How Do I Make My Organization Comply With Yet Another New Model?

Sarah A. Sheard
Software Productivity Consortium

Comments often go something like this when organizations have to change their current process improvement model: "Help! We have to revise all our ISO 9000 processes to meet the Capability Maturity Model® (CMM®)!" Or, "I don't want to write systems engineering processes to EIA/IS 731 because I'm only going to have to rewrite them to CMM Integration℠ (CMMI℠)." If these sound familiar, it is because organizations are currently faced with the prospect of shifting their process improvement focus from a current model such as EIA/IS 731, ISO 9000, or the Software CMM to a new model such as the CMMI. To some, shifting seems a daunting feat; in many cases, it has taken years to adapt to the current model. Starting over seems to negate the value of the current investment. This article shows that capability models (and other process standards such as ISO 9000) generally ask the organization to do similar or identical things; adopting a new model should be approached as mapping and modifying rather than as starting over.

Many companies have begun documenting processes in response to ISO 9000 [1], one of the various capability maturity models, or perhaps to other standards [2, 3, 4]. These standards and models require documentation of, and disciplined adherence to, organizational processes. One concern is that these models are themselves changing. For example, an updated version of the ISO 9000 series of standards was released in 2000. Also, the Software Capability Maturity Model® (CMM®) (SW-CMM) [5] has been merged with the Electronic Industries Alliance/Interim Standard (EIA/IS 731) [6] and with a draft version of the Integrated Product Development (IPD) CMM [7] to form the CMM Integration℠ (CMMI℠) [8].

When a new model comes along, it is normal for the process group, and even for management, to panic. With the abundance of standards and models currently available, and a tendency to want organizational processes that comply with everything, being a member of a process group is not always a calm experience.

Early ISO 9000 auditors used to tell companies to be sure that their Quality Manual follows the ISO 9001 structure exactly so there would be no difficulty getting registration. Auditors may not still make that kind of recommendation, but two of the problems that already have occurred if companies have taken such advice are:

- An ISO 9000 structure may make it difficult for the companies to also comply with the SW-CMM.
- The ISO 9000 standards have changed! Does this mean every company should now restructure its company's quality manual even though the manual is working well?

The question, therefore, becomes how to use a capability model in a manner that will work for years.

This article discusses the nature of capability models first, followed by descriptions of organizational processes and how they should be structured compared to the models. Integrated models and integrated processes are addressed as a special and optimal case. Mapping models and addressing the gaps is presented first in an abstract manner, and then suggestions are made that are specific for making the transition from systems engineering models to an integrated capability model.

"Increasingly, government agencies and their prime contractors are asking bidders to demonstrate maturity in their development efforts."

What Is a Capability Model?

Capability models define the characteristics of good processes and avoid prescribing how the processes must be enacted. The purpose of capability models is to establish a process improvement road-map upon which a route can be drawn from "where we are today" to "where we want to be." In order to determine "where we are today," an organization performs an appraisal, sometimes with the aid of an outsider with specific expertise in the model.

Capability models are not processes. They intentionally do not address a particular life cycle or sequence of activities. They also do not have the necessary characteristics of processes; namely, they do not include inputs, outputs, tasks, roles and responsibilities, and entry and exit criteria.

Rather, capability models are sometimes thought of as containing requirements for good processes. Capability models ask for processes to be written that have inputs, outputs, tasks, roles, entry and exit criteria, verification mechanisms, and measurements. These processes will be unique to the organization for which they are written.

Why Use a Capability Model?

