PART ONE

About CMMI for Development

CHAPTER 1

INTRODUCTION

Now, more than ever, companies want to deliver products and services better, faster, and cheaper. At the same time, in the high-technology environment of the twenty-first century, nearly all organizations have found themselves building increasingly complex products and services. Today, a single company usually does not develop all the components that compose a product or service. More commonly, some components are built in-house and some are acquired; then all the components are integrated into the final product or service. Organizations must be able to manage and control this complex development and maintenance process.

The problems these organizations address today involve enterprise-wide solutions that require an integrated approach. Effective management of organizational assets is critical to business success. In essence, these organizations are product and service developers that need a way to manage an integrated approach to their development activities as part of achieving their business objectives.

In the current marketplace, there are maturity models, standards, methodologies, and guidelines that can help an organization improve the way it does business. However, most available improvement approaches focus on a specific part of the business and do not take a systemic approach to the problems that most organizations are facing. By focusing on improving one area of a business, these models have unfortunately perpetuated the stovepipes and barriers that exist in organizations.

Capability Maturity Model Integration (CMMI) provides an opportunity to avoid or eliminate these stovepipes and barriers through integrated models that transcend disciplines. CMMI for Development consists of best practices that address development and maintenance activities applied to products and services. It addresses practices that cover the product's lifecycle from conception through delivery and maintenance. The emphasis is on the work necessary to build and maintain the total product.

About Capability Maturity Models

In its research to help organizations develop and maintain quality products and services, the Software Engineering Institute (SEI) has found several dimensions that an organization can focus on to improve its business. Figure 1.1 illustrates the three critical dimensions that organizations typically focus on: people, procedures and methods, and tools and equipment.

But what holds everything together? It is the processes used in your organization. Processes allow you to align the way you do business. They allow you to address scalability and provide a way to incorporate knowledge of how to do things better. Processes allow you to leverage your resources and to examine business trends.

This is not to say that people and technology are not important. We are living in a world where technology is changing by an order of magnitude every ten years. Similarly, people typically work for many companies throughout their careers. We live in a dynamic world. A focus on process provides the infrastructure necessary to deal with an ever-changing world, and to maximize the productivity of people and the use of technology to be more competitive.

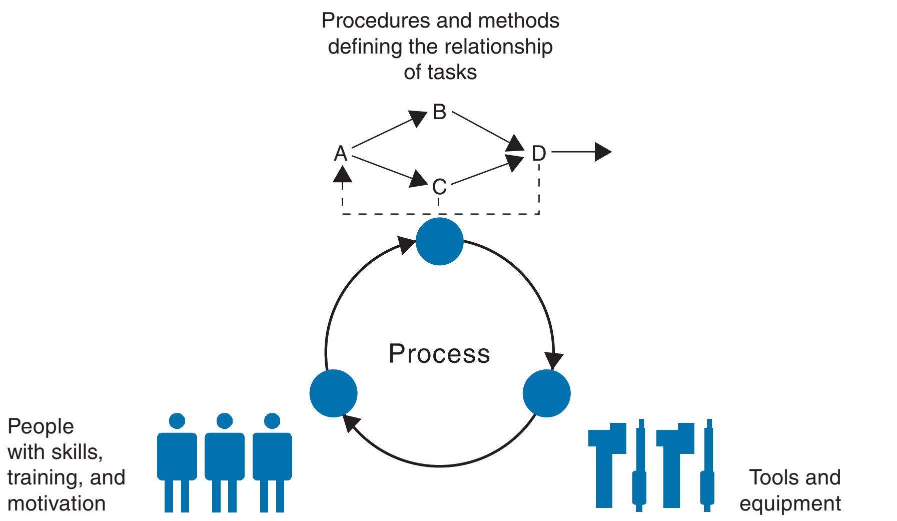
Procedures and methods defining the relationship of tasks
A
B
C
D
Process
People with skills, training, and motivation
Tools and equipment

FIGURE 1.1
The Three Critical Dimensions

Manufacturing has long recognized the importance of process effectiveness and efficiency. Today, many organizations in manufacturing and service industries recognize the importance of quality processes. Process helps an organization's workforce meet business objectives by helping them work smarter, not harder, and with improved consistency. Effective processes also provide a vehicle for introducing and using new technology in a way that best meets the business objectives of the organization.

In the 1930s, Walter Shewhart began work in process improvement with his principles of statistical quality control [Shewhart 1931]. These principles were refined by W. Edwards Deming [Deming 1986], Phillip Crosby [Crosby 1979], and Joseph Juran [Juran 1988]. Watts Humphrey, Ron Radice, and others extended these principles even further and began applying them to software in their work at IBM and the SEI [Humphrey 1989]. Humphrey's book, *Managing the Software Process,* provides a description of the basic principles and concepts on which many of the capability maturity models (CMMs) are based.

The SEI has taken the process management premise, "the quality of a system or product is highly influenced by the quality of the process used to develop and maintain it," and defined CMMs that embody this premise. The belief in this premise is seen worldwide in quality movements, as evidenced by the International Organization for Standardization/International Electrotechnical Commission (ISO/IEC) body of standards.

CMMI: History and Direction

by Watts S. Humphrey

To understand the future, we must consider where we have been, how we got here, and our current direction. I will describe the genesis of CMMI, the ideas that contributed to its design, some current issues, and thoughts on what to do next.

History

Five principal ideas from a broad array of fields originally inspired the CMMI model and appraisal process. These ideas were

1. Planning, tracking, and schedule management
2. Requirements definition and configuration control

3. Process assessment
4. Quality measurement and continuous improvement
5. Evolutionary improvement

We now know that these concepts apply to software and systems development work, but it was not obvious in 1986 when CMM development started, or in 1966 when I first tried to apply some of these ideas as IBM's director of software development.

Planning, Tracking, and Schedule Management
After completing my technical studies, I got an M.B.A. in manufacturing. Professor Judson Neff asserted that the only way to manage complex operations was to manage to detailed and precise plans. In my first job, I inherited a troubled development project. We made plans and tracked schedules and the project recovered.

