SWEBOK

Trent Forkert

0. Table of Contents

1. Software Requirements

This Knowledge Area is concerned with defining the constraints a piece software of software will have, generally involving practical problems, rather than theoretical ones. The Software Requirements KA describes the process, elicitation, analysis, specification, validation, and considerations to be made regarding the software requirements.

In addition, this KA defines different types of requirements, such as functional and nonfunctional requirements, emergent properties, quantifiable requirements, and system requirements. Furthermore, there are built-in standards for reassessing and updating the processes the knowledge area defines.

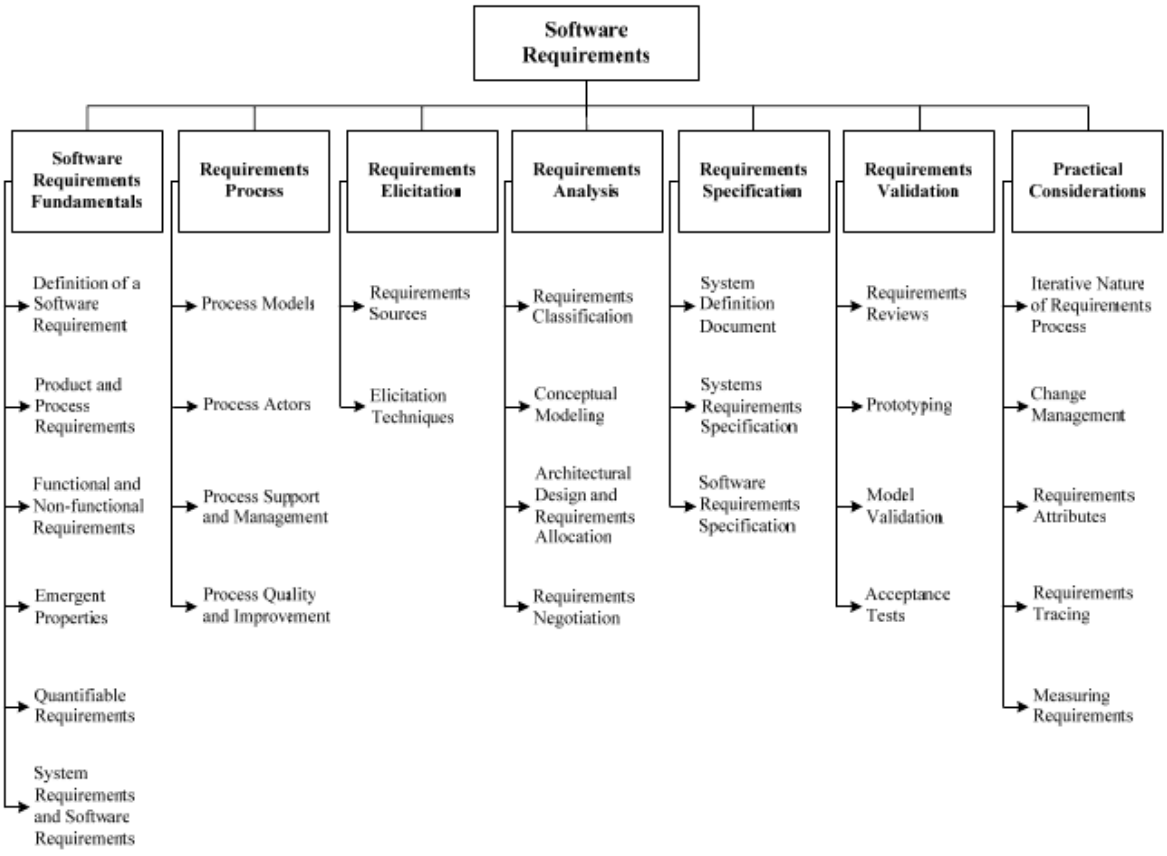

Figure 1 Breakdown of topics for the Software Requirements KA

2. Software Design

This Knowledge Area is concerned with the design of software. That is, breaking a software system in to appropriate pieces to better facilitate the development of the software system. To that end, the Software Design Knowledge Area describes different design patterns, quality analysis, notations and strategies for designing software.

This knowledge area defines ideas such as abstraction, coupling and cohesion, decompostion and modularization, and encapsulation. This knowledge area also describes different notations, such as Entity-relationship diagrams and Component diagrams. It goes on to describe design strategies, such as structured and object-oriented design.

