Training Guidelines: Creating a Training Plan for a Software Organization

Maribeth B. Carpenter

Harvey K. Hallman

September 1995

Training Guidelines: Creating a Training Plan for a Software Organization

Maribeth Carpenter
Harvey Hallman

Community Sector

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Table of Contents

List of Tables

Acknowledgments

Training Guidelines: Creating a Training Plan for a Software Organization

The key process areas at Level 3 address both project and organizational issues, as the organization establishes an infrastructure that institutionalizes effective software engineering and management processes across all projects. ... The purpose of the Training Program is to develop the skills and knowledge of individuals so they can perform their roles effectively and efficiently. Training is an organizational responsibility, but the software projects are responsible for identifying their needed skills and providing the necessary training when the project's needs are unique. [1]

Abstract: This set of training guidelines focuses on issues to be addressed by the training program of a software organization comprised of multiple software projects. While much of the content of the guidelines is equally applicable to training plans for individual projects, this document presumes a coordinated function providing training across software projects.

---

[1]   Paulk, M.C.; Weber, C.V.; Curtis, B.; & Chrissis, M.B. *The Capability Maturity Model for Software: Guidelines for Improving the Software Process.* Reading, Mass.:Addison-Wesley Publishing Company, 1995, pp. 35-36.

---

1   Introduction

Training is one of the most important aspects of improving a software organization. Adequate training is necessary to make investments in improvement profitable and to institutionalize improved practices. Both Capability Maturity Model[SM] (CMM[SM])[2] and the People CMM (P-CMM),[3] developed by the Software Engineering Institute (SEI), contain key process areas (KPAs) addressing training. In both models, these training efforts become focused upon the entire organization at maturity level 3. Training has to be planned properly for long range effectiveness. Without an adequate training plan, funds may be wasted or improperly spent. There are many things to be considered when creating a training plan. This document provides an organization with guidance in preparing an organizational training plan for the first time.

The creation of a training plan is one element within the training process of a software organization. To show that activity in context, Table 1 illustrates an abstraction of the training process with the ETVX paradigm,[4][5] which was first used to document the IBM programming process architecture in the early 1980s. The model has four components:

- (E) entry criteria
- (T) tasks
- (V) validation tasks or criteria
- (X) exit criteria

[2.]   The Capability Maturity Model and the CMM are service marks of Carnegie Mellon University.

[3.]   Curtis, Bill; Hefley, William E.; & Miller, Sally. *People Capability Maturity Model* (CMU/SEI-95-MM-02). Pittsburgh, Pa.: Software Engineering Institute, Carnegie Mellon University, 1995.

[4.]   Radice, R.A.; Roth, N.K.; O'Hara, Jr., A.C.; & Ciafella, W. A. "A Programming Process Architecture." *IBM Systems Journal, vol 24, no 2,* (1985):79-90.

[5.]   Radice, Ronald A. & Phillips, Richard W. *Software Engineering: An Industrial Approach. vol 1.* Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1988.

Table 1: Training Process ETVX

Entry Criteria	Tasks	Exit Criteria
	Conduct training needs analysis	
	Create training plan	
	Design curriculum	
Management support	Create training products	Needed training delivered
Training policy and objectives	Pilot and deliver training	Training objectives met
Resources	Evaluate training	
Organizational context	**Validations**	
	Training plan approved	
	Course development and delivery processes followed	
	Quality standards met	
	Training results analyzed and reported	

The entry criteria are those conditions that must be present prior to beginning the training process. The training function needs management support and adequate resources to perform its tasks; it needs to have an articulated training policy and articulated objectives, and to have a crisp definition of its role and scope of concern within the organization.

The tasks are the essential activities of the process. The guidelines in this document focus on the task of creating a training plan. Other tasks in the training process are discussed in lesser detail but can't be ignored because the training plan documents their outputs and describes procedures and standards used in conducting them.

The training process validations insure that the output of the tasks meet required standards. The exit criteria define the output state the training process produces.

The audiences for these guidelines are

- The training planners within the training group of a software organization.
- The manager of the training group.
- Software Engineering Process Group (SEPG) members who plan training for the organization.

The guidelines presume that readers have the knowledge and experience qualifying them for

the task of planning software engineering training. These characteristics are listed under "Resources" in the guidelines section "Entry Criteria for Creating a Training Plan for a Software Organization." The guidelines help the readers to focus their abilities on the specific task of creating an effective training plan for their software organization.

The guidelines first discuss the context of the CMM Training Program KPA in terms of its placement in the maturity grid, what an organizational training plan is, and what states and conditions must exist to begin an effective software training program at the organizational level.

The guidelines then describe the suggested content of an organizational training plan, including ways to represent software engineering curricula and tactical approaches to implementing the training program, e.g., defining course development and acquisition processes, projecting resource needs, elaborating procedures for student selection and enrollment, setting standards for course delivery, and tracking and evaluating the training.

Most organizations have some aspects of the suggested training plan content documented; few if any have all aspects documented. To illustrate training plan components, the guidelines contain adaptations of selected training plan materials from real organizations, e.g., Motorola, Lockheed Martin Tactical Aircraft Systems, PRC, and Jet Propulsion Laboratory. The reader can tailor the formats presented, synthesize ideas from the software engineering training community, and construct a robust organizational training plan.

2 Why Is the Training Program Key Process Area (KPA) at Level 3?

A cursory reading of the CMM shows that training is an issue that is addressed at all process maturity levels. There has to be training to support improvement at all levels, so why doesn't the notion of a training program show up until maturity level 3?

The key is that at level 3 process improvement efforts are focused, defined, and integrated. Earlier efforts to progress to level 2 have been accomplished. At level 3 improvement efforts are coordinated and focused at the organizational level, and are no longer a loose collection of bottom-up improvement efforts. Besides the Training Program KPA, other KPAs at level 3 are: Organizational Process Focus, Organizational Process Definition, Integrated Software Management, Software Product Engineering, Intergroup Coordination, and Peer Reviews. The first goal of Organization Process Focus is, "Software process development and improvement activities are coordinated across the organization."[6]

From a training perspective, at level 3 there is an organizational focus on training. While project-driven training needs are emerging in a bottom-up fashion, the organizational training group is also concerned with identifying broader organizational training needs and making the provision of project-driven training more efficient when those needs are common among projects.

As with the CMM, the P-CMM brings an organizational focus to training at level 3. The P-CMM has four KPAs that specifically address training issues: one at level 2 and three at level 3. The P-CMM Training KPA resides at level 2 and describes the training program for the unit or project. At level 3 the Knowledge and Skills Analysis KPA focuses on the identification of the core competencies of the organization and the knowledge and skills required to perform them. The Workforce Planning KPA describes developing a strategic workforce plan that sets organization-wide objectives for competency development. The Competency Development KPA addresses establishing training and other development programs in each of the organization's core competencies.

A well-known example of an organization that has made continuous learning an organization-wide focus is Motorola.

> *Motorola's worldwide education and training community is a federation - a linkage of all Motorola training organizations across all regions called Motorola University. Through collaborative, joint venturing, the partners in this federation develop the experience and expertise to enhance corporate competence by:*
> *- Providing visibility to major skill/job shifts and emerging markets,*
> *- Leveraging educational assets - human, technical, physical and sharing resources,*

---

6. Paulk, M.C.; Weber, C.V.; Curtis, B.; & Chrissis, M.B. *The Capability Maturity Model for Software: Guidelines for Improving the Software Process*. Reading, Mass.:Addison-Wesley Publishing Company, 1995, p. 194.

- Improving learning efficiency, eliminating redundancy, and reducing training costs
- Reducing the cycle time of training needs assessment and course development,
- Transferring knowledge through common training platforms.[7]

An organizational focus on training may sound simple in concept but in practice some complex questions can arise.

- What are the relative roles of an organizational training group and training groups at the division and project level? (See "Scope of the Training Plan.")

