CHAPTER ELEVEN

11 Software Engineering and CMMi

THE INFORMATION TECHNOLOGY (IT) applications discussed in Chapter 10 are based on software programs, some purchased from outside vendors or downloaded from Web sources, others purchased and then customized by the enterprise's IT systems development staff, and still others built by in-house programmers and systems developers. In all instances, planning and good procedures are needed to build and implement successful IT software products. Although the process to build these applications once was called computer programming, it is increasingly known today as software engineering.

This chapter provides a high-level introduction to both software engineering and the Software Engineering Institute's Capability Maturity Model® for Integration (CMMi®), an important assessment tool for evaluating the capabilities of an IT function on many levels. CMMi has strong links to the *C*ontrol *Ob*jectives for *I*nformation and related *T*echnology (CobiT) framework, discussed in Chapter 2, and is useful for evaluating IT quality assurance as discussed in Chapter 31. It is also a measure to allow IT audit to assess how well they are doing in building and implementing the computer-assisted audit tools and techniques (CAATTs) discussed in Chapter 13. An understanding of CMMi and software engineering concepts also will help an IT auditor to better review both IT general and application controls.

SOFTWARE ENGINEERING CONCEPTS

Many IT auditors know a little about computer programming from college classes or from writing simple retrieval programs to gather audit data. While those routines are often fairly simple and easy to launch, most software development projects are much more complex. The enterprise resource planning (ERP) applications discussed

in Chapter 10 on auditing IT applications are examples of very comprehensive IT systems and programs. In order to build and maintain these IT systems and programs, application developers generally need better tools to build, maintain, and operate them.

Software engineering—the application of engineering concepts to software development—is a useful tool. It is the application of systematic, disciplined, and quantifiable approaches to software development, operation, and maintenance. Software engineering is more than just designing and writing computer programs; it often involves several other IT skills and disciplines, including:

- **Software requirements analysis.** This is the process of systems needs analysis, the development of systems and program specifications, and validation of software requirements.
- **Software design.** Software often is designed with specialized modeling and development tools, such as the Unified Modeling Language (UML)[1].
- **Software development.** This is the construction of software through the use of programming languages.
- **Software testing.** Before any software is implemented, it needs to be tested in a thorough and comprehensive manner, often using specialized testing tools.
- **Software maintenance.** Software systems often have problems and need enhancements long after they are first completed. Tools need to be in place to install revisions in an efficient and comprehensive manner.
- **Software configuration management.** Since software systems often are very complex with many related components, their versions and overall configurations have to be managed in a standardized and structured way.
- **Software engineering management.** The management of software systems borrows heavily from project management, as discussed in Chapter 16, but software development processes often include nuances not seen in other project management disciplines.
- **Software development processes.** There is a wide range of formal approaches for designing and building application systems and their supporting systems; two popular process names are called agile and waterfall software development processes.
- **Software quality assessments.** Software engineering includes a variety of tools and approaches to improve the overall quality of all aspects of software development.

Some of the software engineering approaches summarized here are discussed briefly in other chapters. For example, Chapter 10 talks about a system development and requirements process called the systems development life cycle (SDLC), and Chapter 7 reviews the set of information technology infrastructure library (ITIL) best practices processes. However, the typical IT auditor does not need to be an expert in software engineering processes. Rather, he or she should understand these general concepts and to apply them when appropriate when reviewing IT internal control processes.

CMMI: CAPABILITY MATURITY MODEL FOR INTEGRATION

Effective IT development processes and a strong supporting IT function are necessary elements for good software engineering processes. However, over the years, often many IT software development functions were not that effective in building and delivering IT systems that met established requirements following an efficient and on-budget schedule. With frequent late deliveries, chaotic change and revision control processes, and no effective budget controls, IT systems commitments were often consistently missed. Often poor-quality deliveries of the IT systems resulted in too much rework, system functions that did not work correctly, and massive complaints from software product customers—the users—after delivery.