There are several reasons why organizations use capability models:

- **To verify process contents.** Capability models encapsulate basic industry knowledge for an organization to use to help improve quality, customer satisfaction, productivity, and cycle time. Many companies examine the models to understand the basic practices: Is practice "X" necessary? If so, is it performed somewhere?
- **To demonstrate progress.** Another primary use of capability models is to demonstrate year-to-year improvement. Periodic ratings of the organization's processes against the models are one indicator of such long-term improvements.
- **To benchmark.** A model can be used to validate process improvement progress in comparison with competitors. Increasingly, government agencies and their prime contractors are asking bidders to demonstrate maturity in their development efforts. Companies that have received high ratings against a

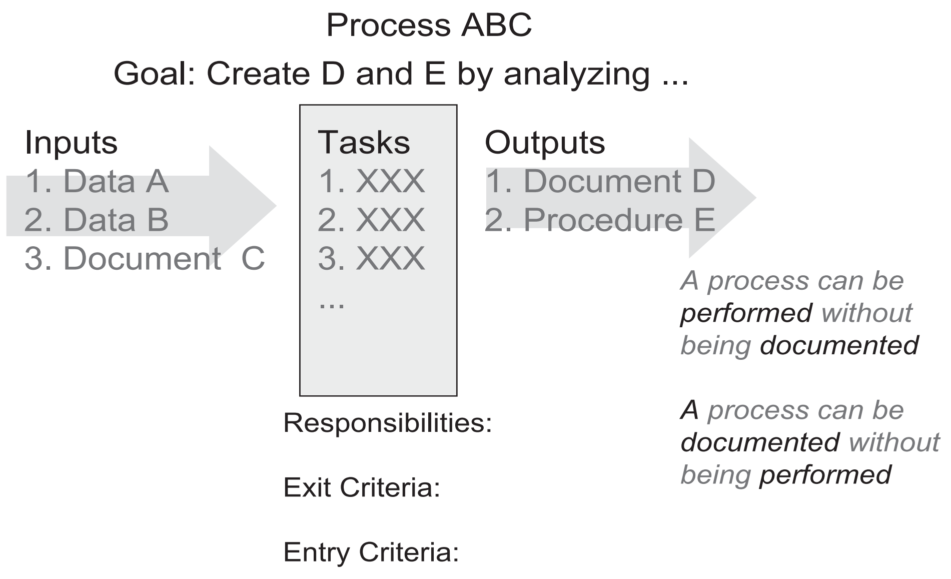


Figure 1: *Process Description Template*

well-known standard are considered to be more reliable, less risky contractors.

- **To structure new processes.** Organizations that have not yet captured their basic engineering practices in documented processes frequently will look at capability models as a list of what needs to be included.
- **Other uses.** Systems engineering capability models also can be used to help establish a definition of systems engineering to which the entire organization can subscribe. The structure of staged models defines a suggested road map for improvement.

Structure of Capability Models

This section describes the structure of capability models from the point of view of systems engineering models.[2]

Continuous models contain between 17 and 24 process areas, also called focus areas or key process areas. These describe the activities that the processes must include in order to satisfy the model. The process areas are grouped into categories, but it is the process areas rather than the categories that are rated.

Models also describe how the process areas are performed, or capability. "Staged" models such as the SW-CMM include "common features," such as "commitment to perform, ability to perform, and verification," as a part of each process area. "Continuous" models such as EIA/IS 731 contain "generic practices" as a second axis, against which each process area is rated. Generic practices address the same aspects as the common features (e.g., resources, responsibility, verification, planning); in fact, some systems engineering models group generic practices into categories called "common features."

What Is an Integrated Model?

The first capability model, the SW-CMM, addressed software development, or more precisely, the management of software development projects. Later models addressed systems engineering [9, 10], integrated product development [7], and other aspects ranging from human resources [11] to security [12]. Because these models targeted different functions, organizations often found themselves using multiple models separately in different areas of the organization.

An example is the Federal Aviation Administration (FAA). The Systems Engineering CMM (SE-CMM, a predecessor of EIA/IS 731), the SW-CMM, and the Software Acquisition-CMM (SA-CMM) [13] were in use in different departments. The FAA released the FAA Integrated Capability Maturity Model (FAA-iCMM)[3] in 1997 to unify its process improvement efforts [14]. The FAA-iCMM included every practice in every source model, as is shown by the detailed traceability tables included with each process area. The Software Engineering Institute (SEI) approved this model.

In late 2000 the SEI, serving as the custodian for a collaboration of industry and government groups, published the CMMI model. This model integrates most of the practices of EIA/IS 731 with the SW-CMM Version 2.0 (an unreleased expansion of the released Version 1.1) and much of the unreleased IPD-CMM. One version of the CMMI also includes some acquisition practices related to the SA-CMM.