Later, when in charge of all of IBM's programming development, my projects were again troubled and I again had everyone make and track to detailed plans. We did not miss a commitment for several years. We quickly got schedule control of an organization of 4,000 developers in 15 laboratories and 6 countries.

Requirements Definition and Configuration Control
I soon learned two more critical lessons. First, if you don't allow any requirements changes, you could build the wrong product and waste the entire development effort. Second, if you don't rigorously control changes, you will never finish development.

Process Assessment
Later, Dr. Art Anderson, IBM's senior vice president for development and manufacturing, asked me to fix IBM's semiconductor operations. We used an assessment method he had tried in IBM Research to help people solve their own problems. In IBM's Burlington, Vermont, semiconductor operation, Art explained that IBM could buy imported chips from Japan at lower prices than Burlington's costs. Even IBM could not afford to do this. If this operation was not soon competitive, we would shut it down. By assessing their own operations, this team solved their cost problem and soon became the world's lowest-cost semiconductor producer.

Quality Measurement and Continuous Improvement
The Burlington engineers controlled their costs through yield management. By more than doubling yield, they cut costs by more than

half. To do this, even the factory workers had to measure, track, and analyze every step of their work.

Evolutionary Improvement

When I was IBM's director of software quality and process, Ron Radice and I attended a Phil Crosby quality management course that used a five-level maturity model. Ron then used Crosby's maturity framework and Anderson's assessment strategy to accelerate IBM's software process improvement. A laboratory's first assessment was generally successful but the second and third ones were not. The problem was that the Crosby maturity levels were based on subjective attitude judgments rather than specific software activities.

Air Force Acquisition

When I retired from IBM, my first SEI assignment was to improve software source selection for the U.S. Air Force. We worked with Col. Jack Ferguson of the Air Force and Martin Owens and others from MITRE on a way to evaluate organizational capability. Organizations that used the best management and technical practices in their development projects seemed likely to do the best work, so we devised an 85-question questionnaire that covered

- Project planning
- Project tracking
- Schedule management
- Requirements management
- Configuration control
- Quality measurement
- Continuous process improvement

To rank the results, we grouped the 85 questions in a Crosby-like maturity framework. This became the first version of what ultimately became CMMI.

Current Challenges

The ideas behind CMMI came from many fields and benefited from many people's experiences. Three issues now lie ahead of us.

First, with increasing marketplace pressure, organizations often focus on maturity levels rather than process capability. Maturity levels cannot comprehensively measure organizational capability. They can indicate risky process areas or guide process improvement by describing a minimum set of activities necessary. We now see cases where high-maturity ratings do not indicate effective, high-maturity practices. It is not that the appraisal process is faulty or that organizations are dishonest, merely that the maturity framework does not look deeply enough into all organizational practices.

The second issue concerns adjusting the CMMI Framework and appraisal methods to address this problem. Without change, we can expect more cases where high-maturity ratings will not generally correlate with better performance. Two lessons from my earlier experiences suggest a way to address this issue.

1. To truly control complex and precise work, everyone must manage to detailed and precise plans.
2. Everyone must measure and manage quality.

To guide software developers in applying these principles to their work, the SEI developed the Personal Software Process (PSP) and the Team Software Process (TSP). When developers have used the PSP and TSP, appraisers have detected these practices with a CMMI appraisal. It therefore appears that the PSP and TSP can help foster mature developer practices. The SEI is now adapting the PSP and TSP to systems development and acquisition work.

This, however, leads to the third issue: flexibility. CMMI does not define in detail how to do development work; the focus is on what to do. However, the PSP and TSP specify how project planning, tracking, and quality management are performed. The issue concerns ways to incorporate such practices and principles into the CMMI model and method without switching the focus from what to how. The need is to encompass these proven principles and practices without constraining development organizations as the technology advances. The SEI is working on these issues as we strive to improve the effectiveness of these methods and models.

CMMs focus on improving processes in an organization. They contain the essential elements of effective processes for one or more disciplines and describe an evolutionary improvement path from ad hoc, immature processes to disciplined, mature processes with improved quality and effectiveness.

The SEI created the first CMM designed for software organizations and published it in a book, *The Capability Maturity Model: Guidelines for Improving the Software Process* [SEI 1995].

The SEI's book applied the principles introduced almost a century ago to this never-ending cycle of process improvement. The value of this process improvement approach has been confirmed over time. Organizations have experienced increased productivity and quality, improved cycle time, and more accurate and predictable schedules and budgets [Gibson 2006].

Evolution of CMMI

CMMI: From the Past and into the Future

by Mike Phillips

As we launch this update to the CMMI models, it is appropriate to view the heritage that led us to this point, and give some insight into where we believe the work is headed in the future.

The Past

Models with levels of improvement go back to the emphasis on manufacturing quality expressed by Philip Crosby. Shortly after the creation of the SEI, the U.S. Air Force asked the SEI to identify key practices that a contractor had to perform to deliver software-intensive systems reliably. By 1991, this tracking of practices, and measurement across a stepped approach for improvement like that pioneered by Crosby, had matured into the Capability Maturity Model for Software (SW-CMM).

The success of this model for one engineering discipline led to similar efforts for other elements of the product development community. Interest in such models for systems engineering process improvement led to two models produced in 1994. The first was the Systems Engineering CMM, created by the Enterprise Process Improvement Collaboration (EPIC), with SEI participation.

The second model, the Systems Engineering Capability and Assessment Method, or SECAM, was created by the International Council on Systems Engineering (INCOSE). Four years later these two models were successfully merged into Electronic Industries Alliance (EIA) Interim Standard 731 as a result of a collaborative effort of EIA, EPIC, and INCOSE. In 1996, a sister to the SW-CMM was created to cover key practices in software acquisition—the Software Acquisition Capability Maturity Model, or SA-CMM. Concerns about preserving and enhancing the capabilities of developmental engineering staff led the SEI to create the People Capability Maturity Model (P-CMM) in 1995.

That year, work was also underway at the SEI to produce an update to the SW-CMM, and to produce a model that would capture concurrent engineering practices in an Integrated Product Development CMM. The Institute's sponsor, the U.S. Department of Defense (DoD), determined that these efforts should be merged into an integrated model, to be called the model we now know as Capability Maturity Model Integration (CMMI).