These IT application development problems are nothing new. As far back as the late 1960s, a series of well-publicized major U.S. software development projects failed dramatically for being massively late, not meeting specified requirements, or going way over budget. These events were known as the software crisis; a search on Google or other search engines will provide good background information on that software crisis era.

The U.S. federal government, and of course its taxpayers, were major victims of this software crisis. Over the years, the U.S. government has been a major contractor for a wide number of IT projects, and it generally purchased or arranged for this work from a variety of contractors. However, government customers frequently found that the contractors selected both missed cost or schedule estimates and delivered poor-quality software products. Faced with continuing software vendor performance frustrations over the years, the federal government entered into a contract with the Software Engineering Institute (SEI) of Carnegie Mellon University (CMU; www.seo.cmu.edu) to develop a better approach and process to measure and assess software development groups.

SEI took a design engineering approach to this software development challenge. That is, an engineering team designing a new tool, such as a military weapons system, takes a very structured approach in developing the product. The design steps are tightly controlled and documented with good procedures over such areas as revision control. The well-run design engineering organization will have its processes so well defined and managed that it can easily pick up a new assignment and deliver it in a high-quality manner.

SEI contrasted the well-ordered procedures in many design engineering functions with the almost chaotic processes often found with software development. Initially it developed a model, the Capability Maturity Model (CMM), that allows an organization to assess themselves and for others to determine where they stand in their software development maturity.

That initial CMM was a model for software development organizational improvement. CMM initially defined what is called software process maturity in terms of five distinct levels of development maturity with a series of processes operating at each of these levels. The CMM model starts with what is called Level 1, where processes are unpredictable, poorly controlled, and incomplete. IT auditors have seen this type of maturity in many enterprises where almost nothing is documented, where systems development controls are weak, and where the software development group is operating in some form of near chaos.

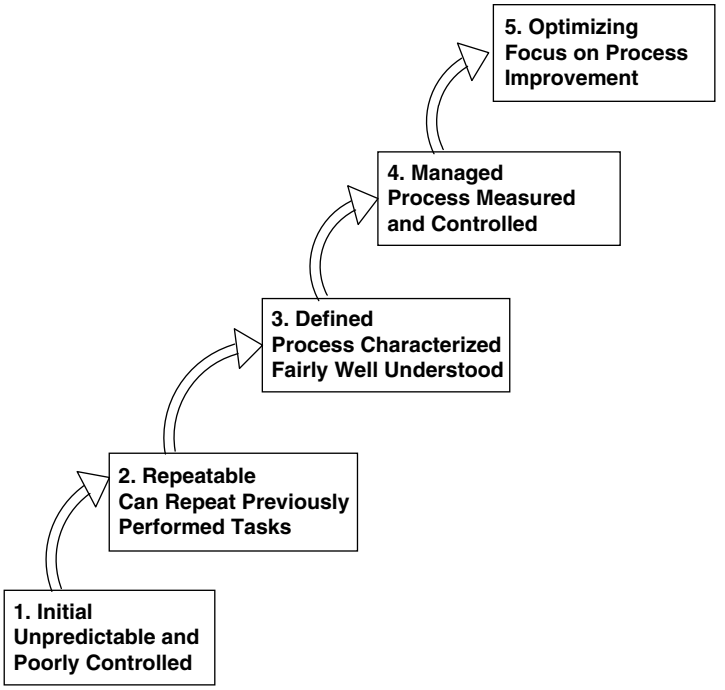

EXHIBIT 11.1 CMMi Maturity Levels Model

The initial CMM said that if a system development organization installs some better key processes, it will improve its operations to move to a next higher level of maturity. That is, an organization can move from the incomplete and unpredictable processes of Level 1 up to Level 2, where processes tend to be repeatable. These five levels of CMM processes are shown in Exhibit 11.1. The sections to come describe each process in greater detail. IT management often is interested in where it stands on these CMM evaluation scales, and internal auditors will find this a useful measure to assess.