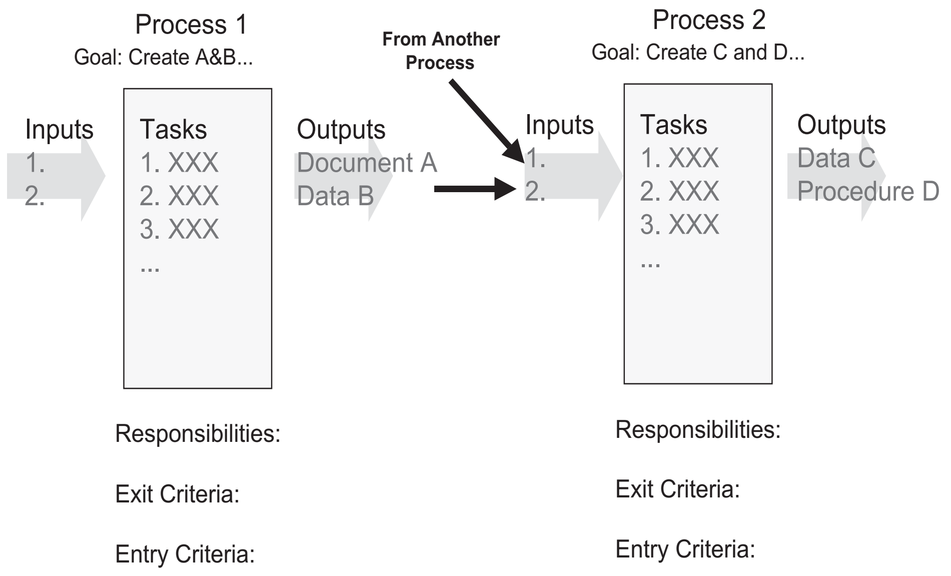
What Good Is an Integrated Model?

Integrated capability models can do two things for the organization. First, they provide a common framework and terminology that encourages process engineering groups in the various disciplines to cooperate, both on the appraisals and also ideally on creating the organization's suite of processes. Such an integrated suite of processes can be significantly beneficial, as many of the process problems in organizations can be traced to poor interfaces between groups. Second, their associated appraisal methods can provide combined appraisals that cost less to perform than separate appraisals

What Is a Process?

A process is a sequence of steps to

Figure 2: *Process Relationships*



achieve a given outcome.[4] Capability models require process descriptions to be documented. What is actually practiced must be available for reference and for instruction of newcomers.

Figure 1 shows a template for a process description. A process description should include the process goal, its inputs and outputs, a sequence of steps or tasks, and also preferably responsibilities or roles, and entry and exit criteria (when the process can begin, and when it is finished). Most process descriptions are textual, but for the visually oriented, a process diagram is helpful. Some organizations prefer to model processes in graphical or database-based tools.

How Are Processes Related?

Relationships among two or more processes should be described in a diagram that depicts the interactions among the processes (see Figure 2). For example, input needed by one process should be identified as the output of another process. Figure 3 shows the relationships of processes in a life-cycle engineering model surrounded by management and support.

Figure 4 shows an integrated systems and subsystems engineering process architecture. This architecture, based on the engineering "vee," shows system processes (Level n) and recursive sub-processes (Level n+1) such as those for subsystems, units, and components. Management processes perform overall planning and monitoring, and control baselines created in development processes as they are handed off from design to develop to integrate and test.

Several types of process architectures are available. The Software Productivity Consortium does not recommend a particular architecture. Organizations should structure processes in the way that best reflects that company's business.

What Are Integrated Processes?

As addressed above, disparate groups can initiate multiple process improvement efforts within an organization. Basing these efforts on different capability models and standards leads to disparate sets of processes such as software processes and systems engineering processes that neither integrate nor even have a defined interface. Unfortunately, integrated models do not provide guidance on how to integrate such disparate processes into a robust and optimized set of organizational processes.