The feasibility of integrating a diverse set of maturity models had been demonstrated earlier that year by the Federal Aviation Administration (FAA), which had developed an integrated capability maturity model (FAA-iCMM v1.0). Due to the widespread focus on integrated product and process development (IPPD) by the DoD and industry, it was decided that the initial focus of the CMMI effort would be integration of systems engineering, software engineering, and IPPD.

The CMMI Product Team produced two draft versions of the CMMI models before settling on the combination that became an initial release in 2000, which included versions for systems engineering, software engineering, and integrated product and process development. Due to minor changes as we released versions, these became known as v1.02 by the December 2000 release. At the same time, we released a draft version to provide initial thinking about acquisition, called v1.02d.

We then took a year to gather results from the initial release before producing a refinement of the material that we wished to stabilize for a longer period. The first of these models was released in December 2001 as v1.1. With CMMI Steering Group approval, we added a variant that included some of the acquisition practices as a Supplier Sourcing addition in April 2002. This version became the basis for the previous edition of this Addison-Wesley book.

The Present

The use of CMMI v1.1 has exceeded our expectations. As I write this in February 2006, the SEI and its Partners have trained more than 45,000 people and conducted about 1,500 appraisals to measure process improvement progress. The model has become a de facto standard for software-intensive system development, and has shown value for demonstrating process discipline against governance audits like Sarbanes-Oxley.

As we investigated what changes might be needed for a next version, the CMMI Steering Group agreed that we should reexamine the architecture of the existing version to see if it might be improved. The Architecture team agreed that with some relatively minor changes, we could make the overall framework more easily extensible into domains of interest to the community that found some parts of the existing models very useful, but other parts difficult to apply in their domains. The changes to the architecture that were approved had little impact on the development elements, but did allow some consolidation of the practices that you will see inside this book. They were, however, highly significant in allowing synergistic expansion into areas closely related to development, like services and acquisition.

The Future

We see both near-term and more distant opportunities to expand the value of the CMMI Product Suite. As I mentioned earlier, we have a near-term opportunity to expand coverage through the use of variants we are currently calling "constellations." Two that are under initial development are constellations for acquisition and for services. Each will have elements that are the same as those in this book, plus some elements of coverage that may be unique to the domain, and perhaps some elements shared with another constellation.

While these new constellations are in early development, the clear commitment of both development teams for these domains is to maximize the commonality across the constellations. This commonality will aid in reducing the amount of training or appraisal preparation required for the various areas of CMMI coverage. Commonality will also aid us in addressing other standards, such as ITIL, in a complementary fashion.

We have also heard from the community of its interest in covering other areas that deserve the focused attention of process improvement. While some may need the full treatment of a new constellation, others may be best addressed by providing interpretive guidance or expanded coverage of specific practices and goals for an area. For areas like safety, security, and design engineering, we will be investigating approaches that will build on the value of the CMMI Framework to provide support to more and more of the community.

Since 1991, CMMs have been developed for myriad disciplines. Some of the most notable include models for systems engineering, software engineering, software acquisition, workforce management and development, and integrated product and process development (IPPD).

Although these models have proven useful to many organizations in different industries, the use of multiple models has been problematic. Many organizations would like their improvement efforts to span different groups in their organizations. However, the differences among the discipline-specific models used by each group, including their architecture, content, and approach, have limited these organizations' capabilities to broaden their improvements successfully. Further, applying multiple models that are not integrated within and across an organization is costly in terms of training, appraisals, and improvement activities.

The CMM Integration project was formed to sort out the problem of using multiple CMMs. The CMMI Product Team's initial mission was to combine three source models:

1. The Capability Maturity Model for Software (SW-CMM) v2.0 draft C [SEI 1997b]
2. The Systems Engineering Capability Model (SECM) [EIA 1998][1]
3. The Integrated Product Development Capability Maturity Model (IPD-CMM) v0.98 [SEI 1997a]

The combination of these models into a single improvement framework was intended for use by organizations in their pursuit of enterprise-wide process improvement.

These three source models were selected because of their widespread adoption in the software and systems engineering communities and because of their different approaches to improving processes in an organization.

Using information from these popular and well-regarded models as source material, the CMMI Product Team created a cohesive set of integrated models that can be adopted by those currently using the source models, as well as by those new to the CMM concept. Hence, CMMI is a result of the evolution of the SW-CMM, the SECM, and the IPD-CMM.

Developing a set of integrated models involved more than simply combining existing model materials. Using processes that promote consensus, the CMMI Product Team built a framework that accommodates multiple disciplines and is flexible enough to support the different approaches of the source models [Ahern 2003].

1. The Systems Engineering Capability Model is also known as Electronic Industries Alliance 731 (EIA 731).

CMMI: Integration and Improvement Continue

by Bob Rassa

It is hard to believe that CMMI is almost ten years old: nearly five years in development and slightly more than five years since its first release. When the National Defense Industrial Association's (NDIA) Systems Engineering Division was just a fledgling, I, as the director of systems engineering, got together with Mark Schaeffer, of the office of the U.S. Under Secretary of Defense, and we decided that the current CMM environment was diverging. We saw that software was going in one direction and systems engineering was going in another direction, and that other discipline-specific maturity models were popping up. Because of the nature of this environment, we decided it was time to take positive action. Despite a competent software process maturity model, the U.S. DoD realized that software problems were still a large cause of program failures. Bringing stronger systems engineering into play was considered an important part of solving this problem. It was clear that the divergence of maturity models kept these two communities apart.

We consulted with Roger Bate, now the CMMI chief architect, about the feasibility of building an integrated model that could support best practices in multiple areas. After conducting in-depth analyses, Roger confirmed that this new idea could be implemented. We then decided it was time to create an integrated CMM.

The result of this decision was CMMI-SE/SW, an integrated maturity model that brought these two important disciplines together in terms of process maturity. As CMMI development ensued, we realized that most of the critical processes for these disciplines were in fact common, thus validating the concept. Shortly after the initial release of CMMI-SE/SW in November 2000, we released the IPPD environment, and then Supplier Sourcing (SS) to round out the product suite.