CMM is a process model; it can be viewed as the glue that ties people and technology together. It represents a collection of best practices and a framework for organizing and prioritizing activities. Since its initial release, the CMM five-level model has been used by many companies. The SEI's official CMM guidance materials have been updated and expanded to multiple versions for such areas as IT organization services processes (CMMi-SVC) and the software acquisition process (CMMi-ACQ). Today, the model usually is referred to as CMMi to describe its role for IT integration-related processes. This chapter refers to the model as CMMi, although many references in software development literature refer to just the original CMM model.

CMMi is a technical process for improved IT software and other process development. Some IT auditors may ask why it is important. However, CMMi concepts are closely tied to the CobiT framework discussed in Chapter 2, ITIL in Chapter 7, and to other internal audit concepts throughout these chapters. On one level, CMMi is a set of software development processes although many business enterprises today do not do

that much of their own IT systems development work. However, it is also a very good model to measure the overall processes in place in an IT organization. Many IT functions set their own internal goals to move their organizations up through the levels of CMMi maturity. An IT auditor working with IT organizations should develop an understanding of this very important CMMi model.

CMMi Level 1: Incomplete, Unpredictable, and Poorly Controlled Processes

An *incomplete* process is one that either is not performed or is only partially performed. One or more of the specific goals of the process area are not satisfied, and no generic goals exist for this level since there is no reason to institutionalize a partially performed process.

In the early days of IT, virtually all systems development efforts and other procedures were run on an ad hoc, seat-of-the-pants basis with very few formal IT development procedures. In those days of mainframe computing, a data processing manager might run into a key user at the water cooler and informally plan some systems change. Rather than using a formal process for requesting new IT projects, the user might ask data processing for a new system or report. Often little was documented. The data processing manager may have made a quick note regarding the request, assigned it to a programmer or analyst, and never discussed the new project until it was complete. The requested report or system may or may not have met the user's needs; it almost never met the user's completion expectations. In addition, generally there were no development standards in place.

This type of scenario was a characteristic of many early IT functions. There were often no established procedures, very little documentation, minimal revision controls, creating what was often an almost chaotic overall IT environment. This was the era of the mid-1960s going through the 1970s when IT was new with few established standard practices. Seemingly, everyone was trying to accomplish lots of things with limited resources.

Over the years, things got better. For example, IBM published a concept it called the systems development life cycle (SDLC) for building new applications. Although not widely adopted at first, this 1960s era standard procedure became the accepted way for designing and developing many new applications. The SDLC terminology and steps have varied over time, and Chapter 10 discusses this application development process from an IT audit perspective. Other procedures, such as revision control, programming standards, and quality assurance standards, were developed later and became widely accepted as the IT application development profession matured.

Unfortunately, some IT functions have not matured much. Although today's new systems are no longer requested and designed over the water cooler, standards and software revision controls often are not good in some IT organizations. These IT functions lack many, if not most, good operating procedures, and systems are developed and operated in an almost chaotic manner. Using CMMi terminology, these are described as Level 1 organizations with incomplete, unpredictable, and poorly controlled processes. The overall philosophy in this type of IT organization is to ''just do it'' to

produce results rather than going through any level of planning or established development processes.

IT organizations today often like to brag that they are CMMi Level 2, Level 3, or higher. While no one wants to admit their IT function is operating at a CMMi Level 1 chaotic state, an IT auditor can often quickly assess when an IT function is operating at such a Level 1. The function will have few standardized procedures, and those that are documented are often not regularly followed.

CMMi Level 1 is the initial phase of the overall CMMi model. The term *incomplete* is appropriate for an IT development group to start at this initial level and improve to higher levels in the overall CMMi model. Each CMMi level is described from a systems development context in terms of (1) the activities performed by the organization, (2) the activities performed by the IT projects, and (3) the resulting process capabilities of the systems development group. These descriptions better describe an IT systems development group at each level of its relative maturity.

Level 1 Activities Performed by the Systems Development Organization

- The systems development group lacks sound management practices with decisions often made on a day-to-day basis.
- Good software engineering and development practices are undermined by ineffective planning and reaction-driven commitments to deliver requested services.