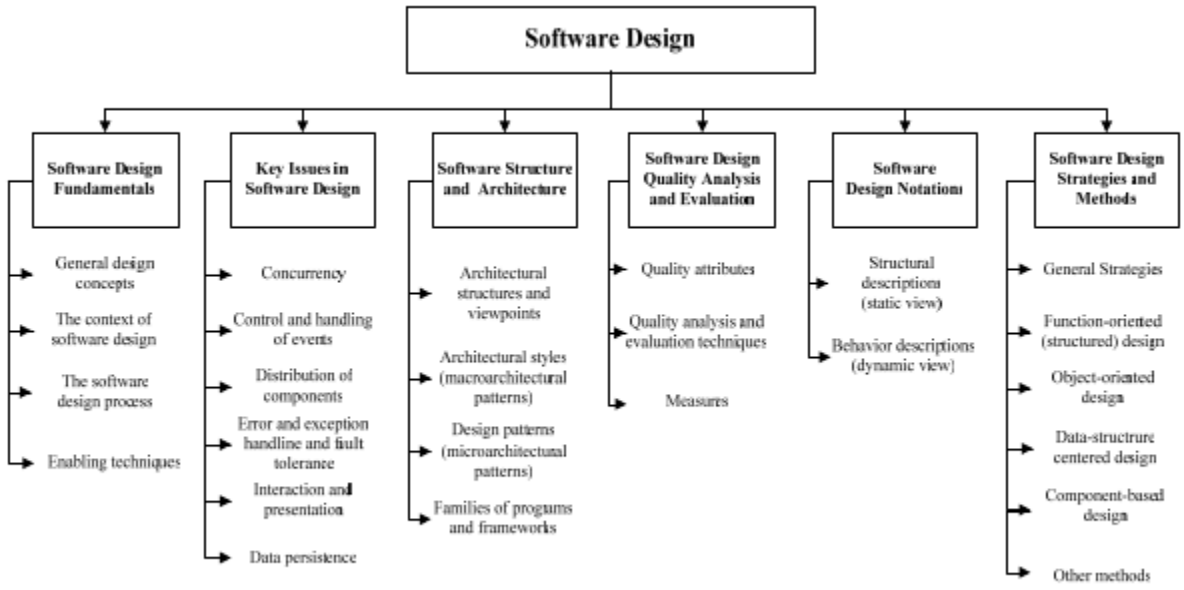

Figure 1 Breakdown of topics for the Software Design KA

3. Software Construction

This Knowledge Area describes the actual implementation of the software system, in accordance to the requirements and design. However, this is more than just programming, but includes some debugging and testing.

Large parts of this knowledge area are dedicated to constructing software in such a way that the task naturally makes itself easier and more fault-resistent. This is where topics such as standards conformance, unit tests are of incredible use and importance.

This KA also describes what I consider good basic programming skills, namely minimizing complexity and code reuse. To me, these parts of the knowledge area seem reminiscent of the UNIX Philosophy, where you make one piece of code do one simple thing well, and build up complexity by combining smaller pieces of code.