There are several approaches to integrating these processes. Two starting points are expansion and integration. The following concepts should be addressed whatever the starting point.

Expansion

If only one group within the organization has documented its processes to date, developing an organization-wide integrated set mostly consists of expanding the scope of the documented processes and expanding participation in writing, reviewing, and using processes to additional groups. Note that the process architecture may require modification to be able to gracefully incorporate processes needed for the expanded scope.

Integration

If two or more groups have already begun documenting processes, there is more material with which to work but possibly more likelihood of misunderstanding. Do not approach the integration effort as trying to determine whose processes are better. Instead, analyze and discuss the following aspects of both sets of processes:

- **Boundaries.** What are the boundaries of the current processes? Do they encompass "everything the software department does" plus, say, "the engineering life cycle of a system, with emphasis on the contribution made by the program systems engineering group?" Are training, human resources, configuration management, program management, or engineering support environment maintenance covered in any way by either or both sets of processes? Draw out the overlaps and gaps.
- **Interfaces.** Next define the interfaces that should occur between the two sets of processes.[5] What should the software configuration management process provide to the systems engineering, or hardware configuration management process, and vice versa? Should they have the same data structures, or not? What data should pass from one group to the next? Is the software group expecting a complete and invariant group of requirements from the "systems engineers?"

Also note where the same processes seem to appear in both places: Requirements development and program planning are two typical examples. Should the software group and systems group plan the same way? Should they use the same data upon which to base task estimates?

Define the data in separate tables at first; for example, data that software is expecting to give and receive from other places vs. the systems group data. Clearly the systems group will be interacting with more than software; it's likely that the software group does the same. Where do the interfaces agree? Where do they disagree? What is missing from one data definition that is in

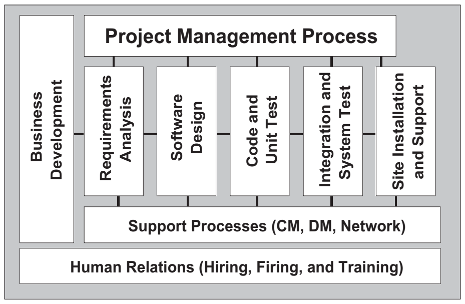
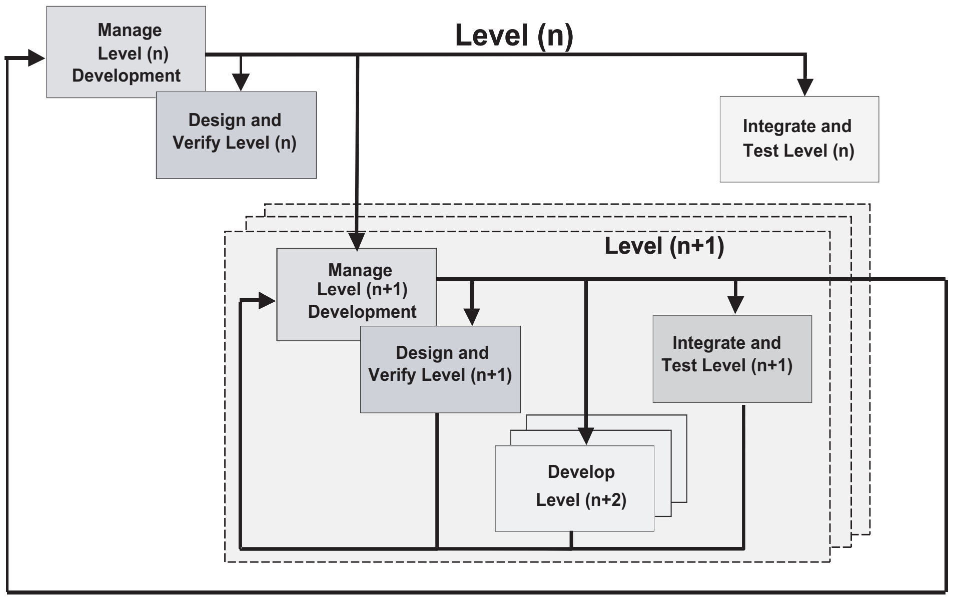