Level 1 Activities Performed in Systems Development Projects

- During any type of crisis, project leadership tends to abandon any planned procedures.
- The lack of sound systems development management practices defeats even strong software engineering development processes.

Level 1 Resulting Software Process Capabilities

- Software processes are ad hoc with unpredictable results because development processes constantly are changed or modified as the work progresses.
- There are few stable software development processes in evidence.

IT auditors of another age sometimes loved the CMMi Level 1 type of organization because their audit report findings and recommendations were so easy. It did not take much work for an IT auditor to write up a set of findings and recommendations in this environment. Often these were merely repeated from an earlier audit. The auditor knew things were chaotic, IT management knew something had to change, but the proper corrective action procedures were never implemented.

The world today is changing. With much information published about CMMi and its recommended approaches, IT managers and even non-IT senior enterprise managers often ask questions about the quality and maturity of their IT organization. Many enterprises have tried to get around this chaotic, unpredictable IT environment by ending all in-house development and relying on purchased software. This approach,

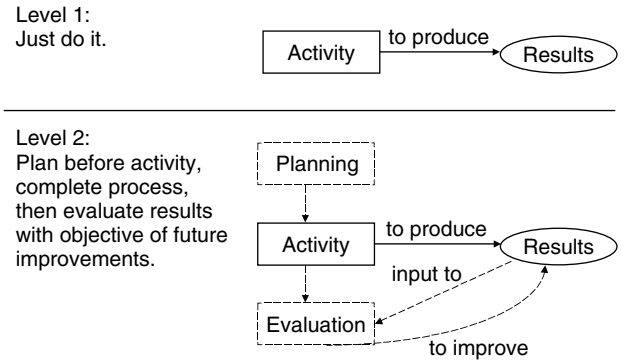

EXHIBIT 11.2 CMMi Level 1 to Level 2 Differences
Source: Robert R. Moeller, *Brink's Modern Internal Auditing,* 6th ed. (Hoboken, NJ: John Wiley & Sons, 2005). Copyright © 2005, John Wiley & Sons. Used with permission of John Wiley & Sons.

however, really does not solve the issue. Purchased software packages can eliminate some problems, but those same packages must be implemented and maintained in an orderly manner. Although many IT groups may never rise much above Level 2 or some aspects of Level 3, most organizations do strive to get above Level 1. Through recommendations and advice, internal audit can help an IT organization to make its processes controlled and predictable.

CMMi Level 2: Repeatable and Consistent Processes

A systems development function should strive to move from the chaotic, unpredictable Level 1 to Level 2. The key defining word for this level is *repeatable*. Rather than doing things in an ad hoc manner, a systems development organization should begin to establish repeatable operating practices. Exhibit 11.2 shows the differences between CMMi Level 1 and Level 2. Rather than just reacting to crises, the systems development organization should devote more attention to planning its activities and then should evaluate results with an objective of process improvement.

Systems development organizations typically do not make an immediate jump from Level 1 to 2 but essentially go through a slow process of adapting various Level 2 processes. The idea with CMMi is to improve systems development processes gradually as an organization becomes more mature. Following the three points raised previously, Level 2 can be described in this way.

Level 2 Activities Performed by the Systems Development Organization

- The systems development organization should establish effective software development policies and procedures.
- Although various specific systems development projects may differ, the systems development function should institutionalize effective project management processes to allow the repetition of successful project management practices developed in earlier projects.

Level 2 Activities Performed in Systems Development Projects

- Realistic IT project commitments should be made based on previous project results and current requirements.
- Processes should be in place to track software costs, schedules, and functionality to identify any problems meeting commitments.
- Control requirements and work products should ensure that project standards are being followed.

Level 2 Resulting Software Process Capabilities

- The software development process capability should be disciplined such that there are stable project planning and tracking processes in place and that earlier successes can be repeated.
- The project management process should be under the effective control of a project management system that follows realistic plans.