- How are the total curriculum needs of software practitioners and managers divided between the organizational training group and the division or project training group? Who performs training needs analysis, curriculum development, training development, and delivery? Are the topics themselves clearly split or does one group develop and maintain the training materials and another group tailor them for specific audiences? (See "Scope of the Training Plan.")

- When organizational policy states that certain training is required, are waivers permitted? (See "Student Selection and Enrollment Procedures.")

- How do people and financial resources flow among training groups? Is some training funded at the organizational level and other training at the division level? Are the divisions "taxed" to support their use of corporate resources or does corporate provide funding for cross-divisional training? Does the organizational-level training group have to pay for the time of people who participate in organizational-level training needs analysis? (See Table 3 and "Estimated Training Costs.")

- How do the various training groups report their activities? What information about training is consolidated at the organizational level? How is the effectiveness of the training program measured and reported? (See Training Evaluation and Tracking Procedures.)

A well-defined organizational training plan addresses these and many other important issues.

[7.] *Motorola University: Catalyst for Change Through Continuous Learning,* Motorola, Schaumburg, Illinois.

3   What Is an Organizational Training Plan?

The term "training plan" is overloaded. It means very different things to different people. If you ask to see a training plan, you may get a number of different things (including blank stares).

- the charter for the software engineering training program
- a list of training needs which have been identified by a project or directly requested by management or the trainee
- the process by which training needs will be verified and prioritized for implementation, by which curricula and courses will be designed, by which the effectiveness of training will be measured
- a candidate list of courses with plans for either internal development or acquisition
- the procedures for developing or acquiring a course, including a defined process, standards for course materials, and quality measures
- the curriculum offered, its medium of delivery, and the schedule of offerings
- the required and optional training available for someone with a particular software-related role
- the plan for developing a particular course, showing task assignments and the development, review, and piloting schedule
- the plan for satisfying the training needs of specific software practitioners during the coming year, showing what training they will get when

While all of these are valuable, none singly is adequate to assure that people are being effectively trained to perform their management and technical roles. Most organizations have several of the above items; few organizations have documented an overall, integrated strategy for software engineering training.

An organizational training plan documents the objectives of the training program, the training needs of the organization, the training to be delivered, and tactical procedures for carrying out training activities. It differs in scope from project or individual training plans in that it looks at how training can serve the strategic interests of the software organization as a whole.

> *As a resource of change within Motorola, Motorola University works best when business objectives are clear. To ensure that the organization is well versed in the issues affecting the company, Motorola University regularly seeks guidance and input from its board[8] of trustees and the Motorola University Congress. The board of trustees is composed of senior managers from each of Motorola's groups and sectors, corporate staff members, and two Chief Executive Office members. The board formulates training policy, sets priorities, and allocates resources based on the corporation's business strategies. The Motorola University Congress ... ensures that the training policies and direction of the Board of Trustees are implemented.*

---

8. *This Is Motorola University,* Motorola, Schaumburg, Illinois.

---

4   Entry Criteria for Creating a Training Plan for a Software Organization

The existence of an organizational training plan does not imply achievement of the CMM level 3 Training Program KPA goals. The creation of an organizational training plan is a step toward achieving KPA goals but embraces only a subset of necessary activities. Creating an organizational training plan addresses the first goal of the Training Program KPA: "Training activities are planned."[9] The other two goals require that needed training be provided to software management, those in software technical roles, and to individuals in software-related groups. The training plan merely defines the context and the definition of some of the activities that will actually accomplish those goals.

When do you know that you are ready to start developing an organizational training plan? The Training Program KPA common features of *commitment to perform* and *ability to perform* provide guidance.

> ### Commitment to Perform
>
> *Commitment 1: The organization follows a written policy for meeting its training needs.*
>
> ### Ability to Perform
>
> *Ability 1: A group responsible for fulfilling the training needs of the organization exists.*
>
> *Ability 2: Adequate resources and funding are provided for implementing the training program.*
>
> *Ability 3: Members of the training group have the necessary skills and knowledge to perform their training activities.*
>
> *Ability 4: Software managers receive orientation on the training program.[10]*

This section discusses four inputs to training planning. The KPA excerpts highlight two important prerequisites for creating an organizational training plan.

1. management policy and support
2. resources

Other desirable inputs include

3. project training plans
4. results of an organizational training needs analysis

---

9. Paulk, M.C.; Weber, C.V.; Curtis, B.; & Chrissis, M.B. *The Capability Maturity Model for Software: Guidelines for Improving the Software Process.* Reading, Mass.:Addison-Wesley Publishing Company, 1995, p. 214.

10. *Ibid*, pp. 214-217.

4.1 Management Policy and Support

CMM Training Program KPA commitment 1 requires a written policy for meeting training needs. The policy may be written by a management group, the SEPG, human resources, or the training group. The policy must be approved by the official policy sanctioning group within the organization. This policy authorizes the training group's work and should be in place before you start developing the organizational training plan.

Many organizations keep a notebook of active policies. Some organizations do not have a single training policy but instead have training policy statements integrated into a broad set of other policy statements. For example, human resource policy may require that all job classifications identify specific skills needed for that classification, and management policies might require that training effectiveness and training needs be assessed annually. Some organizations use policy to recognize and sanction various forms of knowledge transfer, e.g., classroom training, workshops, conferences, "brown bag" or lunchtime seminars, mentoring and apprentice programs. Policy can also clarify the organization's position on the private (individual) or proprietary (corporate) aspects of training records and training program assets.

Some organizations write training policy specifically for their software engineering population. The following is the policy on software engineering training from PRC Federal Systems Group.

Table 2: PRC Federal Systems Group Software Engineering Training Policy

FSG 503 Software Engineering Training

Policy

1. Training shall be conducted to build the skill base within FSG, to fill the specific needs of the projects, and to develop the skills of individuals. Each organization developing or maintaining software products shall conduct a formal training program for all project staff. A formal program is characterized by the following:
   - Annual identification of training requirements
   - A documented training plan
   - Maintenance of training records for each staff member
   - Annual evaluation of training effectiveness

2. The needed skills and knowledge for each of the following software management and technical roles shall be identified and reviewed on a regular basis by SEPGs at all levels of the organization.

   Software practitioner
   Software team leader
   Software project manager
   Project management specialist
   Configuration management specialist
   Quality assurance specialist
   SEPG member

3. SEPGs at all levels of the organization shall on a continuous basis evaluate the effectiveness of the training program and take action to ensure that the training program meets the corporate objectives.

The CMM Training Program KPA ability 4 indicates that software managers throughout the organization are made aware of training policy and plans, as well as their role in supporting the training program. Obtaining management support can never start too soon. You will need management participation to create the content of the training plan and to reach management consensus. Look for ways to make the emerging training plan visible and well accepted by promoting training, not on its own merits, but as a necessary part of achieving organizational goals. The senior management needs to articulate and demonstrate a commitment to the work of the training group, specifically to

- Provide needed training to ensure that individuals possess the requisite skills and knowledge to perform their current software-related roles effectively and efficiently, to prepare the workforce for future core competencies of the organization, and to contribute to the achievement of organization-wide strategic goals. This commitment is embodied by the organizational training program, which is developed and executed by an identified group of people who are funded to perform the software training function.

- Support training needs analysis. A software development project can not be adequately planned without a statement of requirements, neither can you plan training without a training needs analysis. Training needs analysis requires commitment of resources to participate in and carry out the analysis. Some perspectives to be considered when performing training needs analysis are

  - role-based needs. The goal is to identify the skills and knowledge needed for each software management and technical role. This approach to identifying training needs is strictly job related. An individual within a certain job classification should know "x" and be able to perform "y" within acceptable limits of quality and productivity.

  - individual- or project-specific needs. The goal is to identify training needs specific to individuals or specific to projects. This approach to identifying training needs is based upon individual career growth needs and tactical project needs.