Project Management Process
Business Development
Requirements Analysis
Software Design
Code and Unit Test
Integration and System Test
Site Installation and Support
Support Processes (CM, DM, Network)
Human Relations (Hiring, Firing, and Training)

Figure 3: *Life-Cycle-Based Process Architecture*

Figure 4: *Integrated Systems and Software Process Architecture*


Level (n)
Manage Level (n) Development
Design and Verify Level (n)
Integrate and Test Level (n)
Level (n+1)
Manage Level (n+1) Development
Design and Verify Level (n+1)
Integrate and Test Level (n+1)
Develop Level (n+2)

the other? It may be necessary to create joint tiger teams to resolve conflicts – better to resolve them early in the processes.

- **Participation.** Is the systems engineering group expecting software participation in determining what requirements should be allocated to software? Do (or should) systems engineers refrain from giving any requirements to software developers until the requirements are complete and stable? How does a systems change ripple through the systems and subsystem (e.g., software) processes? Work out these and similar issues in joint process meetings.

Integration Artifacts

The artifacts that will arise from the integration effort may include the following:

A) Processes that did not change or have minor adaptations:
- Software processes that will remain strictly software processes as they were before, e.g., code walk-throughs.
- Software processes that remain software processes but have to change their interfaces to match with systems, e.g., software requirements analysis.
- Systems processes that remain the way they were, e.g. factory acceptance tests.
- Systems processes that remain systems processes but have to change their interfaces to map with software and other processes, e.g., System Critical Design Review.
- Processes that don't exist yet in a documented form, but that's acceptable for now (other processes should acknowledge them and describe the assumed interface), e.g., program budgeting process.

B) Processes substantially reorganized to improve the effectiveness of the organization's set of standard processes:
- Systems processes that are no longer strictly systems processes; now merged with software or other subsystem area processes (or into other processes such as program management or configuration management processes), e.g., integration planning.
- Software processes no longer within the software group; now merged into systems or other processes, e.g., integration test processes or risk management processes.
- Processes that have to change to integrate better across the interface (perhaps delete one and use the other), e.g., requirements management.

C) New needed processes:
- Processes that need to be written now to address pressing organizational problems, e.g., a process for terminating a program, or for providing an engineering support environment

Adding Compliance to a New Model

It is time to look at how capability models and processes relate in order to determine how to add compliance to a new capability model. This requires understanding the whole picture and a fairly good understanding of three things:

1. The organization's processes.
2. The models with which the processes are currently compliant, and how the organization's processes relate to them. (Think of the models as the requirements, and the processes as the organization's solution. What is the traceability between the requirements and the solution?)
3. The model with which you wish to comply. (Think of this as another set of requirements with which your solution is already partly compliant.)

The goal is to understand where your existing processes do or do not comply with the new model. If you have documented traceability ("mappings") between the previous models and your processes, and between the new and previous models, you can infer mappings between your processes and the new model.

The bad news is that mapping is a lot of work for even one new model, and there are a lot of models out there.

The good news is that many models ask you for the same things such as "plan your work and work your plan," [15] so what you are doing is likely to transfer without much elaboration into compliance with a new model.

Consider the heritage of the new model. Especially if it is based on the models you have been using, the changes you will have to make in your organization's processes may be minimal.