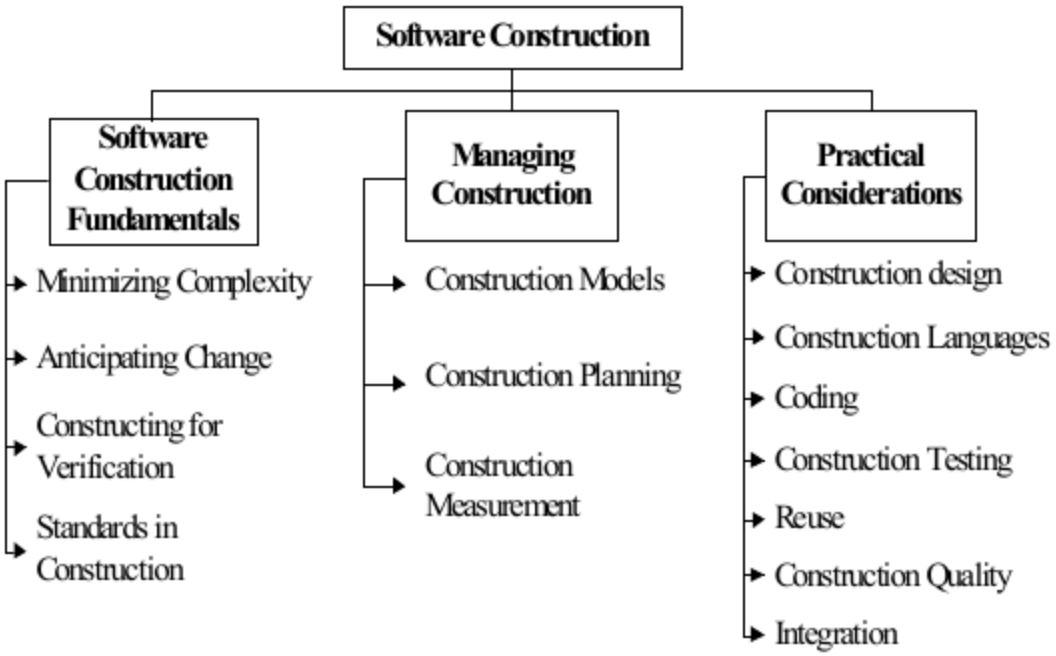

Figure 1. Breakdown of topics for the Software Construction KA.

4. Software Testing

This knowledge area describes testing, as well as techniques and processes for testing software systems. The Software Testing Knowledge Area is stresses the fact that tests must be carefully selected, so that a relatively small number of tests have a significant impact and meaningful results. Testing also explicitly happens by actually executing code.

This KA describes different goals of testing, such as requirements testing and correctness testing, where the software is tested to see if it meets the customers requirements and behaves correctly, respectively. Also described is stress testing, the process of seeing how the software performs under very high load. This will have direct application to our project.

Additionally, a large number of techniques for testing are described, including fault-based techniques and usage-based techniques.

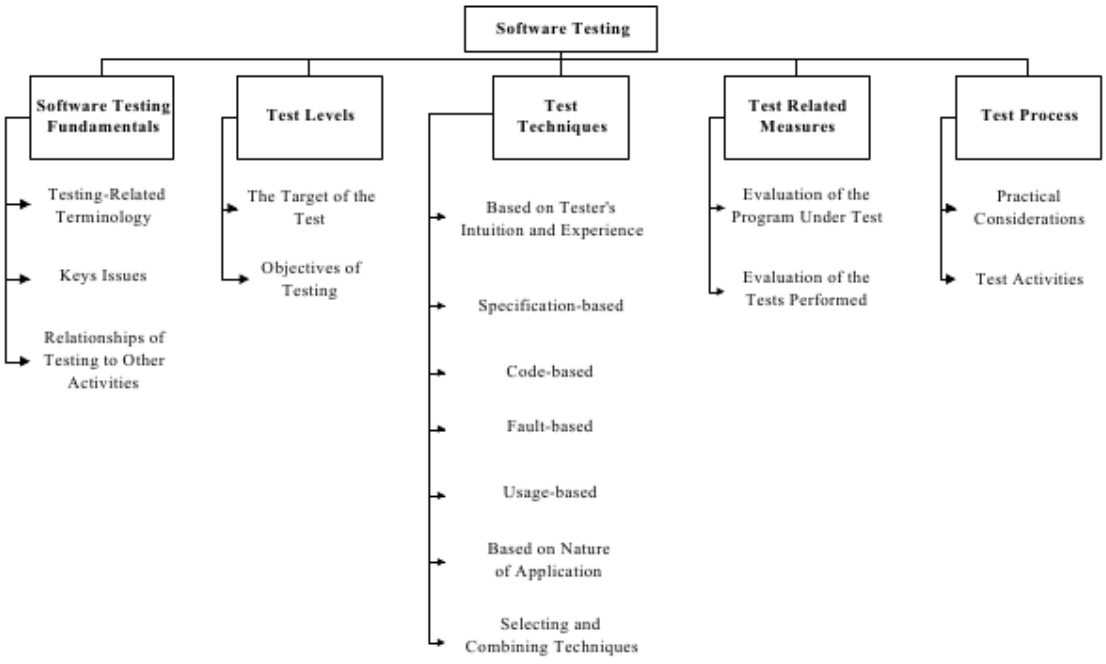

Figure 1 Breakdown of topics for the Software Testing KA

5. Software Maintenance

This knowledge area describes software maintenance, its processes and techniques. This includes planning for maintenance prior to release, as well as fixing problems and providing support after release. Maintenance includes reevaluating the software to make sure it still meets requirements, or indeed whether or not those requirements need updating later in the software's life cycle.