  - strategic needs. The goal is to identify training needed to build personnel capabilities of strategic importance to the organization. Examples of strategies requiring training support are

    - achieving corporate goals for quality, productivity, process maturity, and predictability (such as delivering software at one tenth the current defect density in 3 years)

    - gaining market share based upon exemplary products and services targeted to a particular market segment (new market differentiation), such as developing software for client/server environments

    - becoming the first to market with products based on an emerging technology, such as the use of online services for secure financial transactions

- developing an appropriate mix of software specialists to support the overall work. (See Appendix A for data from a Capers Jones' study on technical specialists in software organizations.)

- Support training development. Internal resources will be expended to develop needed training or external products will be procured and potentially tailored.

- Support training delivery. Physical vehicles for training may be classroom facilities, training centers for individual study, and instructional products. Supporting procedures requiring funds may be formal software engineering apprenticeship and mentoring programs, tracking of planned and required training and associated waivers, and applying policy for charging student time to participate in learning activities.

Texas Instruments demonstrates management commitment to training by supporting a robust annual training planning process. The following is adapted from a description of that process:

> *Our leadership recognizes that Texas Instruments employees are key to a sustained competitive advantage that wins in the marketplace. Our people strategy places a high priority on employee development and education. Each employee has a minimum annual learning goal of 40 hours, which is one of four key business metrics. Our development into a learning organization is supported by many programs. These include a just-in-time[11] learning strategy, assessment and career development processes, a redesigned educational assistance process, Job Enhancement Program, multimedia-based learning, and distance learning processes.*

The Texas Instruments Learning Institute's (TILI) calendar runs from January through December. The training planning process begins in August, when the financial cycle begins and all strategic training needs as well as program-specific training needs have been identified (See "Project Training Plans" for the Texas Instruments program-specific training identification process). The Defense Systems and Electronic Group's Leadership Team determines what the budget and training metric hours will be for the following year. During this time frame, individual training requirements are being collected from various disciplines' model committees. For example, the software engineering discipline receives its training recommendations from the Engineering Education Team (EET), which is the governing body of engineering training. The EET charters a sub-team to develop, through voluntary mentoring discussions, each individual employee's discipline-specific training plan, focusing on career development that enhances the employee's ability to remain technically vital to Texas Instruments. The individual training plans, showing mandatory, recommended, and optional training, are loaded into a database accessible by the employee and by his or her manager.

---

[11]. Fortin, Pauline D.; Jeske, Christine A.; Lakey, John R.; Urquhart, Kristin B.; & Vea, Anthony. "Is This Training? A Unique Approach to Software Process Training in Industry," pp. 409-417. *Proceedings of the 8th SEI Conference on Software Engineering Education*. New Orleans, Louisiana, March/April 1995. Berlin, Germany: Springer-Verlag, 1995. This paper describes the just-in-time approach to software process training used at Texas Instruments.

4.2 Resources

The Training Program KPA abilities 1, 2, and 3 refer to needed resources, which include facilities, funding, and staff. The creation of the training plan is a people-intensive task. The organizational training group team is key to the production of a well-conceived training plan and its effective implementation. Creating the training plans requires a diversity of skills and abilities, including in-depth knowledge of the organization and its strategic direction. While it may not be necessary to have the entire organizational training group staffed and on board when creating the training plan, the team will need expertise in:

- software engineering subject matter expertise
- strategic and tactical planning
- budgeting
- training needs analysis for both skills and higher cognitive abilities
- instructional design and development
- media design
- written and oral communication, particularly with management-level people
- synthesis and analysis
- expertise in organizational behavior

Patrick L. Doran, superintendent of the Training Division at U.S. Transportation Command at Scott Air Force Base, suggests a list of 10 professional competencies for the training profession.[12]

1. Knowledge of learning theory; motivational psychology, instructional systems design; delivery-method success rates, limitations and alternatives; and emerging technology applications to the training industry.

2. Expertise in training needs analysis.

3. Ability to speak in public and to make professional presentations to executives.

4. Ability to use and modify automated training systems, to assess and acquire training systems, and to manage information about training resources.

5. Understand different theories of training organization; know the pros and cons of decentralized and centralized training delivery plans.

6. Exercise superb management and leadership skills.

7. Ability to analyze budget and financial results, including value-added concepts and returns on investment.

8. Ability to conduct strategic training analysis and design.

[12.] Doran, Patrick L. "Training Is a Terrible Thing To Waste." *Training* (July 1995): 11-12.

9. Understand how to use critiques and post-training measures to integrate quality into training.

10. Understand corporate environments and how training is "postured" within them.

4.3   Project Training Plans

Project training plans, as described in *Activity 1* below, will exist for each software project in a CMM level-3 compliant organization.

Activity 1: Each software project develops and maintains a training plan that specifies its training needs. The plan covers:

1. The set of skills needed and when those skills are needed.

2. The skills for which training is required and the skills that will be obtained via other vehicles.

3. The training that is required, for whom it is required, and when it is required.

4. How training will be provided.[13]

An organizational training plan uses the project training plans as input. Project training plans are likely to cover project-driven needs (software languages, techniques, and tools), individual career development needs, and training needs associated with the particular technical (databases, GUI, artificial intelligence) or application (manufacturing, telecommunications) domain of the project. The training needs identified in the project training plans provide a basis for finding common training needs across the projects. The project training plans also represent a significant training needs analysis effort that you do not need to repeat, although you may need to clarify and verify the content of the plans.

Within the Texas Instruments Defense Systems and Electronics Group, a training specialist is assigned to each program at start-up. The training specialist follows a defined training needs identification 19-step process[14] designed to provide program-specific and timely training. Steps within the process include

- Identifying training needs for staff involved in the proposal phase.

- Assessing unique skill requirements of the program, both specified and anticipated.

- Identifying training needs for staff involved in the program execution phase.

[13.] Paulk, M.C.; Weber, C.V.; Curtis, B.; & Chrissis, M.B. *The Capability Maturity Model for Software: Guidelines for Improving the Software Process.* Reading, Mass.:Addison-Wesley Publishing Company, 1995, p. 217.

[14.] Brockette, Ann Harrington & Gibbons, Mary Frances, eds. "Program Start-Up Training Process." *Texas Instruments Learning Institute Catalog,* Dallas, Texas: Defense Systems and Electronics Group, Texas Instruments,1995.

- Locating sources of training.
- Preparing the training plan and training budget. Training is a non-negotiable budget item.

Some organizations have created computer-based tools to assist managers in collecting and prioritizing training needs for individuals and for projects. The priority of the training need may be based upon the time frame in which the training is needed, upon the criticality of the training to project success, or a combination of factors. Having the training needs in a uniform online format allows for rolling up the needs to an organizational level, where the needs can be analyzed and tracked globally. These automated tools offer encouragement to managers to create individual and project training plans and facilitate the analysis of training needs at the organizational level.

- The Knowledge and Skills Assessment Tool (KSAT) is used by the Naval Surface Warfare Center in Dahlgren, Virginia. The SEPG commissioned a training needs assessment that was conducted by the Oak Ridge Associated Universities, Oak Ridge Institute for Science and Engineering. The training needs analysis results were used to populate a database with job requirement profiles for specific tasks. For each task, the profile provides the required knowledge and skills, with associated proficiency codes, and sources of training in those knowledge and skills. Individuals are assigned job functions comprised of tasks. A manager can use the tool to develop an individual training plan and roll up the individual training plans into a project plan. The tool runs on a Novell network on a PC with a minimum of 33 Mhz and a 386 processor. KSAT is DOS-based and programmed in CLIPPER from Computer Associates International Inc. The tool is intended to facilitate the development of individual and project training plans tailored to specific task-based performance needs.