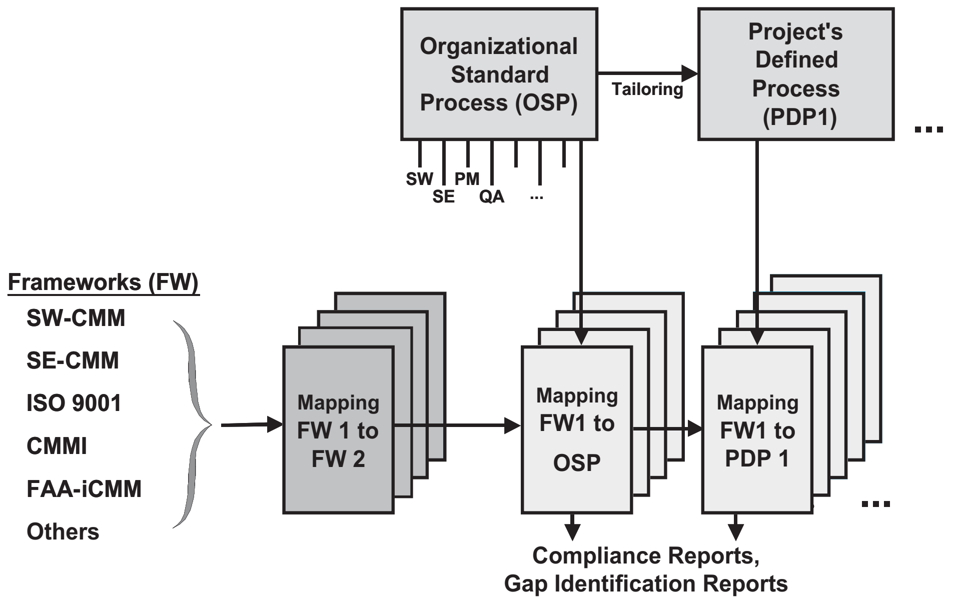
Mapping

When mapping to a single model, you are first setting up your processes to comply with a model; you need to know where in your processes you perform the practices that the model requires. A good mapping is two-way, meaning you could either look at an activity in a process and see what model practices this activity fulfills, or you could look at a practice in a model and see where these are included as activities in the organization's processes.

Once you have such a mapping, you should then continue mapping your processes to other models or standards of interest. Consortium members have access to a tool called Quagmap[6] that is pre-loaded with paragraph titles of popular models and shows how they map to each other. This allows input and mapping of an organizational set of processes to any of the preloaded models. Once a practice in one of your processes is mapped to a section of one model, the tool will give you an "inferred mapping" of the paragraphs in another model that also may map to this part of the process.

Figure 5 shows some of the many kinds of mappings, including mapping between the organizational standard processes and project processes. This kind

Figure 5: *Mapping Among Models and Processes*



of mapping, a record of project tailoring, is required for Level 3 in capability models.

When mapping processes to process areas, none of the capability models requires the processes of the organization to match the process areas of the model one for one. A process may combine the practices of two different process areas such as configuration management and data management, or planning and tracking. Several processes may be written to satisfy one process area (most organizations find a need to write more than one test process, perhaps unit test, integration test, and system test).

Gaps

A mapping between models should be able to highlight practices in the new model that were not well covered in the previous model. Concentrate on these gaps and address them on a case-by-case basis. Here are several ways to handle gaps:

- If your processes already include a practice because your business needs the practice, you may have covered the gap already.
- If your business truly does not need a new practice, you may be able to tailor out a requirement. For example, a system integrator may be able to omit practices related to implementing the design of a component.
- If the new model specifies additional practices that you do not currently perform but should, an activity might have to be added to (or changed in) one of your existing processes. An example might be certain measurement or risk management activities.
- If the old model did not include one aspect of your business that really does apply and should be improved, and the new one does, you might decide to write a new process to address this aspect (for example, appraisal of customer satisfaction).

You certainly do not, and should not, create a whole new set of processes to meet this new model.

It Sounds Like so Much Work

It is not as hard as it seems. The biggest difficulty in complying with a process improvement model is in making people understand what processes are, getting them written down, and getting management to enforce disciplined use of these processes. Specific activities that may be unique to a particular model are much less important than getting the organization to behave in a disciplined manner. Whether initial processes are configuration management processes, trade studies analysis processes, or even processes for running efficient meetings, it is easier to transfer discipline to additional processes than to perform processes in a disciplined manner in the first place.

Similarly, once organizationally standardized processes are understood, and programs know how their own processes differ from the standard processes, new programs begin to realize substantial savings in startup time and cost because the basic processes are already in place, and training is also easier. Adding new practices to the set of standard organizational processes, or even adding a few new processes, is easier than establishing an organization-wide process infrastructure in the first place.[7]

Moving From a Systems Engineering Model to an Integrated Model

The above suggestions address mappings in general. This section looks at specific changes an organization might need to make when the current model is a systems engineering model and the new model is an integrated capability model.

