

## **SWEBOK Review Comments** **Trial Version 1.00 (downloaded 6/4/03)**

Douglas Hoffman BACS, MSEE, MBA, ASQ Fellow, ASQ-CSQE<sup>1</sup>  
[Doug.hoffman@ieee.org](mailto:Doug.hoffman@ieee.org)      [www.SoftwareQualityMethods.com](http://www.SoftwareQualityMethods.com)

### **Summary:**

Outlining and describing the current state of the practice is an excellent undertaking, but at the same time it can be misleading and dangerous. My broad experience with computer system manufacturers, software companies, established organizations, startups, embedded systems, operating systems, middleware, applications, tools, regulated and unregulated industries, and more has given me a unique perspective on what constitutes good, useful, helpful practices for a huge range of circumstances. I've found that nearly all documented techniques are extremely valuable for some context, but those same techniques are counterproductive in other contexts. I believe that SWEBOK must identify the good practices while allowing for contexts for which they are inappropriate. Indicating that they are "best practices" (and thus always applicable) is harmful to the profession.

My biggest concern with a body of knowledge (BOK) like SWEBOK is that it tends to exclude many valuable emerging and niche elements and at the same time indicating that some specific techniques should be universally practiced. Worse, published BOKs are used for certification of professionals and become the basis of professional malpractice when techniques described in them are not blindly followed. A BOK is only valid for certification when the contents are universally applicable or when the applicable contexts are clearly understood and articulated. A technique that is applicable and useful for software used in controlling an airplane may not be applicable or useful for software used in a DVD player.

Unfortunately, I do not see enough context specific qualifiers with practices in the SWEBOK. (Indeed, I'm not sure we computer scientists understand enough about computers and software to describe relevant context characteristics.) Given a choice, I would vote "No" on accepting the SWEBOK, not because it is necessarily bad or wrong, but because it will mislead and possibly confound progress in computer science. It's the first step toward legislating the one way all software will be "engineered."

I began reviewing the SWEBOK intending to take a cursory look at the sections I am most proficient with and providing suggestions and possibly a few critiques. I spent several hours going into the first two chapters before skipping to the chapters on Testing and then Software Quality. I was encouraged by the explicit recognition that different organizations, users, and products require different techniques in both chapters. But, I was discouraged by the many deficiencies in the Testing chapter and the gaping holes in the Software Quality chapter.

I am shocked by the fact that no reference is made to ASQ's CSQE, if even to criticize it. Are the drafters of the SQEBOK really ignorant of the existence of a sister society's related Software Quality Engineering Body of Knowledge, or have they chosen to ignore it because it might be inconvenient or at odds with SWEBOK?

---

<sup>1</sup> Douglas has over 30 years experience as a practitioner and instructor in computer science since earning his BA in Computer Science. He has extensive experience in quality assurance, testing, test automation, requirements, development processes, and project management in particular.