Part of this knowledge area is concerned with the categorization of different types of maintenance, with four main categories (preventive, perfective, corrective, and adaptive). Also discussed are the cost implications of continued maintenance, and the cycle of the software maintenance process.

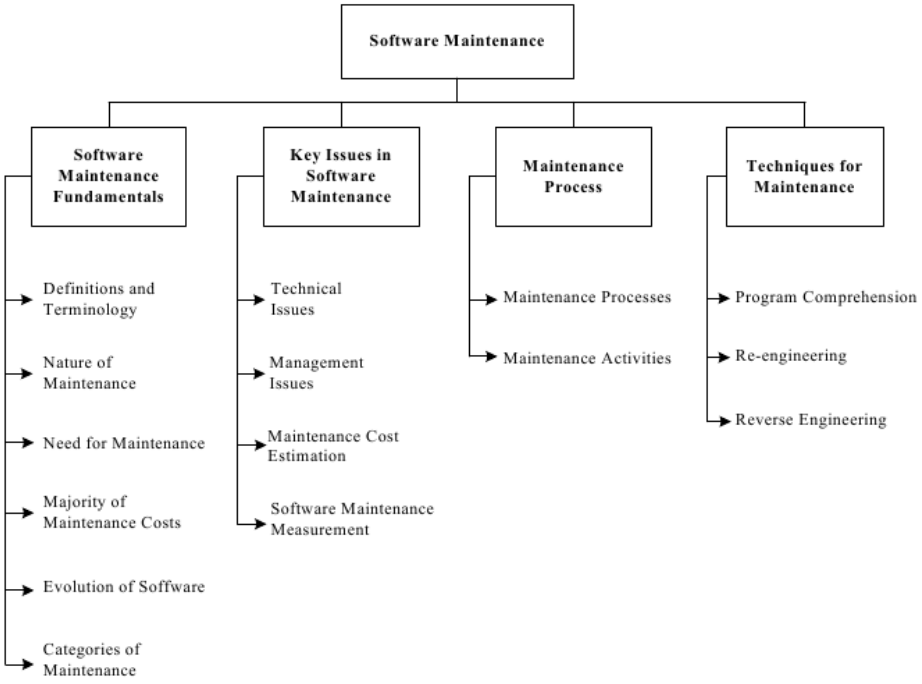

Figure 1 Breakdown of topics for the Software Maintenance KA

6. Software Configuration Management

This Knowledge Area describes standards for the management of the various properties of the different elements of the system.

This includes the process by which changes to the software are requested, evaluated, approved and implemented. The Software Configuration Management Knowledge Area also describes the tracking of software status, and the processes by which software is built and released, including documentation and release notes.

To me, the most interesting thing out of this knowledge area is the explicitly labeled difference between versions, revisions, and variants. It makes sense why that would be standardized, but in my experience, these terms are often used interchangeably, which I don't consider to be a good thing.