Maturity Rating

If you are used to continuous models and choosing process areas to emphasize, note that both the CMMI and the FAA-iCMM also include the concept of an official organizational rating, or maturity level, comparable to a rating in a staged model. An official Maturity Level 2 rating will require the achievement of the model's specified Level 2 process areas (and Maturity Level 3 requires both Level 2 and Level 3 process areas). Check to see if these are areas where you have put attention to date, and if you are seeking a rating, work hardest on any gaps.

Measurement

Measurement and the use of measurement data are called out much more explicitly in both integrated models than in the systems engineering models, and much earlier in CMMI (Level 2). This may be a surprise, but it is a good change. Work hard to comply with the measurement requirements and you will be rewarded with a much smoother process improvement effort afterward.

Process Improvement Infrastructure

Organizational process focus is much more clearly spelled out in the CMMI than in the systems engineering models. Systems engineering groups generally realize that they need a process infrastructure, but this process area, originally from the SW-CMM, helps define clearly what is needed.

Skills and Knowledge

The integrated models include process areas on training or organizational training. These may have a smaller scope than the Manage Competency focus area of EIA/IS 731, but may be more explicit in what must be done to comply. Review the practices against those of your processes that are mapped to manage competency.

More Good News

If you have reached mostly Level 2s or Level 3s in a systems engineering model, you will find you have most of the CMMI's requirements for comparable levels covered. This is much easier than starting over.

Some Questions to Ask

This section suggests some additional areas to investigate when adopting a new model.

Who does "systems engineering?" Do not assume the model considers "systems engineering" to be a particular organizational group. Most models do not specify who performs the systems engineering activities, just that they need to be done.

Do not make the mistake of assuming your "systems engineers" have to do everything, including configuration management, training, or program planning and tracking. While your systems engineers will need to be involved in these processes (managing the configuration of items they work on; requesting, taking, and possibly even planning training; and providing estimates and measurement data to program managers), many processes are run by groups other than those called "systems engineers." Furthermore, most subsystem areas, including software development, also have to perform systems engineering activities such as trade studies among potential architectures and validation against user need. Be sure they do not think they are exempt because the systems engineers do that.

How specific is the model? ISO 9000-2000 is fairly general, requiring certain aspects of processes to be well documented but not specifying much of their content (e.g., what is feasibility analysis). Some models such as EIA/IS 731 state that analyses must be done "as appropriate" while others, such as military standards and guidebooks, may explicitly specify a "feasibility analysis" deliverable and include a required data item description.

How specific should your processes be? The temptation is to document processes by specifying all items in great detail so that any auditor can see the documentation is complete. This is not a good idea because (a) it will be more difficult to get engineers to follow 500 pages of documents than 50, (b) 500 pages is more expensive to write, (c) 500 pages is more expensive to maintain, and (d) some models, such as ISO 9000-2000, are strict about following that which is documented. This emphasizes that you should not build rigidity into a documented process if the business needs it to be flexible.[8]

Finally, most processes that have been in place for some time probably either meet most business needs, or else everyone knows they are broken. Look first at those places where your processes are broken. You may find that the model calls out performance of some practices that you do not do, and should. You can use the model as leverage.

Conclusions

Complying with a new capability model is much easier than starting fresh if the organization already complies with another model, particularly when the new model is an evolution of the old model. Especially if an organization's processes describe essential business practices and are mapped to the old model rather than strictly based on it, it is fairly straightforward to use other mappings to infer what parts of the processes may be lacking, according to the new model. Use your standard process improvement process to incorporate changes to meet the requirements in the new model. Other than your process group, the rest of the organization only has to understand the processes they are already using and the few changes; they do not have to understand the new model at all.◆