Even though it was designed to apply to all aspects of product or service design (including hardware design) the model retained the SE and SW designations to preserve ties to the legacy models (i.e., EIA 731 and the Software CMM). Whether this was a good decision is moot, since the results are impressive. Far more than 1,000 Class A appraisals have been reported in just four years after the release of CMMI, and as of January 2006, more than 45,000 individuals have received "Introduction to CMMI" training, and this number typically increases between 1,000 and 1,500 per month. The success in CMMI recognition and adoption by the industrial complex is undeniable.

One downside to the naming convention used for CMMI models has been the view that the principal activities within the organization to which CMMI applies are software and systems engineering, and of course, nothing can be further from the truth. To gain maximum benefit from CMMI adoption and implementation, CMMI must be applied to the entire development structure of the organization, and to that end the latest release of CMMI (v1.2) is called CMMI for Development (CMMI-DEV) to clearly signify its application to the full spectrum of product and service design.

The v1.2 architecture has also undergone a slight morphing to accommodate two additional applications of CMMI, designated CMMI for Acquisition (CMMI-ACQ) and CMMI for Services (CMMI-SVC). CMMI-ACQ is the name of a process maturity model for acquisition organizations. CMMI-SVC is the name of the model for organizations providing services. Both of these models are being developed at the request of the industrial complex and will appear shortly after CMMI-DEV v1.2 is released. These two additional "constellations," as they are called in CMMI parlance, will round out the product suite.

One additional legacy of the pre-CMMI models is retained in v1.2—namely, the concept of both staged and continuous representations—but they are taught in a common "Introduction to CMMI" course which is the only introductory course now offered.

CMMI is truly state of the art in terms of process maturity, and substantive benefits have been reported to and summarized by the designated CMMI steward, the SEI, of Carnegie Mellon University. However, to be truly effective CMMI must be applied in a conscientious manner within the organization. When we started the initial development of CMMI, it was well publicized that its purpose was to integrate the divergent maturity models. We soon realized that the real purpose that should have been communicated as the ultimate benefit of CMMI was that CMMI would integrate the design disciplines in terms of both process and performance. To achieve this ultimate benefit, care is needed to make sure that integrated processes are put into place within the organization, that such processes are implemented across the enterprise on all new programs and projects, and that such implementation is done in a thorough manner to assure that new programs start out on the right foot.

This book provides the latest guidance toward CMMI implementation. It covers all the specifics, addresses nuances of interpretation, and contains expert advice useful to both new and experienced practitioners. Hundreds of process improvement experts have contributed to CMMI development, and many of them contributed their expertise to this volume for the benefit of the industrial complex. We trust you will enjoy their work.

History of CMMs

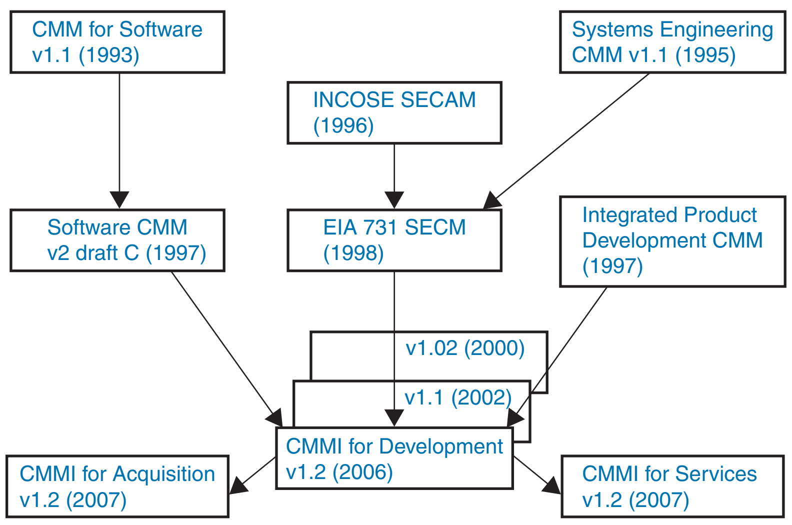
CMM for Software v1.1 (1993)
Systems Engineering CMM v1.1 (1995)
INCOSE SECAM (1996)
Software CMM v2 draft C (1997)
EIA 731 SECM (1998)
Integrated Product Development CMM (1997)
v1.02 (2000)
v1.1 (2002)
CMMI for Development v1.2 (2006)
CMMI for Acquisition v1.2 (2007)
CMMI for Services v1.2 (2007)

FIGURE 1.2
The History of CMMs

Since the release of CMMI v1.1, we have seen that this improvement framework can be applied to other areas of interest [SEI 2002a, SEI 2002b]. To apply to multiple areas of interest, the framework groups best practices into what we call “constellations.” A constellation is a collection of CMMI components that are used to build models, training materials, and appraisal documents.

Recently, the CMMI model architecture was improved to support multiple constellations and the sharing of best practices among constellations and their member models. Work has begun on two new constellations: one for services (CMMI for Services) and the other for acquisition (CMMI for Acquisition). Although CMMI for Development incorporates the development of services, including the combination of components, consumables, and people intended to meet service requirements, it differs from the planned CMMI for Services (CMMI-SVC), which focuses on the delivery of services. The CMMI models that have been available in the community prior to 2006 are now considered part of the CMMI for Development constellation.

The Architecture of the CMMI Framework

by Roger Bate

The CMMI Product Suite has been published in two main versions: version 1.0 released in 2000, and version 1.1 released in 2002. As the product suite was used in disparate industries and organizations, it became apparent that CMMI could be applied to all kinds of product development, especially if the terminology was kept general for similar practices.

A further revelation was that the process and project management practices of the model are suitable for a wide range of activities besides development. This discovery led me to propose that we should enable the expansion of CMMI, including the extension of the scope of the CMMI Framework, by creating a new architecture for the CMMI Framework.