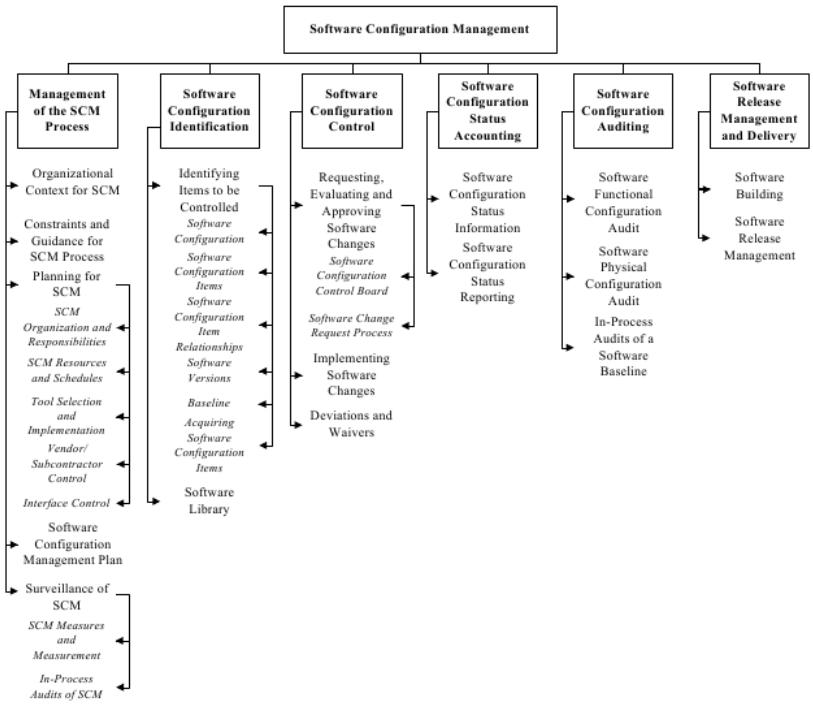

Figure 2 Breakdown of topics for the Software Configuration Management KA

7. Software Engineering Management

This Knowledge Area defines items concerning Software Engineering Management, considering any special characteristics software engineering projects have compared to general engineering projects, namely the rapidly changing nature of technology - and therefore the rapidly changing nature of requirements - coupled with the often underappreciated complexity of software engineering.

The Software Engineering Management Knowledge Area describes concepts in software project planning, enactment, and evaluation. Together, much of this sounds very similar to what we have done in following RUP for class.

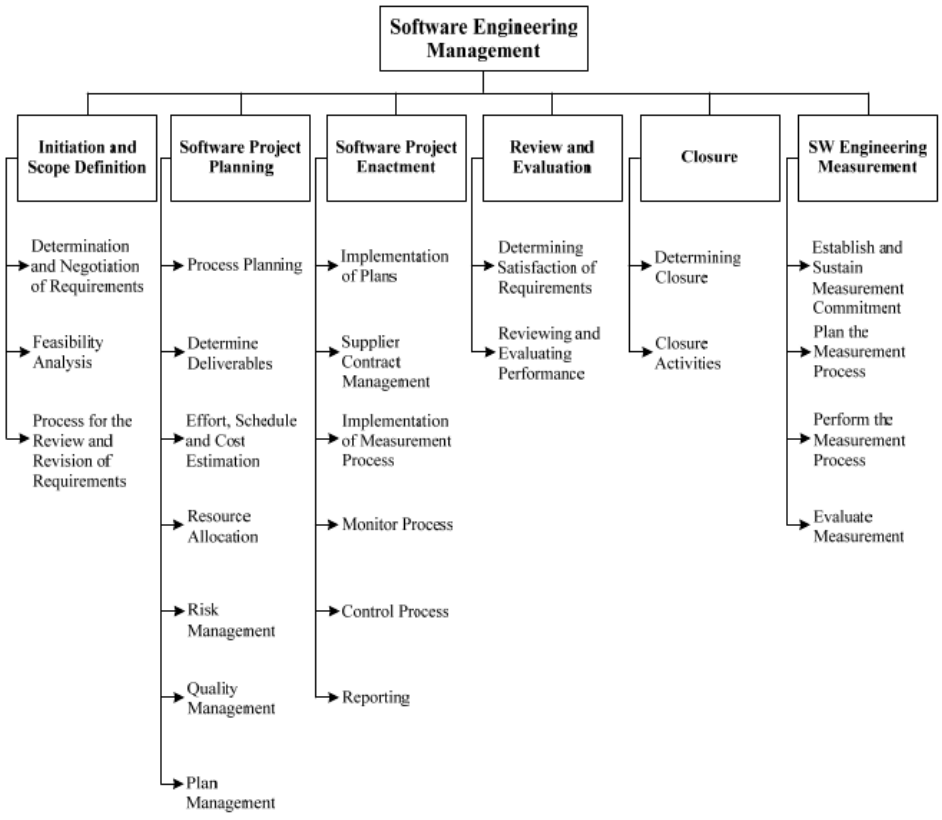

Figure 1 Breakdown of topics for the Software Engineering Management KA

8. Software Engineering Process

This Knowledge Area defines “software engineering process” in three different ways, though only focuses on one.

- The process used to engineer software
- processes related to software engineering
- set of activities performed within an organization

The Software Engineering Process Knowledge area defines ideas relating to software life cycles, and process assessment, among others.