- An online tool is used by PRC Inc. to collect individual and project training needs and to identify employees who are eligible for waivers from required training. The tool is based on Excel spreadsheet templates. PRC has a required curricula for individuals in specific job roles within projects at specific CMM maturity levels (see Table 8 for curricula requirements). For each employee, the manager can identify which courses are required and for which courses the requirement can be waived. The manager also can specify courses that are required for project needs and courses that are of interest to the individual. At the organizational level, these needs are rolled up, analyzed, and tracked. From these project needs, PRC can develop an organizational training plan specifying what courses will be developed and when they will be delivered.

4.4 Results of an Organizational Training Needs Analysis

The organizational training group will need to conduct a training needs analysis for those areas not well covered in the project training plans. Projects without training plans will require in-depth needs analysis to capture project-driven and special career development needs. Areas for the organizational analysis to focus on might include

- Undocumented project needs, such as the overlooked need to train the technical writers in the design notation used to describe the software product.

- Training for career path progression that is missing from individual training plans.

- Training needs associated with organizational improvement initiatives, such as understanding the principles of the CMM, which is being used as the basis for a software process improvement initiative.

- Training associated with new directives on corporate processes, such as the integration of software life-cycle models into an overall corporate product development process.

- Training on global issues related to the organization's business domain, such as understanding what features clients value in products for which the organization supplies embedded software.

- Cultural change initiatives, such as a restructuring of software personnel into teams of specialists or teams of generalists.

- Skills needed to achieve a new organizational desired state, which may include the need to grow new organizational capabilities or core competencies, such as the use of Cleanroom software development methods to reduce the number of faults experienced by customers.

The general strategy for conducting a training needs analysis is to

- Identify, through general knowledge of the organization, its goals, and its barriers to achieving those goals, the high leverage areas to investigate for performance improvement.

- Choose a method of data collection and analysis (see Appendix B). When designing a training needs analysis approach, collect and analyze data that is calibrated with the intensity of training needed, using a scale such as Bloom's Taxonomy as described in Appendix F. This calibration is essential to the specification of the training to be later developed or purchased.

- Collect the data.

- Analyze the data and synthesize training recommendations.

There are several methods (called study types) for collecting and analyzing data. No single method is adequate for the complex discipline of software engineering. It is best to use a combination of methods, including at least one method that yields objective data. The simplest of the study types described in Appendix B is the needs survey, which involves asking software practitioners and their managers what training is needed. The use of focus groups may provide a more complete and objective view of the needs, but this study type is likely to be biased by short-term perspectives.

Problem analysis and task analysis are most applicable to jobs where the performance is directly observable and to tasks that can be described with great specificity. There are software-related tasks that fall within this category, but they usually also contain significant elements of judgment. For example, one can describe in detail how to record errors identified in a software design inspection, but the validity of the data recorded is ultimately based upon the judgment of the inspection team in identifying, categorizing, and accurately describing software anomalies.

As with most engineering tasks, software engineering activities are predominantly judgment- and competency-based rather than procedure based. It is not possible to define a sequential list of observable steps that lead to the successful completion of a typical software engineering task, as it is for a procedure-based task. Software engineering tasks require the exercise of judgment and a degree of autonomy in deciding how a task will be accomplished. A task for a software engineer may be to design the software components of a system. The task requires a broad scope of knowledge about the system, the application, software design principles, implementation constraints, etc. Likewise, a broad base of skills is needed to represent the design in precise notation, to communicate the design to appropriate stakeholders, and to validate the design against the requirements. Overarching the knowledge and skills is the application of judgment to arrive at an optimal design.The outcome of a software engineering task isn't known *a priori,* which makes training needs analysis for software engineers more difficult than for more structured disciplines. The most complex study type, performance analysis, may be the most appropriate for software engineering jobs. One technique is to observe and interview the best performers to ascertain what knowledge, skills, and abilities they are calling into play.

Training needs analysis is the most critical part of the training process. Performing it well is both an art and a science. The training group would be wise to invest in their own professional development to hone their skills in this area. Research is needed on how to perform training needs analysis for the profession of software engineering. There are opportunities for reporting experiences with training needs analysis in software engineering organizations at the annual Conference on Software Engineering Education, which is currently co-sponsored by the SEI and IEEE. There are also books[15] [16] that contain practical advice.

Following the training needs analysis, the organizational training group must approach the complex task of designing curricula to meet the prioritized list of identified needs. It is important to recognize that the training needs will not express themselves in neat packages that easily identify courses or curricula. More analysis will be needed to cluster the requirements into groups that suggest the course material and order of presentation. Furthermore, determining

---

[15]. Zemke, Ron & Kramlinger, Thomas.*Figuring Things Out: A Trainer's Guide to Needs and Task Analysis.* Reading, Mass.: Addison-Wesley Publishing Company, Inc., 1982. This book emphasizes techniques.

[16]. Swanson, Richard A. *Analysis for Improving Performance: Tools for Diagnosing Organizations and Documenting Workplace Expertise.* San Francisco, Calif.: Berrett-Koehler Publishers, 1994. This book emphasizes linking training to organizational goals and taking a systems approach to performance analysis.

---

what is appropriate for the organizational training group to deliver requires the consensus of the affected organizational entities. In most organizations this consensus will not be easily obtained, as there are many stakeholders with conflicting priorities, agendas, and reward systems.

5   Content of an Organizational Training Plan

The purpose of the organizational training plan is to document decisions and to gain consensus on the direction of training for the software organization. Stakeholders must review the plan. Senior management must approve the plan and track its execution. The training group must reexamine the plan at least annually to keep it current with organizational policy, strategic direction, and needs.

With this purpose in mind, the format of the plan is of secondary importance. Select its form of existence (paper, electronic) to maximize its availability to all stakeholders. The training plan must be both public and dynamic.

The following information needs to be included in the training plan.

1. Scope of the Training Plan

2. Responsibility for the Plan

3. Training Objectives

4. Technical Strengths and Weaknesses of the Software Organization

5. Software Engineering Curriculum

6. Course Development and Acquisition Process

7. Estimated Training Costs

8. Student Selection and Enrollment Procedures

9. Course Delivery Standards

10. Training Evaluation and Tracking Procedures

How the content is allocated to sections of the training plan is a design decision of its author, however, a sample outline is shown in Appendix C. The following sections are meant to highlight content areas of the plan. If only parts of the proposed content are available at first writing, you can add material to the plan later.

5.1   Scope of the Training Plan

The scope of the training plan is the organizational entities, the personnel categories, and the topic and performance areas covered by the training plan.

The training plan explicitly describes the software organization covered by the training program. The program may apply to an entire company, a single division of an organization, all the parts of an organization within a specified geographic area, or those parts of the organization under the management of a specified reporting structure. However the organization is described, the delimiters must be as discrete as possible. Ambiguity will lead to confusion. The plan may eventually be ignored if its coverage is not understood.

The training plan states which categories of personnel are covered by the training program. The program, for example, may cover only those exempt categories of personnel in software-related roles and those supervisors within two levels of direct management of people in software-related roles. If there are circumstances in which personnel outside the scope of this training program are permitted to participate, these circumstances should be stated. Stating which personnel roles are included is important in scoping training needs analysis activities and funding issues.

The training plan identifies which performance areas are addressed by the training program. For example, the training program may cover knowledge and skills associated with software development and maintenance practices, but may exclude knowledge and skills associated with particular application domains, such as avionics or medicine.