Level 2 and beyond requires the effective installation of a series of what CMMi calls key process areas (KPAs). These define the detailed processes necessary in each level and to get to Level 2 and beyond, Level 2's KPAs are:

- CMMi Level 2 KPA requirements management processes
- CMMi Level 2 KPA requirements for software project planning
- CMMi Level 2 KPA requirements for software tracking and oversight
- CMMi Level 2 KPA requirements for software quality assurance
- CMMi Level 2 KPA requirements for software configuration management
- CMMi Level 2 KPA requirements for software subcontract management

CMMi Level 2 KPA Requirements Management Processes

This software development process covers both the technical and customer requirements of software about to be developed and implemented. It is not unusual for an application development group to implement a software application without a full understanding of what the application is supposed to accomplish as well as its overall functional specifications. The CMMi requirements management process calls for IT developers to install formal processes defining the requirements of IT development efforts with these KPA objectives:

- To ensure that the requirements for software products—both new applications being developed and other software tools—are defined and understood
- To establish and maintain agreement on the requirements with all information services requestors: users, customers, and other interested parties
- To ensure that the requirements are met

Requirements should be documented and controlled to establish a basis for software development and project management use. Changes to requirements

should be documented and controlled to ensure that plans, deliverables, and all related activities are consistent with these requirements. Good project management practices are discussed as part of the process description later in this chapter and in Chapter 16.

CMMi Level 2 KPA Requirements for Software Project Planning

In addition to the project management processes discussed in Chapter 16, the KPAs here call for documented estimates of the size, cost, and schedule for use in planning and tracking all software development projects. All affected groups or individuals involved in a software development effort should receive information on commitments and agreements regarding the project. In addition, the project should follow a formal management process for project planning with adequate tracking and status reporting, including measurements for the completion of milestones.

This CMMi project management KPA calls for a much higher level of detailed project planning and tracking than is used by many IT organizations. IT auditors should consider using this KPA as a standard for assessing the progress of IT development when reviewing the management of selected IT projects. Exhibit 11.3 summarizes audit procedures for an IT audit review of IT project management key processes.

CMMi Level 2 KPA Requirements for Software Tracking and Oversight

The purpose of a formal project tracking process is to monitor a project's actual progress against its plan. Monitoring is accomplished by collecting significant information about schedule, resources, costs, features, and quality and comparing this information to the

1. Determine that documented estimates are in place for planning and tracking all IT systems development projects.
2. Processes should be in place for documenting activities and commitments surrounding all significant project activities.
3. All affected groups involved in project development processes should provide commitment agreements regarding their projects.
4. Project planning should follow documented and approved organizational project-planning policies.
5. Management processes should be in place to track the status of all project-planning activities.
6. All project costs and activity schedules actual results should be tracked and compared with estimates in project plans.
7. Processes should be in place to initiate corrective actions when results differ significantly from project plans.
8. Project progress should be regularly tracked against the planned schedules, effort, and budgets.
9. Senior management should review project-tracking status on a regular basis.
10. Ongoing programs of corrective actions, based on lessons learned, should be in place.

EXHIBIT 11.3 KPA Status Review IT Audit Procedures

Source: Robert R. Moeller, *Brink's Modern Internal Auditing*, 6th ed. (Hoboken, NJ: John Wiley & Sons, 2005). Copyright © 2005, John Wiley & Sons. Adapted with permission of John Wiley & Sons.

original and currently approved project plan. The objectives of the CMMi software tracking and oversight KPA are:

- The process should include the information needed to conduct periodic project planning status meetings and reviews.
- Project managers and management should be provided with sufficient information to make data-based business decisions.
- The tracking process should provide information to assist future projects in their estimation and planning efforts.

In many respects, the project tracking process is a by-product of the project management review KPA process.