This new architecture would accommodate other areas of interest (e.g., services, acquisition, and development). I was musing one day about the valuable best practices that were contained in models. I began to think of them as the *stars* of process improvement. I pushed this metaphor a little further to call the collection of components that would be useful in building a model, its training materials, and appraisal documents for an area of interest a *constellation*. This was the beginning of the architecture that was eventually created.

There are two primary objectives for the CMMI Framework architecture.

1. Enable the coverage of selected areas of interest to make useful and effective processes.
2. Promote maximum commonality of goals and practices across models, training materials, and appraisal methods.

These objectives pull in opposite directions; therefore, the architecture was designed as a bit of a compromise.

The CMMI Framework will be used in CMMI v1.2 and beyond to accommodate additional content that the user community indicates is desirable. The framework contains components used to construct models and their corresponding training and appraisal materials. The framework is organized so that the models constructed will benefit from common terminology and common practices that have proven to be valuable in previous models.

The CMMI Framework is a collection of all model components, training material components, and appraisal components. These components are organized into groupings, called "constellations," which facilitate construction of approved models and preserve the legacy of existing CMM and CMMI models.

In the framework, there are constellations of components that are used to construct models in an area of interest (e.g., Acquisition, Development, and Services). Also in the framework, there is a CMMI model foundation. This foundation is a skeleton model that contains the core model components in a CMMI model structure. The content of the CMMI model foundation is apropos to all areas of interest addressed by the constellations. A CMMI model for a constellation is constructed by inserting additional model components into the CMMI model foundation.

Since the CMMI architecture is designed to encourage preserving as much common material as is reasonable in a multiple-constellation environment, the framework contains and controls all CMMI material that can be used to produce any constellation or model. However, a majority of components of the framework are expected to be shared among most of the constellations and models.

CMMI models have a defined structure. This structure is designed to provide familiar placement of model components of various constellations and versions. If you look at the structure of a process area, you'll see components including Process Area Name, Category, Maturity Level, Purpose, Introductory Notes, References, and Specific Goals. You will also find that every process area in this model (i.e., CMMI for Development) and all other CMMI models produced from the CMMI Framework have the same structure. This feature helps you to understand quickly where to look for information in any CMMI model.

One of the benefits of having a common architecture and a large portion of common content in the various models is that the effort required to write models, train users, and appraise organizations is greatly reduced. The capability to add model components to the common process areas permits the models to expand their scope of coverage to a greater variety of needs. In addition, whole new process areas may be added to provide greater coverage of different areas of interest in the constellations.

CMMI models have a great deal of well-tested content that can be used to guide the creation of high-performance processes. The CMMI architecture permits that valuable content to continue to work in different areas of interest, while allowing for innovation and agility in responding to new needs.

So, you see that CMMI is growing beyond the star practices of the three original source models, and into constellations. This expansion into the "galaxy" is possible only with a well-thought-out and designed architecture to support it. The CMMI architecture has been designed to provide such support and will grow as needed to continue into the future.

CMMI for Development

The CMMI for Development constellation consists of two models: CMMI for Development +IPPD and CMMI for Development (without IPPD). Both models share much of their material and are identical in these shared areas. However, CMMI for Development +IPPD contains additional goals and practices that cover IPPD.

Currently, only one model is published since the CMMI for Development +IPPD model contains the full complement of practices available in this constellation, and you can derive the other model from this material. If you are not using IPPD, ignore the information that is marked "IPPD Addition," and you will be using the CMMI for Development model. If the need arises or the development constellation is expanded, the architecture will allow other models to be generated and published.

CMMI for Development is the designated successor of the three source models. The SEI has retired the Software CMM and the IPD-CMM. EIA has retired the SECM. All three of these models are succeeded by CMMI for Development.

The best practices in the CMMI models have gone through an extensive review process. CMMI version 0.2 was publicly reviewed and used in pilot activities. The CMMI Product Team evaluated more than 3,000 change requests to create CMMI version 1.0. Shortly thereafter, version 1.02 was released, which incorporated several minor improvements. Version 1.1 incorporated improvements guided by feedback from early use, with more than 1,500 change requests submitted as part of the public review, and hundreds of comments as part of the change control process.

CMMI version 1.2 was developed using input from nearly 2,000 change requests submitted by CMMI users. More than 750 of those requests were directed at CMMI model content. As you can see, not only is CMMI widely adopted, but it is improved based on the feedback received from the community.

Stewardship of the CMMI Product Suite

by Bill Peterson

CMMI has become widely used in various industries around the world. CMMI's development and enhancement are sponsored by two organizations: the Office of the Under Secretary of Defense for Acquisition, Technology, and Logistics (OUSD/AT&L), and the National Defense Industrial Association (NDIA) Systems Engineering Committee (SEC). These two organizations, and others from government, industry, and the SEI, work hand in hand to ensure that CMMI continues to meet the needs of and is available for government and industry use.

The SEI serves as the steward of the CMMI Product Suite. As steward, the SEI has the responsibility to coordinate CMMI-related activities and communicate with CMMI users and the public. This role fits nicely with the SEI's mission to "advance software engineering and related disciplines to ensure the development and operation of systems with predictable and improved cost, schedule, and quality." Certainly CMMI aids organizations in meeting these goals, and the SEI's role as steward enables it to maintain quality in the product suite, promote proper application of CMMI, and communicate to those who need information about CMMI. The SEI as steward provides coordination to ensure that members are involved across government, industry, and academia.

The CMMI Steering Group is the executive team that steers the development of CMMI and makes decisions on the direction of the CMMI Product Suite. The steering group consists of members appointed from government, industry, and the SEI. This group now directs and oversees CMMI maintenance and enhancements, the introduction of new disciplines to be included in CMMI, and the chartering of new development and maintenance projects. The group also reviews plans and sponsorship support of work proposed by others.

The development and maintenance of CMMI rest with the CMMI Product Team, a multiorganizational mix of people from government, industry, and the SEI. The CMMI Product Team is composed of different project teams that develop and maintain CMMI products, such as CMMI models and "Introduction to CMMI" training, as well as the SCAMPI A appraisal method. Team members represent the various disciplines and domains covered in CMMI's best practices. Also included in the product team are a project manager and a chief architect. The project manager coordinates the work of the product teams and the chief architect coordinates with the CMMI Steering Group to chart the future expansion of CMMI models and related products.