In the organizational training plan, reference other relevant training plans and describe their relationship to the organizational training plan. Other training plans include those that are

- applicable to suborganizations, such as projects
- directed toward external audiences, such as customers or suppliers
- directed to more narrowly scoped audience segments, such as only those projects working in medical applications
- applicable to parent organizations, such as corporate management training programs

5.2 Responsibility for the Plan

The persons responsible for the plan must be identified. Responsibility is typically shared by a number of organizational entities including

- the managerial owner of the training initiative or improvement initiative who is ultimately responsible for an effective training program
- the organizational unit responsible for creating and maintaining the training plan document
- the stakeholders responsible for providing input to the training plan and for supporting its implementation
- the contact point for suggested improvements to the training program

The CMM states that "Activity 2: The organization's training plan is developed and revised according to a documented procedure."[17] That procedure must be documented or referenced within the training plan and must designate who is responsible for creating, updating, reviewing, approving, managing, and controlling it.

[17.] Paulk, M.C.; Weber, C.V.; Curtis, B.; & Chrissis, M.B. *The Capability Maturity Model for Software: Guidelines for Improving the Software Process*. Reading, Mass.:Addison-Wesley Publishing Company, 1995, p. 218.

5.3 Training Objectives

The objectives of the training program are goals that will be achieved and benefits that the organization will accrue as a result of training program activities. Training objectives are distinct from learning objectives, which describe goals for the changed behavior of participants in learning activities. Training activities encompass more than providing learning opportunities, i.e., performing training needs analysis and communicating the results across the organization, designing curricula and gaining consensus on the priorities the curricula represent, defining the core competencies of the organization, and evaluating the training program as a contributor to organizational goals. All of these activities affect the communication channels within the organization, the culture of the organization, and the image the organization portrays. Training objectives must state how the training program will affect these broad issues.

In writing training objectives, reference the organizational training policy and the organizational goals the training program supports. These may be stated in other documents or may have to be reiterated as part of the training plan. Some organizations map the course learning objectives to organizational objectives by stating in each course description which business objectives the course supports. See Table 9 for an example of a course description that includes strategic and tactical goals as well as learning objectives. It is important to relate individual training activities to the objectives of the organization. In this way, the training program assumes the objective of supporting certain organizational objectives in specified ways.

Training policy and organizational goals often imply constraints as well as success measurements for the training program. The training plan must make these constraints and measurements as explicit as possible. For example, a training policy may state that personnel covered by this plan will receive a minimum or a maximum of 40 hours of training per year. Organizational goals might indicate a desire to win contracts requiring CMM level 3 capability or that 60% of mandatory training will be completed by the end of year. Such statements have direct effect upon the breadth of any training needs analysis activity, the scope of any curriculum designed, and the rigor applied to training needs prioritization. When an organization has clearly stated organizational goals, training results must be evaluated with respect to the contribution of training to the achievement of those goals. Organizational policy and goals must be an explicit part of the rationalization of training program activities.

Few organizations do an adequate job of relating the training activities to strategic and visible goals. There are far more good tactical training plans describing how a curriculum will be implemented than there are good strategic training plans explaining how the training program has a vital role in the organization's success. Attention to the strategic goals provides more leverage than attention to the tactical approaches. The organizational training plan must provide the rationale for selecting the method of determining the training needs and designing the curriculum, showing how these activities are driven by the organization's goals.

A very effective way of showing the benefits of the training program is to call out explicitly the consequences of not providing training. What market opportunities will be missed? What will happen if there is no improvement in software quality or productivity? How many contracts will be lost if the organization does not have the requisite software process maturity or skill in an emerging software technology? What is the effect on productivity when individuals are allowed to "figure it out on their own" or subject matter experts solve others' problems individually on an interrupt basis? What is the cost of errors attributable to lack of training? In essence, what is the cost of not training?

5.4 Technical Strengths and Weaknesses of the Software Organization

The training plan highlights the results of a training needs analysis that focused on identifying the gaps between the knowledge and skills needed for the organization to achieve its goals and the current state. The methods used to conduct the training needs analysis are described.

Indications of technical strengths and weaknesses are found through

- Classical training needs analysis procedures (see Appendix B).
- Knowledge and skills analysis and workforce planning activities (see P-CMM level 3 KPAs).
- Software process maturity assessments.
- Software risk assessments.
- Training needs suggested by the CMM.
- Future state analyses.
  - change in business direction dictates need to develop technical expertise the organization currently doesn't have, e.g., building client/server applications
  - desire to develop a capability needed to introduce a new technology, e.g., a formal specification language, highlights training needs
  - a new process paradigm to improve the organization's performance, e.g., shorter build cycles, requires that current processes be adapted to the new vision

CACI International, Inc.[18] derives its training needs from analysis of the following sources of requirements:

- Standards - CMM, ISO-9000, and various other standards with which the company has chosen to comply.
- Internal policy statements - human resources and management policies that require the identification of skill sets for various job classifications.
- Contract or project requirements - specific skills that must be acquired and maintained during contract or project performance.

---

[18]. Discussion with Gary Coleman of CACI on August 20, 1995.

- Other plans - business and technology strategic plans that contain explicit and implicit information relating to training needs and approaches.

Some organizations use information about the career tracks of specific individuals to determine training needs and to prepare individuals for more responsible roles.

Since all of the identified training needs can't be addressed immediately, organizations often form training working groups to prioritize the training needs and create consensus across the organization on high-leverage training interventions. Curriculum designers participate in the prioritization process and bring focus to the curriculum design issues.

5.5 Software Engineering Curriculum

The training plan lists and describes the training opportunities to be provided. Throughout this section, the term "course" means a unit of learning, e.g., a classroom experience, a self study, a mentoring session, or any other form of learning activity.

Good software designers know that no single design notation can represent all relevant design perspectives, e.g., functional, behavioral, informational. Multiple representations are usually required to express all critical design decisions. Similarly, there are several kinds of information about the software engineering curriculum that can be conveyed in the training plan.

- content descriptions for the courses
- intended audience for the course
- recommended sequencing of the courses for someone performing a specified role within the organization
- length of the course and its delivery medium
- funding sources for course development and delivery
- designation of mandatory and recommended courses
- planned frequency of course offerings or the course schedule
- course development and retirement schedule

There are many ways to represent this information. Different representations highlight different aspects of the curriculum. The choice of representation depends upon the complexity of the information. If there is little variability in the information, i.e., all courses are funded in the same way, use the same delivery media, or are intended for the same audience, simple text will do. When curriculum components have a number of variable characteristics, more complex representations are needed. Select the representation of your curriculum information based upon its complexity. Matrices are a popular and highly readable way to summarize curriculum information. Following are some examples of curricula representations used by real organizations.

Table 3 represents planning done by a Raytheon training working group in the early 1990s and conveys information on planned courses, their audience, course length, how many times courses will be offered during the year, what part of the organization will fund course development and teaching, what part of the organization will fund the students' time, and which courses are yet to be developed (boxed). For courses not yet developed, any interim solutions, such as purchasing seats in vendor courses, need to be described.

Table 3: Raytheon Curriculum Plan Excerpt[a]

Course	Coverage	Duration (Hours)	1991 Iterations	Preparation & Instructor Funding Source	Attendance Funding Source
- - -	- - -	- - -	- - -	- - -	- - -
Design & Code Inspections	Senior Engineers	14	6	- - -	- - -
Software Testing	Engineers	24	2	- - -	- - -
Software Project Management	Development Managers	32	1	- - -	- - -
- - -	- - -	- - -	- - -	- - -	- - -

a. A presentation by the Training Working Group of Raytheon, Equipment Division, Software Systems Laboratory, 1991.

Lockheed Martin Tactical Aircraft Systems (LMTAS) uses a three-tiered training architecture that is updated and maintained by the Software Engineering Process Group (SEPG) to represent their process improvement curriculum. Each tier has an associated funding model.

Table 4: Lockheed Martin Tactical Aircraft Systems (LMTAS) Process Curriculum Architecture[a]

Process Awareness Modules	• Orientation • Process Fundamentals
Project-Specific Process Training Modules	• Phase Checklists • Phase Overview Briefings • Practices • Related Group Roles • Document Templates
Software Technical Supporting Knowledge & Skill Training Modules	• External Regulations / Internal Policies • External / Internal Organizations • Resources / References • Tools • Technical / Professional Skills • Interpersonal / Managerial Skills

a. Data provided by Michele A. Nimerick of Lockheed Martin Tactical Aircraft Systems, Fort Worth, Texas.