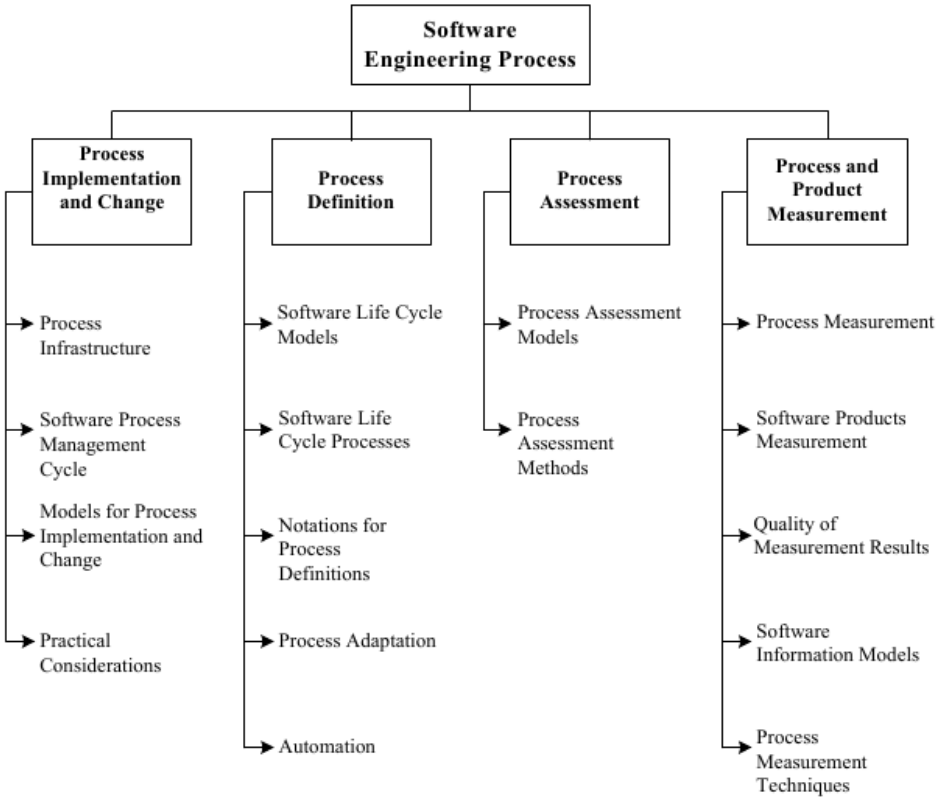

Figure 1 Breakdown of topics for the Software Engineering Process KA

9. Software Engineering Tools and Methods

This Knowledge Area defines terms related to the tools and methods of software engineering. The tools are usually computer-based, and provide some automation of the software engineer's task. These tools aid in reducing human error by removing the burden of repetitive tasks.

The Software Engineering Tools and Methods Knowledge Area defines several categories for tools: Requirements, Design, Construction, Testing, Maintenance, Configuration Management, Engineering Management, Engineering Process, and Quality. These reflect the other Knowledge Areas that compose SWEBOK.

This Knowledge Area also defines different methods the tools and engineers can use to go about their task.

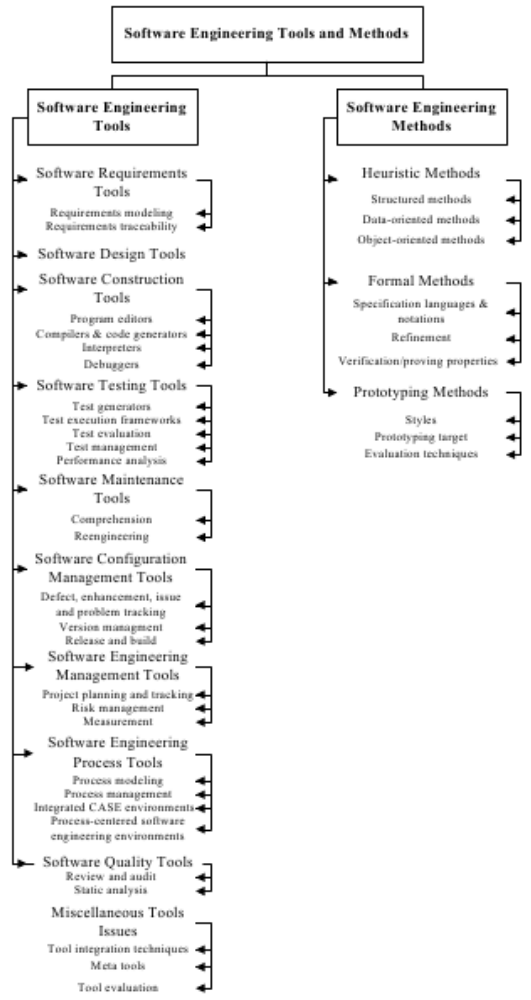

Figure 1 Breakdown of topics in the Software Engineering Tools and Methods KA

10. Software Quality

The Sofware Quality Knowledge Area describes ways of achieving software quality, through both static and dynamic techniques.