As part of the stewardship role, the SEI coordinates a change request process to gather feedback about the CMMI Product Suite from CMMI users and the public. Any individual from anywhere in the world can submit a change request. These change requests are reviewed and designed into product improvements proposed by the Product Team, which are then reviewed and approved or rejected by the CMMI configuration control board (CCB) for inclusion into an update release of the product suite. The CCB consists of members from multiple organizations and represents different roles/segments of the CMMI user base. Change requests that would result in significant change to the CMMI Product Suite are also reviewed with the CMMI Steering Group.

All of these groups must interact in order to continually improve the CMMI Product Suite in a way that best meets the needs of CMMI users. Further, the results of the efforts of these groups must be communicated to CMMI users and the public.

The activities of the CMMI steward support and facilitate the coordination of all of these groups as well as the maintenance and evolution of the CMMI Product Suite. The steward is responsible for providing project management coordination of CMMI Product Suite maintenance and enhancements. This responsibility includes facilitating the gathering of feedback from CMMI users and the product team, distributing new and improved best practices in the field, and integrating new disciplines or additional features (as directed by the steering group).

Beside managing updates to the CMMI Product Suite, the steward has other responsibilities that occur on a regular basis and include the following:

- Providing broad access to CMMI models, the appraisal method definition, and other information on the SEI Web site
- Creating and maintaining training materials
- Conducting and managing the authorizations of CMMI instructors and lead appraisers and licensing the network of CMMI partners
- Maintaining a quality assurance program that oversees appraisal and training activities to ensure that they support results that are valid, consistent, repeatable, comparable, and of the highest integrity and credibility

- Communicating to CMMI users and the public about the CMMI Framework and Product Suite
- Supporting relevant national and international standardization activities as part of leveraging developer and user investments in CMMI
- Identifying, measuring, and reporting on successes and barriers to success as experienced by CMMI users
- Funding stewardship activities
- Ensuring that standard procedures are followed for soliciting, processing, reporting, and testing improvements to existing CMMI work products

CMMI's sponsors, steering group, product team, CCB, and steward all share a long-term vision for CMMI. These groups see CMMI continuing to be widely adopted both nationally and internationally. Because CMMI supports the business goals and objectives of organizations that use it, the groups also see CMMI as the framework of choice for process improvement across multiple disciplines in an organization. Finally, these groups envision CMMI as being supported by appraisal methods that ensure efficient and cost-effective appraisals that provide the highest quality and integrity measures of an organization's capabilities.

The role of the CMMI steward, with the assistance of all the other CMMI groups, continually improves the CMMI Product Suite so that it better meets the needs of organizations worldwide, and realizes the vision shared by all.

The Scope of CMMI for Development

CMMI for Development is a reference model that covers the development and maintenance activities applied to both products and services. Organizations from many industries, including aerospace, banking, computer hardware, software, defense, automobile manufacturing, and telecommunications, use CMMI for Development.

Models in the CMMI for Development constellation contain practices that cover project management, process management, systems engineering, hardware engineering, software engineering, and other supporting processes used in development and maintenance. The CMMI for Development +IPPD model also covers the use of integrated teams for development and maintenance activities (IPPD).

The Group of IPPD Additions

In CMMI, “additions” are used to include material that may be of interest to particular users. For the CMMI for Development constellation, additional material was included to address IPPD.

The IPPD group of additions covers an IPPD approach that includes practices that help organizations achieve the timely collaboration of relevant stakeholders throughout the life of the product to satisfy customers’ needs, expectations, and requirements [DoD 1996]. When using processes that support an IPPD approach, you should integrate these processes with other processes in the organization. To support those using IPPD-related processes, the CMMI for Development constellation allows organizations to optionally select the IPPD group of additions.

When you select CMMI for Development +IPPD, you are selecting the CMMI for Development model plus all the IPPD additions. When you select CMMI for Development, you are selecting the model without the IPPD additions. In the text in Part One of this book, we may use “CMMI for Development” to refer to either of these models, for the sake of brevity.

Resolving Different Approaches of CMMs

The definition of a CMM allows the community to develop models supporting different approaches to process improvement. As long as a model contains the essential elements of effective processes for one or more disciplines and describes an evolutionary improvement path from ad hoc, immature processes to disciplined, mature processes with improved quality and effectiveness, it is considered a CMM. CMMI enables you to approach process improvement and appraisals using two different representations: continuous and staged.

The continuous representation enables an organization to select a process area (or group of process areas) and improve processes related to it. This representation uses capability levels to characterize improvement relative to an individual process area.

The staged representation uses predefined sets of process areas to define an improvement path for an organization. This improvement path is characterized by maturity levels. Each maturity level provides a set of process areas that characterize different organizational behaviors.

Choosing a Representation

If you are new to process improvement and are not familiar with either the staged or the continuous representation, you cannot go wrong if you choose one representation or the other. There are many valid reasons to select either representation.

If you have been using a CMM and you are familiar with a particular representation, we suggest that you continue to use that representation because it will make the transition to CMMI easier. Once you have become completely comfortable with CMMI, you might then decide to use the other representation.

Because each representation has advantages over the other, some organizations use both representations to address particular needs at various times in their improvement programs. In the following sections, we provide the advantages and disadvantages of each representation to help you decide which representation is best for your organization.

Continuous Representation

The continuous representation offers maximum flexibility when using a CMMI model for process improvement. An organization may choose to improve the performance of a single process-related trouble spot, or it can work on several areas that are closely aligned to the organization's business objectives. The continuous representation also allows an organization to improve different processes at different rates. There are some limitations on an organization's choices because of the dependencies among some process areas.

If you know the processes that need to be improved in your organization and you understand the dependencies among the process areas described in CMMI, the continuous representation is a good choice for your organization.

Staged Representation

The staged representation offers a systematic, structured way to approach model-based process improvement one stage at a time. Achieving each stage ensures that an adequate process infrastructure has been laid as a foundation for the next stage.