LMTAS provides a matrix of mandatory (m) and recommended (r) courses for job roles (See Table 5). Annual goals were set for the percentage of mandatory training completed by the population. An individual may perform more than one role. Each role has an associated curriculum path that supplements the curriculum data with the length of each module as it is tailored to the role (x pages, y hours) and the media types involved (job aid, workbook, video and workbook, classroom). In the training plan for each individual, planned and completed dates are shown for each learning module.

Table 5: Lockheed Martin Tactical Aircraft Systems (LMTAS) Courses by Job Roles

TRAINING COURSE	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
Software Related Personnel:																		
Executive Management	m							r	r							r	r	
Middle Management	m							r	r							r	r	
Software Management:																		
Project Leader	m	r	r	r	m			m	m	r				r		m	m	
Chief (1st Line SW Manager)	m	r	m	r	r	m		m	m					r		m	m	
SEPG	m	m	m	r	r		r	m	m	r		r			m	m	m	
SQA	m	r	r	m			m	m	m	m						m	m	
SCM (Company Level)	m	r	r	r			r	m	m	r						m	m	
Software Planner/Estimator	m	r	r		r							m						
Training Facilitator	m										m							
Software Development:																		
Software Product Manager	m	m	m	m	r		r	m	m	m		r	m	m		m	m	
Software Product Acquisition Manager	m	r	r		m							r	m	m				m
Review Leaders	m	r	r	m			m	r	r	m			r			m	m	
Review Participants	m	r		m			m	r	r	r			r			m		
SCM (Project Level)	m							r	m	m			m			m		
System Engineer	m	r						m	m	m			r			m	m	
Software Engineer:																		
Design	m	r		r				m	m	m			m	r		m	m	
Code	m	r		r				m	m	m			m	r		m	m	
Test	m	r		r				m	m	m			m	r		m	m	
System Test Engineer	m	r		r				m	m	m			r	r		m	m	

A: Overview of the Software Development Process
B: Process Fundamentals
C: Managing Software Development
D: Software Product Evaluations
E: Project/Program Management
F: Chief Training
G: Inspections
H: Software Quality Assurance (SQA)
I: Software Configuration Management (SCM)
J: Software Corrective Action System/ Software Problem/Anomaly Reports
K: Train the Trainer
L: Software Estimating
M: Software Requirements Management
N: Risk Management
O: Software Engineering Process Group
P: Working as a Team
Q: Software Engineering Disciplines
R: Subcontractor Management

Motorola University represents its software engineering and core curriculum by listing the courses under the categories of fundamental, languages, CMM and Quality System Review (QSR), systems engineering, process and technology, and management. Regional offices of Motorola University draw upon, tailor, and augment the core curriculum to serve their specific business needs.

Table 6: Motorola University Software Engineering Core Curriculum[a]

Course Name
Fundamental
Developing Quality Software
Personally Shaping the Software Solution
Executive 5-Ups Seminar
Languages
Advanced C
C++ for C Language Programmers
C Language Fundamentals
SEI Capability Maturity Model and Quality System Review
Introduction to the SEI CMM
SEI CMM Advanced Concepts
Assessor Training for Software QSR, Motorola's Quality System Review
Systems Engineering
Systems Engineering Process
Systems Requirements Engineering
Process and Technology
Structured Methods
Structured Methods for Database Applications
Software Reviews
Software Technology Planning for Practitioners
Implementing Continuous Improvement
Implementing Software Configuration Management
Implementing Software Sizing and Estimation

Table 6: Motorola University Software Engineering Core Curriculum[a]

Course Name
Process and Technology (continued)
Software Process Framework for Small Projects
Gathering Requirements
Effective Code Reading for C
Object-Oriented Analysis Using Object-Modeling Technique
Object-Oriented Design Using Object-Modeling Technique
Implementing Software Continuous Improvement
Software Metrics for Process Improvement
Software Reviews for Information Systems
Software Unit and Integration Testing
Software System Testing
Management
Structured Methods Management Overview
Managing the Software Development Process
Software Management Seminar
Understanding Continuous Improvement for Managers
Understanding Software Sizing and Estimation for Managers
Understanding Software Continuous Improvement for Managers
Executive Workshop on Software Competency
Software Management Seminar

a. Adapted from Software Engineering and Core Curriculum, Motorola University, Schaumburg. Illinois, 1995.

Some organizations may wish to represent a software process improvement curriculum within the framework of the IDEAL[SM 19] model. The IDEAL model represents the phases of a software process improvement initiative, which are iteratively applied for each process improvement cycle: initiating, diagnosing, establishing, acting, and leveraging. The advantage of using the IDEAL model for curriculum mapping is that the training activity is closely tied to strategic goals of software process improvement and to the other highly visible work of the SEPG. As each software process improvement cycle is related to a time frame, courses described within

---

[19.] IDEAL is a service mark of Carnegie Mellon University.

the model context are also time sequenced, helping the training group to plan resources based upon when the training will be needed. Table 7 shows a sample curriculum by job category and IDEAL phase for one cycle through the IDEAL model. Each course name is annotated by two parameters: (number of hours, Mandatory (M) / Recommended (R)). "IDEAL Cycle 1" may be interpreted to mean the first cycle through the IDEAL model, such as a group might follow to achieve the goals of CMM level 2.

Table 7: Software Process Improvement (SPI) Training: IDEAL Cycle 1

Job Category	Initiating Phase	Diagnosing Phase	Establishing Phase	Acting Phase	Leveraging Phase
Executives	SPI Overview (4,M)				
Mgmt Steering Group	SPI Overview (8,M) Managing Technological Change (8,M)	Introduction to the CMM (32,R)	Software Process Definition (8,R)		
SEPG	SPI Overview (8,M) Managing Technological Change (40, M) Consulting Skills Workshop (32,R)	Introduction to the CMM (32, M) Appraisal Process (40,R) Software Measurement (24,R)	Software Process Definition (24,R)		
Appraisal Team		Introduction to the CMM(32,M) Appraisal Process (40,M)			

Table 7: Software Process Improvement (SPI) Training: IDEAL Cycle 1

Job Category	Initiating Phase	Diagnosing Phase	Establishing Phase	Acting Phase	Leveraging Phase
Senior, Middle, 1st-Line Managers			Managing SPI (8,M) Software Process Definition (8,R)	Software Project Management (40,M/R) Software Quality Assurance (8,R) Software Configuration Management (8,R) Software Estimation (16,M/R)	
Process Action Team Leaders			SPI Overview (8,M) Software Process Definition (24,M) Software Process Measurement (24,M)		
Process Action Team Members			SPI Overview (8,M) Software Process Definition (24,R) Software Process Measurement (24,R)		

Table 7: Software Process Improvement (SPI) Training: IDEAL Cycle 1

Job Category	Initiating Phase	Diagnosing Phase	Establishing Phase	Acting Phase	Leveraging Phase
Software Practitioners		Introduction to the CMM(32,R)	SPI Overview (8,R) Software Process Definition (8,R)	Software Quality Assurance (8,R) Software Configuration Management (8,R) Software Estimation (16,R) Inspections (4,M)	

PRC Inc. has integrated CMM-related training needs and other organizational training needs into a curriculum of required courses, mapped to CMM maturity level and software roles (see Appendix D). This required curriculum is shown in Table 8.