CMMi Level 2 KPA Requirements for Software Quality Assurance

The purpose of this KPA is to provide management with appropriate visibility into the processes used and the software products being built. This KPA involves reviewing and auditing software products and activities to ensure that they comply with applicable procedures and standards. The objectives of the software quality assurance KPA are:

- All software quality assurance activities should be planned with their plans reviewed and approved by appropriate levels of management.
- Software products and activities should adhere to applicable standards, procedures, and requirements.
- Software quality noncompliance issues that cannot be resolved with a given project should be addressed by senior management.

CMMi Level 2 KPA Requirements for Software Configuration Management

This KPA relates to establishing and maintaining the integrity of all software process products throughout their life cycle. The scope of configuration management involves identifying and controlling configuration items and units. This is similar to the ITIL configuration management process discussed in Chapter 7. In addition, the scope of this KPA includes systematically controlling all changes to active configurations as well as maintaining their integrity and traceability throughout their software life cycles. The goals of this KPA are:

- Software configuration activities should be planned.
- Changes to software work products should be formally identified and controlled.
- Configuration baselines should be established and affected groups and individuals should be informed of their status and content.

CMMi Level 2 KPA Requirements for Software Subcontract Management

The purpose of the subcontract management KPA is to select qualified subcontractors and to manage them effectively. The scope of this KPA includes processes for selecting

appropriate software subcontractors, establishing commitments with those subcontractors, and tracking and reviewing subcontractor performance and results. An IT organization should have the following KPA goals and activities:

- Select only qualified software subcontractors.
- The prime contractor and the subcontractors should formally agree to their commitments and obligations to each other.
- The prime contractor should track actual results and performance against commitments.

CMMi Level 3: Defined and Predictable Processes

As we move up the systems development process improvement chain, CMMi Level 3 calls for am IT organization to use lessons learned to provide inputs to the planning and evaluation processes as well as to improve results. Called the CMMi defined level, this improved systems development level has the following characteristics.

Level 3 Activities Performed by the Systems Development Organization

- Strong documentation should be in place for the organization's standard processes for maintaining and developing software.
- Processes should be in place to integrate project management and software engineering/systems development activities to exploit effective IT system development.
- There should be ongoing support for each of the KPAs within this and other levels, including a training program to ensure skills development.

Level 3 Activities Performed in Systems Development Projects

- Projects should tailor standard software processes to develop an organization's own defined project software processes.
- Because the software process has become well defined, management should have good insights into the technical progress of all projects.

Level 3 Resulting Software Process Capabilities

- This software process capability should be standard and consistent because both software engineering and management activities are stable and repeatable.
- Costs, schedule, and functionality should be under control with the software quality tracked.

Although Level 3 here calls for IT systems development activities such as cost and schedule management, a move from Level 2 to Level 3 is often difficult. CMMi Level 3 calls for a software development organization to achieve a much higher level of organization coordination. Few organizations are able to achieve a CMMi Level 3 that contains the next SEI-defined KPAs.

CMMi Level 3 KPA Requirements for an Organizational Process Focus

This KPA calls for the organization to raise the overall importance of the software development process. All too often, IT auditors are faced with answers like ''I don't know; that was an IT management decision'' when asked why certain application controls are not performing with adequate control considerations. The scope of this KPA involves developing and maintaining an understanding of software development and related project processes throughout the enterprise. To operate at CMMi Level 3, there should be coordination throughout the organization to assess, develop, maintain, and improve these processes.

CMMi Level 3 KPA Requirements for an Organizational Process Definition

Very similar to the process focus of most KPAs, the purpose of this KPA is to integrate project software engineering and management activities that improve software process performance and provide a basis for cumulative, long-term benefits. The goal is to develop and maintain a standard software development processes where the data surrounding those activities are collected, reviewed, and adjusted for ongoing improvements. As a step beyond the normal software development process, attention should be devoted to ongoing improvements.

CMMi Level 3 KPA Requirements for Training Programs and Intergroup Coordination

This KPA calls for an ongoing software development training program to enhance the skills and knowledge of individuals so they can perform their roles effectively and efficiently. It also requires disciplined interaction and coordination of all project and software engineering groups within the organization. This Level 3 training KPA should be planned to support training needs of IT projects, the organization, and individuals. This intergroup coordination KPA calls for customer or end-user requirements to be recognized and agreed to by all groups. All related issues should be identified, tracked, and resolved on an intergroup basis.

