SCM: More Than Support and Control

Software Configuration Management (SCM) supports people, controls data, and provides data integrity to baselines. Advances in automated tools and how baselines are created, maintained, and delivered requires careful, detailed planning. To be effective, SCM must be involved in day-to-day evolving, developmental, and ongoing maintenance activities.

Depending on where you go and whom you ask, you will always get different answers to the question, "Is SCM a *support* or *control* organization?" Even if you ask two people working on the same program, you likely get different answers. Those who are afraid it is a trick question, or who want to appear profound, will respond, "Why, it is both, of course." But when asked what they mean, they usually have other matters to attend to.

The answer is that SCM is a support *and* control organization, and when it is handled properly, a third factor is drawn into the picture, that of being a *service*.

If you talk with software developers, they likely will tell you they want support and some measure of control, but not too much. Management will generally say controls are a more important issue, as long as they do not impact costs and schedules. A SCM person with 10 or more years of experience, probably a first-line supervisor or manager, will agree with management, but will insist that sometimes cost and schedule must be impacted. A SCM person with fewer years under his or her belt will generally side with the software developer and strive to do whatever is necessary to assist him and get the job done.

The question becomes, "How can SCM best accomplish both support and control issues and be of service while adding value to the program effort?"

Support: SCM is a support organization in that it supports program engineers and developers, the program, the corporation, and in many situations, the customer.

Control: SCM is a control organization in that it controls specifications, documents, drawings, requirements, tools, software, and other deliverables.

Service: SCM is a service provider in that it *supports people* and *controls data*. This simple sentence is the primary key to a successful SCM operation.

The staff must be able to wear two different hats: one to support people, and one to control data. When those two hats get mixed up, i.e., SCM trying to control people and what/how they do things, problems and bottlenecks appear on the horizon. When this happens, often SCM is bypassed for the sake of "get the job out the door and we will fix it later."

It becomes the SCM manager's responsibility to:

- Ensure SCM personnel are properly trained and have necessary resources (budget and tools) to do an efficient and effective job.
- Ensure that a proper balance of control and support is tailored to each program that is supported.
- Ensure the SCM function is flexible and can accommodate changing needs and requirements of developers, customers, the program, and the company.

Future Directions

The SCM task has not changed much during the last 20 to 30 years. However, the environment within which SCM operates has significantly changed; that is likely to continue. Software language bases have changed: from Basic, COBOL and FORTRAN, to Ada and Pascal, to C++, Java, and many others. But that has not been the real impact to SCM; after all, code by any other name is still code.

More significant impacts to SCM have centered on automated tools and the library systems upon which they operate.

The tools have progressed from version control and semiautomatic build operations to systems that can establish and monitor the entire software development and production environment. Tools are more sophisticated and suppliers more numerous. Not long ago it was a somewhat simple matter for SCM to determine the best tool for the job. But in today's market, new issues have to be addressed before a decision is made. It is becoming increasingly important for representatives from each department within engineering organizations to consider, evaluate, and weigh their requirements against capabilities of various available tools.

SCM automated tools available today, and those on drawing boards, are much more versatile than their forerunners. But when asked, "Is there not one tool that will do it all?," the answer is still *no.* It is largely due to the fact that the SCM operating environment is still evolving.

In the recent past, SCM dealt with code and a few documents tucked away into a baseline where they could be easily controlled. With the introduction of Web-based repositories and increased involvement with commercial off-the-shelf vendors and subcontractor applications, baselines, as originally defined, are quickly becoming things of the past by becoming conceptual in nature. After all, when was the last time you *saw* a functional, allocated, or product baseline? Probably when you were dealing with hard copy documents and printed program lists. In the electronic office, these are nowhere to be found. The tendency is becoming to refer to the current version of controlled entities (code, documents, requirements, etc.) as the baselined version; previously, controlled versions were starting to be referred to as archived baseline versions.

In the past, most programs operated with three baselines: functional, allocated, and product (or requirements, design, and product, or some other set of descriptive labels). NASA once tried a system with nine baselines; it had a short life span. NASA, and a majority of other developers, was operating on the assumption that it was important to know into which baselines controlled items were placed. The primary concern was that the item was baselined, in that it could not be changed without a formal process, and without giving a name to the electronic partition where the item resides.

What drives this move away from the three former baselines? It is SCM and other developers raising questions such as, "Where do I put this .JPG file that is used in different documents?," "Should build scripts and make files also be controlled?," and "Where do we control corporate assets known as or to be captured as reusability or re-engineering issues?"