Process areas are organized by maturity levels that take some of the guesswork out of process improvement. The staged representation prescribes an order for implementing process areas according to maturity levels, which define the improvement path for an organization from the initial level to the optimizing level. Achieving each maturity level ensures that an adequate improvement foundation has been laid for the next maturity level and allows for lasting, incremental improvement.

If you do not know where to start and which processes to choose to improve, the staged representation is a good choice for you. It gives you a specific set of processes to improve at each stage that has been determined through more than a decade of research and experience with process improvement.

Comparison of the Continuous and Staged Representations

Table 1.1 compares the advantages of each representation and may assist you with determining which representation is right for your organization.

Factors in Your Decision

Three categories of factors that may influence your decision when selecting a representation are business, culture, and legacy.

Business Factors

An organization with mature knowledge of its own business objectives is likely to have a strong mapping of its processes to its business objectives. Such an organization may find the continuous representation useful to appraise its processes and in determining how well the organization's processes support and meet its business objectives.

If an organization with a product-line focus decides to improve processes across the entire organization, it might be served best by the staged representation. The staged representation will help an organization select the critical processes to focus on for improvement.

TABLE 1.1 Comparative Advantages of Continuous and Staged Representations

Continuous Representation	*Staged Representation*
Grants explicit freedom to select the order of improvement that best meets the organization's business objectives and mitigates the organization's areas of risk	Enables organizations to have a predefined and proven improvement path
Enables increased visibility of the capability achieved in each individual process area	Focuses on a set of processes that provide an organization with a specific capability that is characterized by each maturity level
Allows improvements of different processes to be performed at different rates	Summarizes process improvement results in a simple form—a single maturity-level number
Reflects a newer approach that does not yet have the data to demonstrate its ties to return on investment	Builds on a relatively long history of use that includes case studies and data that demonstrate return on investment

The same organization may opt to improve processes by product line. In that case, it might select the continuous representation—and a different appraised rating of capability might be achieved for each product line. Both approaches are valid. The most important consideration is which business objectives you would like your process improvement program to support and how these business objectives align with the two representations.

Cultural Factors

Cultural factors to consider when selecting a representation have to do with an organization's capability to deploy a process improvement program. For instance, an organization might select the continuous representation if the corporate culture is process based and experienced in process improvement or has a specific process that needs to be improved quickly. An organization that has little experience in process improvement may choose the staged representation, which provides additional guidance on the order in which changes should occur.

Legacy

If an organization has experience with another model that has a staged representation, it may be wise to continue with the staged representation when using CMMI, especially if it has invested resources and deployed processes across the organization that are associated with a staged representation. The same is true for the continuous representation.

Why Not Both Representations?

Whether used for process improvement or appraisals, both representations are designed to offer essentially equivalent results. Nearly all of the CMMI model content is common to both representations. Therefore, an organization need not select one representation over another.

In fact, an organization may find utility in both representations. It is rare that an organization will implement either representation exactly as prescribed. Organizations that are successful in process improvement often define an improvement plan that focuses on the unique needs of that organization and therefore use the principles of both the staged and the continuous representations.

For example, organizations that select the staged representation and are at maturity level 1 often implement the maturity level 2 process areas but also the Organizational Process Focus process area, which is included at maturity level 3. Another example is an organization that chooses the continuous representation for guiding its internal process improvement effort and then chooses the staged representation to conduct an appraisal.

CMMI and Six Sigma

by Lynn Penn

Capability Maturity Model Integration (CMMI) and Six Sigma are increasingly being discussed in the same conversation. While they do not share a common heritage—one is from the software engineering world and the other is from the manufacturing world—they do share common roots in the principles of Crosby, Deming, et al. Organizations that have endeavored to apply them both typically have managed, budgeted, and resourced the two initiatives separately. This led to not only minimal integration of the initiatives, but also competition between them. Recent research, however, has shown that these initiatives can be jointly leveraged to accelerate implementation of both and accomplishment of mission.[2]

Six Sigma is a holistic approach to business improvement that includes philosophy, performance measurements, improvement frameworks, and a toolkit—all of which are intended to complement and enhance existing engineering, service, and manufacturing processes. Because of its many dimensions, Six Sigma can serve as both an enterprise governance model and a tactical improvement engine.[3]

Six Sigma originated in the manufacturing industry. It was a clear way of identifying acceptable variances around the production of material. It also could be associated with identifying the ability to measure the variance or tolerance around the use of that product or the actual performance of the product itself. Thus, Six Sigma became the quality engine for manufacturing. Although Six Sigma does not guarantee quality, it does provide

2. Siviy, Jeannine; Penn, M. Lynn; and Harper, Erin. Relationships Between CMMI and Six Sigma (CMU/SEI-2005-TN-005). Pittsburgh: Software Engineering Institute, Carnegie Mellon University, December 2005; www.sei.cmu.edu/publications/documents/05.reports/05tn005/05tn005.html.

3. Bergey, J.; et al. Results of SEI Independent Research and Development Projects and Report on Emerging Technologies and Technology Trends (CMU/SEI-2004-TR-018). Pittsburgh: Software Engineering Institute, Carnegie Mellon University, October 2004.

expectations of program performance that can be equated to customer satisfaction, thus implying quality.

CMMI is the integrated approach to process and product development. Over the past several years, numerous Capability Maturity Models (CMMs) have been developed. Software engineering, systems engineering, integrated teams, risk management, and acquisition each had its own model. So industry and government collaborated to establish one model with common terminology, common appraisal methods, and common disciplines.

CMMI, although a collection of multiple models, is most closely associated with the Software and Systems Engineering models. Therefore, it has been adopted primarily by software development organizations. The model is not prescriptive, but is a collection of best practices that, when interpreted in a specific organization, imply a quality product. Like Six Sigma, there is no guarantee of quality, but there is an expectation of product quality as it relates to performance.

There is no question that these two approaches are focused on quality. They are different but not disjointed methodologies. When integrated, these two methodologies can stimulate even more benefits for the organization than if they were used alone.

