)

Although Lehman’s Laws hold true under most circumstances for large systems developed by large organization, they do sometimes break down. According to Sommerville (2011), the laws do not hold true and need to be modified for the following circumstances:

* + Shrink-wrapped software products;
  + Systems that incorporate a significant number of COTS components;
  + Small organizations;
  + Medium Sized systems.

**2. Under which circumstances might an organization decide to scrap a system when the system assessment suggests that it is of high quality and of high business value?**

Earlier in this course we learned that many companies hold on to legacy systems, either because they still hold high business value or because they are simply high quality products. In fact, according to Sommerville (2011), many companies hold onto these systems long after they should because of the initial investment and the cost of changing out these systems, but they may have a limited budget for maintaining and upgrading these systems. In order for a company to decide to scrap the system completely, the business processes should have changed enough from the time the system was installed to warrant no longer using the system. In other words, if the company no longer relies on the legacy system for business processes, it is more likely that the system will be scrapped completely. If the system still performs business processes, then it is more likely that the company will attempt to either maintain the system, re-engineer it, or replace parts of the system (Sommervill, 2011).

**3. It seems odd that cost and schedule estimates are developed during software project planning-before detailed software requirements analysis or design have been conducted. Why do you think this is done and under which circumstances do you think this should not be done?**

The majority of the software budget in large companies is devoted to changing and evolving existing software rather than developing new software. Said in a different way, “the costs of software maintenance usually exceed the software development costs” (Sommerville, 2011). It is ironic that maintenance schedules and the costs involved are usually estimated during the earliest stages of software development. In fact, the cost and schedule are often estimated during project planning even before specific software needs have been identified and before the design has been determined. This is likely because the customer needs to know the cost of not only the development of the software product, but also of maintaining that product long term. The written agreement between the customer and developer may stipulate the cost and schedule for maintenance as well as development. In environments that are slow to change, this may not be very problematic. But in environments which change rapidly, the estimates for cost and schedule for maintenance will vary wildly. Therefore nailing down cost and schedule estimates early in the development stages should not be done when the environment is very volatile (Sommerville, 2011).

**4. Do you think performance can be interpreted differently depending upon the software application area? Justify your answer.**

Performance is a factor that is used to assess software systems. According to Sommerville (2011), assessing the performance of a system involves answering the questions “Is the performance of the system adequate?” and “Do performance problems have a significant effect of system users”? With such vague questions, it is obvious that system performance can be interpreted differently depending on which application area a software is used. Sommerville (2011) states, “To assess a software system [...], you need to consider both the application system itself and the environment in which the system operates”. The environment is important in assessing a system for a number of reasons. For one thing, when you consider the environment, it can encompass all of the related hardware and any software in support of the system. So any changes in the environment, like upgrades to hardware or operating system, could warrant changes in the system. For this reason, performance needs continual assessment as the environment changes.

**References**

“Lehman’s Laws” (n.d.). Table. Retrieved from <http://1.bp.blogspot.com/-GPTPCnZrVII/Ti78uP5VFtI/AAAAAAAAAkY/aA4cCZOZmDQ/s1600/cats3.jpg>

Sommerville, I. (2011). Software Engineering. Edition 9.