The questions are becoming, "Does the control board control the document or the information it conveys?" and "Does the control board control the software code or what it does? How does it do it, and and what is used to create it?" In the past, SCM controlled code and at times, documentation. What can be baselined in the environments with which we are now beginning to deal? The easier question is "What cannot be baselined?" Answer: SCM can baseline anything the program needs to control/make available. Answer: SCM controls data in any form so it can support people and provide an integral service to the program.

Lessons Learned

With only a few exceptions, if you look at any of the SCM standards, manuals, guides, books, etc., you will likely find that SCM has four major functions:

1. Identification
2. Change Control
3. Status Accounting
4. Audits and Verifications.

In nearly every case, *planning* is left out. And yet, SCM is using much more complex equipment to establish and maintain complex environments, multiple baselines, multiple environments on multiple platforms, etc. Like everyone else, SCM has to do this faster, cheaper, smarter, and better than before. Planning has become more important than ever. As recently as 10 years ago, there was still some truth to the statement, "Have them do SCM. Someone has to do it and anyone can learn it quickly enough." That is no longer true. The job has become too technical, too complex, and dependent on too many different variables to make it an easy job that anyone can pick up.

It is true that SCM still relies fairly heavily upon on-the-job training. No universities or colleges offer a four-year program in SCM. However, academia has recognized the evolving complexity of the job. Some two-year community colleges offer SCM certification programs. In 1998 I worked with three people who were pursuing doctorate degrees and centering their respective theses on SCM and its functions.

If this is all true, then *planning* cannot be interpreted as "A SCM plan has been written." That is a good start, but much more than a document explaining SCM's roles and responsibilities is needed. SCM planning activities must also include:

Metrics—how long, how many, when and where?

Skill Mix—what is needed and who has it or who can get it?

Infrastructure—who is doing what, where, when, how?

Contingencies—if this happens, then what?

Effort Tracking—manpower levels, roll on, and roll off.

Subcontracts—responsibility and authority.

Resources—budget, tool licenses, training, head count.

Matrix Management—decentralized workforce.

Control Transitions—informal to formal to field.

Records Retention—what gets kept, where, and for how long?

Control—who controls what and how do they do it?

Process—standardized procedures for repeatability.

Case Study 1

Last year, at a large aerospace corporation in the southeast, the SCM manager recommended the purchase of an automated SCM system that would satisfy all requirements laid out by SCM groups. Management placed the recommendation on hold to give other engineering departments time to review the tool. In the end, the recommendation to purchase it was cancelled. While the tool supported the SCM organization, it did not adequately address other developmental considerations the engineering ranks thought were important. Sometime later, a different tool was purchased that satisfied all the major requirements of SCM, software developers, software quality assurance, test, integration, and management organizations.

Case Study 2

During a recent visit to a private sector corporation (one that did not deal with government contracts) in New England, it was discovered that the developers' major concern about implementing SCM activities was so-called restrictions that would have to be addressed. They had been led to believe that SCM meant formal controls, restricted access, limited ability to apply creative solutions, and so on. When it was suggested that data can transition to formally controlled baselines through a series of informal control steps, and that SCM did not mean a lockdown or bottleneck, they became eager to be involved. After a number of meetings, a phased approach to formal SCM allowed for the placement of informal controls and data gathering, which led to baselined items. Everyone was pleased with the process. The developers soon realized they could work as a team with SCM to solve problems rather than as two separate organizations with their own concerns and desired solutions. More importantly, the SCM group learned that when it got out of its corner office and on the engineering floor (being support- and service-oriented) it quickly became an integral part of the engineering and development process and team.

Conclusion

In both of the above cases, SCM and developers soon realized they could work as a team to solve problems rather than as separate organizations with their own concerns and desired solutions. World class SCM operations can only be realized after properly planning for the implementation of SCM's four major functions. While SCM automated tools and baseline applications have become more and more complex, the fundamentals of SCM have taken on a new slant. SCM supports people, controls data, and provides a baseline integrity service. If that is not true where you work, it is time to ask why not.

About the Author

Bruce Angstadt is a Software Process Improvement Manager at Xerox Corp. in Webster, N.Y. With 35 years of government and industry experience in engineering support disciplines, his specific areas of interest are technical training and software configuration management. Prior to Xerox, he was employed by the Navy, Bell Laboratories, and Harris Corp. He also teaches a number of courses in configuration management for Systems Technology Institute of Malibu, Calif. He has a bachelor's degree in psychology and business management from Rollins College in Orlando, Fla.

Xerox Corp.
800 Phillips Road
Webster, N.Y. 14580
Voice: (716) 265-8700
Fax: (716) 422-0780
Email: Bruce.Angstadt@usa.xerox.com
Internet: http://www.scmtoday.com