Table 8: PRC Required Core Courses Per Software Role

CMM Level	Course	A	B	C	D	E	F	G	H
2	Introduction to Software Process Improvement	X	X	X	X	X	X	X	X
2	Project Management and Tracking	X	X	X			X	X	X
2	Requirements Management	X		X	X	X	X	X	X
2	Software Quality Assurance	X		X	X		X	X	X
2	Software Configuration Management	X		X	X	X	X	X	X
2	Sizing and Estimating of Software and Computing Resources	X	X	X	X			X	X
2	Project-Specific Orientation	X	X	X	X	X	X	X	X
2	Quality Improvement Awareness	X	X	X	X	X	X	X	X
2	Managing Quality Improvement	X		X				X	X
2	Team Leader	X		X				X	X
2	Process Management	X	X	X	X	X	X	X	X
3	Engineering of Software	X		X	X	X	X	X	X
3	Peer Review	X		X	X		X	X	X
3	Introduction to Software Process Assessments	X		X				X	X
3	Software Engineering Process Group							X	X
3	Instructional Techniques								X
3	Principles of Team Building	X		X				X	X
4	Quantitative Process Management	X	X	X	X	X	X	X	X
4	Software Quality Management	X	X	X	X	X	X	X	X
5	Defect Prevention	X		X	X		X	X	X
5	Technology Change Management	X						X	X

A: Project Manager
B: Project Management Specialist
C: Software Team Leader
D: Software Practitioner

E: CM Specialist
F: QA Specialist
G: SEPG Member
H: Training Group Manager

Each course within the curriculum should be described, possibly within the training plan, but more likely as appendices or separate documents referenced within the training plan. A course description may serve as a specification for a course that is to be developed or acquired, or as documentation of an existing course. Jet Propulsion Laboratory in Pasadena, California, uses a course description format that is illustrative of the elements of a good course description (See Table 9).

Table 9: JPL Course Description Format[a]

Course Title: Software Engineering Methods for Requirements and Design Quality
Description: {Explain what "methods for requirements and design quality" are. Explain how the course fits into the context of other NASA /JPL quality improvement efforts.}
Purpose: Strategic: The purpose of this training is to improve the quality of JPL software.   Tactical: The strategic purpose implies the following objectives: • Create a core of JPL software engineers with consistent requirements and design skills.   • Improve engineering practices during the early phases of software development.   • Provide the prerequisite knowledge for the successful introduction of Computer-Assisted Software Engineering (CASE) tools.   • Enable the production of software with a longer useful life through increased maintainability.   • Make the development of quality software a repeatable and predictable process.
Course Objectives: The student upon exiting the course will be able to   General Knowledge   • List ...   • Show ...   • Distinguish ...   Requirements Skills   • Describe ...   • Create ...   • Write ...   Design Skills   • Design ...   • Perform ...   • Identify ...

Table 9: JPL Course Description Format[a]

Prerequisite Knowledge: {Describe the knowledge, skills, experience, and abilities the students are expected to possess prior to the course.}
Fundamental Assumptions: • The software engineering methods presented will be compatible with JPL standards. • The methods will be supported by "commercial off-the-shelf" CASE tools. • The methods taught will be based upon techniques proven successful through use on government and industry projects. • The methods presented will not become mandatory for JPL projects. • Two courses will be produced: a short course for managers, a longer detailed course for software engineers.
Uncertainties: {State risks in terms of staff resources, content volatility, tool acquisition, etc.}
Course Format: {Describe course length, daily schedule, media for delivery, exercises included, homework, etc.}
Students: {Describe intended audience in terms of job roles, home departments, etc.}
Training Materials: {List all materials participants will receive (slide copies, job aids, exercises, copies of articles, textbooks, bibliography, glossary of terms) and all materials that will be presented (slides, videos, demos).}
Course Outline: {Show major topics in the order they will be covered, mapping each to a time allocation and the course objectives addressed.}

a. Data provided by John C. Kelly of Jet Propulsion Laboratory in Pasadena, California.

Add to the above format a section on measuring the effectiveness of the course. Try to look at effectiveness from several aspects (see also "Training Evaluation and Tracking"):

- Was the learning experience satisfactory? This feedback typically comes from the students in the form of class evaluation forms. One can learn whether there are barriers to learning that can be eliminated from the environment, whether there is a good match between student expectations of the course and what is being offered, whether presentation pacing is too fast or too slow, etc.

- Did the students learn anything new? Some organizations use pre-tests and post-tests to measure the improvement in knowledge, skills, and abilities. Did all the students leave the course with at least a minimal level of performance capability? Learning objectives can be stated in terms of Bloom's taxonomy (see Appendix F) and goals can be set for building the organization's capability, e.g., that all people should be at the knowledge level, 80% at the comprehension level, 50% at the application level, 20% at the analysis level, and 5% at the synthesis level.

- Did the organization experience improved performance? Objective observers, such as managers and those in the software quality assurance function, could be asked to evaluate improvement with respect to the items listed as tactical goals in the JPL example.

- Did the organization make progress toward meeting its strategic goals? Executive management should report improvements related to the strategic purpose that the training supports.

Course descriptions may also include information on whether a refresher course is needed after a specific interval or whether a student must "requalify" after some time.

5.6 Course Development and Acquisition Process

Here is a rhetorical question for the reader. Answer true or false: *The development or acquisition of instructional materials should be governed by the same level of management rigor as a software development or acquisition effort of equal monetary magnitude?* One underlying supposition of the question is that software development and acquisition projects are adequately managed, an assumption that itself poses an interesting question. Another supposition is that monetary magnitude is the appropriate basis for comparison.

The similarity between the development cycle of software and the development cycle of instructional materials is often cited. For both the decision whether to build, buy, or tailor is paramount. For both the development phases include requirements, design, implementation, testing, and maintenance, with appropriate points of product review by a designated review body. The process should also include the management aspects of size and effort estimation, requirements management, schedule and cost tracking, product demand analysis, and quality and customer satisfaction tracking.

As with software, the development and acquisition processes for training must be documented and followed. Table 10 shows a sample list of activities to include in a process for developing instructional materials. Each activity needs to be placed in the context of an overall process flow and to be elaborated using the ETVX paradigm (entry criteria, task descriptions, validation procedure, exit criteria) or similar model.

The Defense Systems and Electronics Group of Texas Instruments has developed an excellent *Instructional Technology Design Guide*[20] outlining the training process in 6 phases: customer focus, requirements, design, development, implementation, and operation and control. The document is replete with discussions of methods and illustrations of checklists and forms. Its intent is not to force a rigorous process upon instructional specialists, but to remind the experts of options and alternatives and to shorten the learning curve of novices.

---

[20]. Finley, Kenneth W. Jr. *Instructional Technology Design Guide,* Dallas, Texas: Defense Systems and Electronics Group, Texas Instruments, 1992.

When acquiring instructional materials, most of the "Determining Requirements" activities shown in Table 10 still apply. Instead of focusing on product development, focus on matching existing products to the requirements gathered. Estimate acquisition and delivery costs. Before committing to large-scale acquisitions, involve technical experts in a dry run or dress rehearsal and representatives of the intended audience in a test offering.

Table 10: Instructional System Development Activities[a]

Activities
Determine Requirements • Analyze needs (needs collection, needs selection, training solution selection). • Determine audience (job level, education, experience, prerequisite knowledge). • Set educational objectives and relate them to organizational objectives. • Select media and materials (lecture slides, videotape, audiotape, exercises, textbook, job aids, tools, case studies, workbook, mentoring tools for on-the-job training, glossary, handbook). • Consult subject matter experts on content and topics to include and on instructor qualifications. • Identify constraints (resources, date required, delivery environment). • Sketch development process (activities, resources). • Identify marketing/delivery strategy (student recruitment, frequency of offerings, pricing, instructor resource, marketing collateral). • Prepare financial estimates for development, delivery, and maintenance.
Create Design • Build architectural design to map training solution components to educational objectives, delivery medium, and means of learning assessment. • Build detailed design for each component: the content outline, learning objectives, learning assessment mechanism, bibliography, associated readings and exercises. • Involve technical experts in content issues through dry runs and dress rehearsals. • Determine packaging of instructional materials. • Plan development process with milestones and assigned responsibilities. • Update cost estimates based upon more detailed design. • Develop incremental test plan to include dry runs, pilot offering, choice of participants, evaluation mechanism, and method of incorporating feedback.