CMMi Level 3 KPA Requirements for Peer Reviews

Sometimes difficult to launch, peer reviews involve the methodical and systematic examination of work products by other common or peer-level members of a software development team to identify defects and areas where changes are needed. The goal here is to establish a process that will identify defects in the software development process early and efficiently.

CMMi Level 3 KPA Requirements for Integrated Software Management

This KPA calls for aggressively integrating the organization's software engineering and management activities into coherent and defined software processes, tailored to the organization's software assets. Although defined as a separate CMMi KPA, this really says an organization should give particular attention to software development planning and coordination.

CMMi Level 3 KPA Requirements for Software Product Engineering

The software development process should consistently perform software development activities as a well-defined engineering process that integrates all software development activities in a manner to produce correct, consistent software products effectively and efficiently. Although CMMi is tailored to the consulting groups or software development type of organization that wishes to release a range of quality products to outside customers, these same concepts should exist for internal IT groups. The goal should be to develop and implement high-quality software in a consistent manner.

CMMi Level 4: Managed, Measured, and Controlled Processes

CMMi compliance becomes a very difficult challenge to a software development organization as it moves up to Levels 4 and 5. Level 4 calls for an organization to begin predicting the results needed and creating opportunities to get those results. Exhibit 11.4 describes CMMi Level 4 activities where a software development organization should attempt to predict needed and expected results and then to create opportunities to get those results.

Level 4 Activities Performed by the Systems Development Organization

- The systems development organization should set quality goals for both software products and processes.
- There should be measures of productivity and quality for all important software process activities across all projects as part of an organizational measurement program.
- The systems development organization should provide a foundation for quantitative systems development evaluations.

Level 4 Activities Performed in Systems Development Projects

- Projects should achieve control over their products and processes by narrowing the variation in their process performance to fall within acceptable boundaries.
- The risks of moving up the learning curve if a new application domain are known and carefully managed.

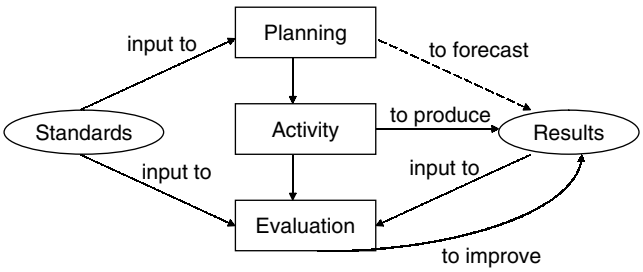

EXHIBIT 11.4 CMMi Level 4 Activities

Source: Robert R. Moeller, *Brink's Modern Internal Auditing*, 6th ed. (Hoboken, NJ: John Wiley & Sons, 2005). Copyright © 2005, John Wiley & Sons. Adapted with permission of John Wiley & Sons.

Level 4 Resulting Software Process Capabilities

- Software process capability should be predictable because the process is measured and operates within measurable limits.
- Allowances should be made for predictive trends in process and quality within quantitative bounds to allow for corrective action when any limits are exceeded.

Two specific KPA areas have been identified at this level: qualitative process management and software quality management. Their emphasis is to integrate software development activities throughout the organization.

CMMi Level 5: Optimizing Processes

Level 5 is the maximum, best practices level of CMMi. It is difficult to implement and achieve. When an enterprise has reached Level 5, it is often a case for a press release. Few enterprises have honestly achieved this level to date. Level 5 calls for an enterprise to install self-correcting mechanisms that are implemented in such a way that activities are improved constantly and continuously. The emphasis is very much on defect prevention throughout the organization and the aggressive and active management of changes.

Level 5 Activities Performed by the Systems Development Organization

- The entire organization should be focused on continuous process improvement with the goal of defect prevention.
- Data should be used for cost-benefit analysis of new technology and new process changes.
- Innovations in systems development and software engineering practices should be transferred to the entire organization.