References

1. ISO 9000 International Standards for Quality Management. International Organization for Standardization, Switzerland, 1994 and 2000.
2. Sheard, Sarah A. "The Frameworks Quagmire." CROSSTALK, Sept. 1997 <www.stsc.hill.af.mil>.
3. Sheard, Sarah A., "The Frameworks Quagmire, A Brief Look." Proceedings of INCOSE. 1997b.
4. Sheard, Sarah A., and J. Lake, eds. "Systems Engineering Standards and Models Compared." Proceedings of INCOSE. 1998.
5. Paulk, Mark, B. Curtis, M. Chrissis, and C. Weber, eds. Capability Maturity Model for Software, Version 1.1. Software Engineering Institute, Carnegie Mellon University, 1993.
6. Systems Engineering Capability Model EIA/IS 731. Electronics Industries Affiliates, 1999.
7. An Integrated Product Development Capability Maturity Model, Version 0.9. Software Engineering Institute and Enterprise Process Improvement Collaboration, Carnegie Mellon University, 28 Oct. 1996.
8. Capability Maturity Model Integration, Software-Systems-IPPD-Acquisition, Version 1.02b. Software Engineering Institute, Carnegie Mellon University, 2000.
9. Bate, Roger, et al. Systems Engineering Capability Maturity Model, Version 1.1. Software Engineering Institute, Carnegie Mellon University, 1995.
10. Systems Engineering Capability Assessment Model, Version 1.5. International Council on Systems Engineering (INCOSE), 1995.
11. Curtis, Bill, William E. Hefley, and Sally Miller, eds. People Capability Maturity Model. Software Engineering Institute, CMU/SEI-95-MM-02, Sept. 1995.
12. Hefner, R., et al. Systems Security Engineering Capability Maturity Model. International Systems Security Engineering Association, 1999.
13. Software Acquisition Capability Maturity Model. Version 1.01. Software Engineering Institute, Carnegie Mellon University, 1997.
14. Ibrahim, Linda, et al. The Federal Aviation Administration Integrated Capability Maturity Mode, Version 1.0. Federal Aviation Administration, Nov. 1997.
15. Ferguson, JoAn, and Sarah A. Sheard, eds. "Leveraging the CMM for ISO/IEC 12207 Compliance." Nuts and Bolts, IEEE Software, Sept./Oct. 1998.

Notes

1. This paper originally appeared in the Proceedings of the Eleventh Annual International Symposium of the International Council on Systems Engineering, Melbourne, Australia, July 2001.
2. Other models place more emphasis on practices (parts of process areas) and goals (groupings of practices).
3. The FAA also released an appraisal method that includes an internal and an external "full appraisal" as well as several reduced-cost appraisal versions. This method is called the FAA-iCMM Appraisal Method or FAM.
4. This is a fairly narrow definition of process, chosen for understandability. Other broader definitions can be more abstract.
5. For process robustness, involve as many process users as possible in any restructuring of processes.
6. Quagmap is a trademark of the Software Productivity Consortium.
7. In fact, the most mature organizations have the easiest time of all. The essence of Level 5 is continuous improvement. At this level, inserting new or better processes is just the normal way of doing business.
8. A classic case of a process that needs to be flexible is how to develop a design that meets requirements. Some engineers think linearly and can do Step 1, Step 2, Step 3, but the best engineers often develop a picture in their head and cannot follow a stepwise procedure. Your processes should specify outputs and required tasks to produce the outputs, but should leave open how to do the tasks, particularly those that can be done by a single person, whenever possible.

About the Author

Sarah A. Sheard has 20 years experience in systems engineering, on systems from satellites to air traffic control. Currently at the Software Productivity Consortium, Sheard helps companies begin systems engineering process improvement efforts and integrate the software and systems engineering aspects of both development work and process improvement efforts. Sheard is currently the chair of International Council on Systems Engineering's (INCOSE's) Measurement Technical Committee and in the past has served INCOSE as chair of the Communications Committee and as programs chair of the Washington Metropolitan Area Chapter. Sheard received a bachelor's degree in chemistry from the University of Rochester and a master's degree in chemistry from the California Institute of Technology.

2214 Rock Hill Rd.
Herndon, VA 20170-4227
Phone: (703) 742-7106
E-mail: sheard@software.org