This knowledge area defines concepts concerning quality assurance, verification and validation, reviews and audits, as well as practical measures that can be taken to ensure a software's quality meets the client's expectations.

Like many other KAs, this KA defines a Management process by which software engineers can systematically achieve high software quality.

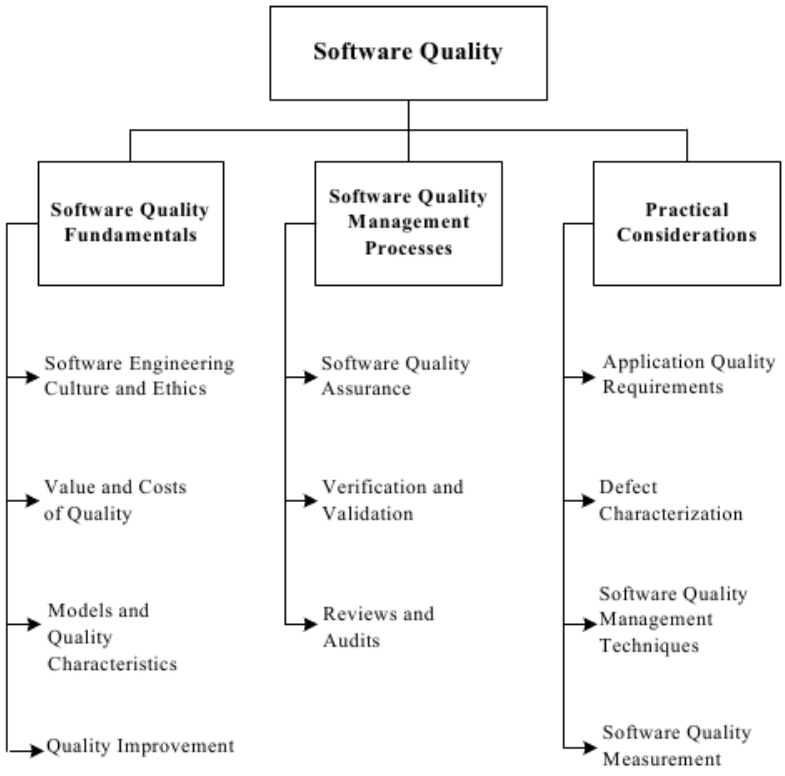

Figure 1 Breakdown of topics for the Software Quality KA

11. Knowledge Areas of Related Disciplines

This knowledge area provides information about a variety of disciplines closely related to software engineering. These include: Computer Engineering, Computer Science, Management, Mathematics, Project Management, Quality Management, Software Ergonomics, and Systems Engineering.

Each of these disciplines is described in brief, and summarize the Body of Knowledge for each field, if it has one. This is accomplished through a simple bullet point list, with refences to external documents mentioned where appropriate.

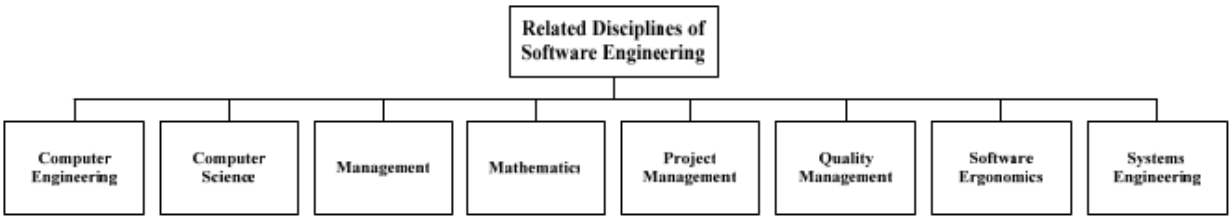

Figure 1 Related Disciplines of Software Engineering