Level 5 Activities Performed in Systems Development Projects

- Project teams should analyze defects and determine their causes.
- Project teams should evaluate processes to prevent known types of defects from recurring.

Level 5 Resulting Software Process Capabilities

- Software process capability should be improving continuously because the organization improves the range of capability and process performance of its software development projects.
- Improvements should occur both by incremental advancement of existing processes and by innovations using new technologies and methods.

CMMI BENEFITS

CMMi had its origins as a tool for improving systems development processes. This chapter has described the original IT systems development CMMi model and its levels

and KPAs. Many enterprises today are not involved in heavy software development initiatives, and the other CMMi models for services or acquisition may be more appropriate. No matter what the overall enterprise objectives, however, the CMMi model provides a place to start improving processes as an enterprise moves from the unstructured Level 1 to an environment of more refined processes. It is a framework to help an enterprise develop a better shared vision, a common language, and a way to define process improvement throughout an organization.

Although CMMi got its start with government contract software developers, it has been implemented by many IT development organizations today. CMMi's roots are very much tied to the American Society for Quality quality assurance standards discussed in Chapter 31. The SEI maintains CMMi standards and publishes guidance and training materials and process-related standards.

As an example of CMMi potential benefits, this author became heavily involved with CMMi several years ago at a large U.S. insurance company where he was working on a contract basis as a project manager. The project involved a large systems implementation involving some new development work as well as extensive modification to existing systems. The company had published extensive IT procedures, but because of a very casual management approach, those procedures often were not followed. For example, this author found breakdowns in such relatively simple but important tasks as software revision control. Two programmers, for example, sometimes attempted to make changes to copies of the same program source code version. The result, of course, would be two incompatible software versions.

Although there are other ways to remedy such problems, this author suggested that the insurance company adopt CMMi processes. The effort began with some internal seminars to the IT staff on the CMMi process improvement model, explaining its benefits and outlining that the organization was in a Level 1 chaotic state of operations. With lots of training and guidance, we were able to bring the operating unit from an unstructured Level 1 to a solid Level 2 with some Level 3 KPAs completed during the author's project period.

IT AUDIT, INTERNAL CONTROL, AND CMMI

The CMMi model describes a process for improved quality software development for organizations. Although some aspects of CMMi and supporting published materials often seem tailored to the software development organization developing products for external customers, with an emphasis on major government contractors, the core CMMi model is an effective way to think about how an organization is performing and what steps it should take to improve processes. Even though a development group may not be directly focused on CMMi, an IT auditor can use this model to evaluate current performance and to recommend areas for future improvements.

What does an organization need to do to claim that it is operating at some CMMi level? The chapter describes some of the specific and important KPAs required to make any such claim. However, an organization must go through some extensive process improvement along with supporting documentation to assert that it is operating at a

specific CMMi level. An organization should contract with certified registrars to review their processes and certify their CMMi level. This is particularly important if an organization needs to document its status for contract purposes.

The typical IT auditor will not have the knowledge to appraise whether the IT organization being audited is operating at CMMi Level 2 or 3, or whether it has installed and implemented the correct mix of KPAs. However, an IT auditor should have a general understanding of CMMi, as described in this chapter, and be able to ask appropriate questions if the organization reviewed is operating with CMMi processes.

Perhaps most important, when an IT auditor reviews an IT development group with very poor internal controls that are really operating at a CMMi Level 1, he or she might recommend that the IT group look at CMMi as a method to improve processes and develop better internal controls. The effective implementation of CMMi is a good way for an organization to improve its internal controls. Although most organizations will never reach CMMi Level 5, each level points to important processes for improved internal controls.

NOTE

1. Unified Modeling Language (UML) is a standardized general-purpose modeling language in the field of software engineering. UML includes a set of graphic notation techniques to create visual models of software-intensive systems. For more information, consult www.uml-forum.com