Let's say an organization has adopted Six Sigma and then decides to adopt CMMI as well. Six Sigma has already established the measurement program, thus satisfying the multiple CMMI generic practices for each CMMI process area associated with measurement and improvement. Six Sigma has also laid the foundation for satisfying the Measurement and Analysis process area. The maturity of the Six Sigma program can also enhance the capability of the organization to adopt CMMI high-maturity process areas such as Quantitative Process Management and Causal Analysis and Resolution.

If an organization first adopts CMMI and then decides to adopt Six Sigma as well, a measurement program already exists. The measurements may be immature, strictly collected, and in some way analyzed, but perhaps not statistically managed. Six Sigma will lead the organization into a high-maturity measurement program. Six Sigma will also be the stimulus for process improvement since it will target variance and where improvements will best benefit the organization, its products, and its customers.

In some cases, organizations simultaneously adopt Six Sigma and CMMI. It has been demonstrated that the rate of recognizing CMMI maturity is clearly enhanced by the adoption of Six Sigma.

Likewise, by using CMMI, the organization can recognize the measurements and process effectiveness that would be of the greatest benefit to assign a tolerance. Realistic control limits or tolerances assist managers in using measurement effectively to manage the project. Without realistic tolerances on measurement, decisions can be erroneous at worst, or hampered at best. The Six Sigma methodologies coupled with CMMI processes allow the organization the capability to focus on the voice of both the customer and the process. Thus, the benefits are recognized both internally and externally. In addition to enhancing the normal rate of level recognition, benefits associated with defect reduction and defect rate detection can be enhanced significantly.

It is very difficult for any organization to spend internal dollars wisely. There is a constant battle for resources. There is also the burden of proof for return on investment. An organization has a short timeline for seeing the benefits of adopting process improvement methodologies. Therefore, an informed decision is critical for the organization. It is valid to assume that two separate process initiatives, disjointed but overlapping, will cost the organization more than two integrated process improvement initiatives. I hope that this perspective starts some organizations with a thought process toward ascertaining that integrated adoption of CMMI and Six Sigma is cost effective for organizations committed to quality. References are provided to enhance the organization's decision analysis and resolution process as it decides how to spend its resources to improve quality.

Your Approach to Process Improvement

To demonstrate how to use this model, let us look at two different scenarios. Scenario 1 is an electronic systems developer that wants to improve its product development processes using a continuous approach. Scenario 2 is a software development company that uses IPPD, has been using the Software CMM, and now wants to use CMMI. This company most recently has been rated at maturity level 3 using the Software CMM (version 1.1).

Scenario 1

In this scenario, you are using a continuous approach and therefore you select the processes that are important to your business objectives.

Since there are 22 process areas to choose from, this is usually too many to focus on when starting out. You may need to narrow your focus. For example, you may find that your competitor always releases its product before yours. You may choose to focus on improving your engineering and project management processes.

Building on this decision, you select all the Engineering process areas as a starting point: Product Integration, Requirements Development, Requirements Management, Technical Solution, Validation, and Verification. You also select Project Planning and Project Monitoring and Control.

You may at this point decide that eight process areas are still too many to focus on initially, and you decide that the requirements process is really where the problems are. Consequently, you select the Requirements Development and Requirements Management process areas to begin your improvement efforts.

Next you decide how much improvement is needed in the requirements area. Do you have any processes in place already? If you do not, your process improvement objective may be to get to capability level 1.

Do you have your requirements development and management processes in place for each project, but they are not managed processes? For example, policies, training, and tools are not implemented to support the processes. If your requirements processes are in place but there is no supporting infrastructure, your process improvement objective may be to get to capability level 2.

Do you have all your requirements development and management processes and their management in place, but each project performs these processes differently? For example, your requirements elicitation process is not performed consistently across the organization. If this is the case, your process improvement objective may be to get to capability level 3.

Do you consistently manage and perform your requirements development and management processes, but do not have an objective way to control and improve these processes? If this is the case, your process improvement objective may be to get to capability level 4.

Do you want to ensure that you are selecting the right subprocesses to improve based on quantitative objectives to maximize your business? If so, your process improvement objective may be to get to capability level 5 for selected processes. In the description of each process area, remember to look for amplifications introduced by the phrases "For Hardware Engineering," "For Systems Engineering," and "For Software Engineering." Use all information that has no specific markings, and the material in the shaded boxes labeled "Continuous Only."

As you can see from this scenario, you need to understand which processes need improvement and how much you want to mature each process. This way of proceeding reflects the fundamental principle behind the continuous representation.

Scenario 2

In the second scenario, you are a software development company using IPPD, using the Software CMM, and you want to use CMMI. You select the process areas at maturity levels 2 and 3 and choose the CMMI for Development +IPPD model.

This selection includes the following seven process areas at maturity level 2: Requirements Management, Project Planning, Project Monitoring and Control, Supplier Agreement Management, Measurement and Analysis, Process and Product Quality Assurance, and Configuration Management. It also includes the following 11 process areas at maturity level 3: Requirements Development, Technical Solution, Product Integration, Verification, Validation, Organizational Process Focus, Organizational Process Definition +IPPD, Organizational Training, Integrated Project Management +IPPD, Risk Management, and Decision Analysis and Resolution. You will also include the IPPD additions.

Since you have already been rated at maturity level 3 for the Software CMM, look at the CMMI process areas that were not in the Software CMM. These process areas include Measurement and Analysis, Requirements Development, Technical Solution, Product Integration, Verification, Validation, Risk Management, and Decision Analysis and Resolution. Determine if you have these processes in your organization even though they were not described in the Software CMM. If any processes in place correspond to these process areas and the other process areas that were in the Software CMM, perform a gap analysis against the goals and practices to make sure you addressed the intent of each CMMI process area.

Remember, in each process area you select, to look for information labeled "For Software Engineering" and "IPPD Addition." Use all information that has no specific markings, as well as the material in boxes labeled "Staged Only."

As you can see, the information provided in this book can be used in a variety of ways, depending on your improvement needs. The overall goal of CMMI is to provide a framework that can share consistent process improvement best practices and approaches, but can be flexible enough to address the rapidly changing needs of the community.