Table 10: Instructional System Development Activities[a]

Activities
Build Product   • Implement design by creating planned instructional materials and marketing collateral.   • Conduct a test offering with participants representative of the intended audience.   • Evaluate the test offering by summarizing key issues and discrepancies and by assessing student learning.   • Respond to test evaluation by changing the materials, the planned delivery approach, or the marketing collateral, as appropriate.   • Review packaging of the course materials for consistent quality, general impression, and usefulness to the student.   • Update financials based upon actual development and materials costs.

Revise Product[b]
 • Redesign product. Triggers for redesign might be:
 	
      • change in learning objectives
      • content obsolescence
      • consistent student comments
      • change in delivery media or time allocation
      • change in instructor qualifications
 	
 • Get input from technical experts.
 	
 • Rebuild product and update the design documents and marketing collateral.
 	
 • Conduct a test offering with participants representative of the intended audience.
 	
 • Respond to test evaluation by changing the materials, the planned delivery approach, or the marketing collateral, as appropriate.
 	
 • Review packaging of the course materials for consistent quality, general impression, and usefulness to the student.

a. Adapted from work at the Software Engineering Institute by the Education and Training Review Board and the Education Process Project Quality Improvement Process Team, 1994.

b. Consider retiring the product instead of revising it.

5.7 Estimated Training Costs

The training plan indicates how resources are to be allocated against planned activities. These activities include performing training needs analysis, administering the training program, developing and acquiring training, delivering training, and evaluating the effectiveness of training. A rule of thumb is that it takes 20 to 40 working days to collect the data for a training needs analysis,[21] plus the time required to report the findings. Administrative and evaluation costs

are ongoing and can be accounted for as a straight-line budget item. The rest of this section will focus on the costs of the training.

There are three types of training costs that need to be considered when creating a training plan.

1. Direct costs. These are all of the costs directly associated with the development, acquisition, and delivery of the training.

2. Indirect costs. These are the costs incurred by having the trainees inactive on their projects during the training.

3. Negative costs. These are the costs that would be incurred if necessary training were not delivered. Negative costs are also a form of indirect costs but need to be categorized separately.

5.7.1 Direct Costs

Direct costs include the costs of creating or purchasing training plus the costs incurred in delivering the training. When a course is purchased and used as is, these costs can be obtained from the vendor. When a course is to be developed or tailored, a number of factors contribute to the cost, including the length of the course, the compensation level of the developer of the training, the compensation level of the deliverer of the training, and the selected delivery media.

For training development, the pivotal question is, "How much effort is required to create a course?" If your organization has an algorithm based on experience, use it. An algorithm used by several course development organizations at IBM in the early 1980s was that it took 50 hours to develop each hour of a stand-up presentation when the presentation was to be given by the developer. Add more time if the presenter and the developer are different people, as the presenter has a learning curve to adapt to the material and to adapt the material to his or her teaching style. It may be possible, if the training is to be given only one time, to subtract time because the content can be tailored to a known audience and teaching artifacts associated with multiple deliveries, such as detailed instructor notes and elaborate visuals, don't need to be developed. As much as 50% additional effort may be necessary to develop computer-aided training. Complex media such as CD-ROM requires 100-200 hours for each hour of material. More accurate algorithms are obtainable by collecting your own metrics on the training development process and then using that data to estimate future development efforts.

A way to reduce the direct cost of training is to purchase off-site training for one person who has teaching skills and is technically competent in the subject area. That person can then teach other people. In all such circumstances, care must be taken that the legal rights, e.g., copyright and intellectual property rights, of the original course developer are not violated. Use of a vendor's training materials generally requires a formal licensing agreement. This ap-

---

21. Zemke, Ron & Kramlinger, Thomas. *Figuring Things Out: A Trainer's Guide to Needs and Task Analysis.* Reading, Massachusetts: Addison-Wesley Publishing Company, Inc., 1982, p. 14.

proach may work well with courses on the use of software tools or systems where there are manuals available to document the way the software or hardware works; it is less effective when the topic of the training is less tangible. For example, a course on the use of a spreadsheet tool can easily be taught by someone who has been trained in its use, but a course on formal methods of software verification would be difficult to provide in this manner. An alternative for more complex training may be to contract with the developer of the course to train one of the organization's employees to teach the subject matter to the organization.

Direct costs include

- The time of the instructors.
- The costs of facilities, e.g., the classroom, computers, and audiovisual equipment.
- The costs of student materials, e.g., notebooks, textbooks, and course completion certificates.
- The costs of consumable items,e.g., flipchart paper, pens, and catering.
- The time of administrative personnel to process enrollments, set up the classroom, and handle pop-up problems during the class.

Direct costs also include the cost of getting the recipient to the training or the training to the trainee. If training is purchased, the tuition and travel costs are obvious. Direct expense can also be incurred when training is presented internally if the employee is sent to the training location or the instructor and the training materials come to the location of the employee. These travel and shipping costs are direct and will result in additional money being spent by the organization.

5.7.2   Indirect Costs

Indirect costs are costs incurred by having the trainees inactive on their projects during the training. Part of this indirect cost is the time of the student. Another indirect cost, the productivity of other project members, is often negatively affected by having the trainee unavailable during the training. Occasionally, indirect costs can be minimized by training all of the people who need to work together at the same time. This reduces the inefficiencies of having the trainee unavailable to support the project. Indirect costs can be the most significant cost of the training, e.g.,1 hour of training that is to be given 5 times a year with a class size of 20 people will cost approximately 55 person hours of direct development and delivery costs (50 development hours per hour plus 5 delivery hours) and more than 100 person hours of indirect student time costs (20 people multiplied by 5 classes, plus lost productivity of others).

Indirect costs will vary depending on the training delivery method. Stand-up delivery in a classroom is the easiest to calculate but results in the highest indirect costs.

"Finger-tip" learning[22] has the lowest indirect costs. When the software practitioner needs to have a better understanding of some particular subject, the training is brought up on the computer and is performed immediately. There is no interruption of the individual's work environment. Some of today's systems that are based on multimedia presentation techniques provide this capability[23] [24]. In the future, "finger-tip" learning will become a viable alternative to classroom training. The direct development costs may be greater, but the indirect costs can be minimized and even become invisible. "Finger-tip" learning is accomplished by an individual on an as required basis.

5.7.3 Negative Costs

Negative costs occur as a result of not providing the training. They are best characterized as lost opportunity costs. Unrealized productivity gains could perhaps have been obtained with specific software training, e.g., training on modern spreadsheet programs that help software developers use the tools as more than online adding machines and take advantage of the various functions that provide more powerful analysis tools. Without training across the organization, a knowledgeable software developer on the project will inevitably become either the official or unofficial expert and will incur a productivity hit while helping others on a one-by-one basis.

Negative costs also include the costs attributable to poor quality. For example, training on software inspection methods is very inexpensive compared to the costs incurred by fixing bugs later in the development cycle and during the operational phase. The cost of not training people on the inspection process is a negative cost. The costs of not training the testing people on the proper techniques for testing is a negative cost. The cost of not training the software managers on the benefits of an organized process can result in negative costs when that manager prevents the software practitioners from doing the job they are trained to do. Sometimes these costs are easily calculated by using the experience of other parts of the organization that utilize the tools or techniques or by using the documented experience from other organizations.[25]

---

22. In today's systems, the online help function is a form of "finger-tip" learning. It provides a mechanism whereby the entire technology can be learned on the desktop system. Unlike computer-based training (CBT) in which the student is dedicated to the CBT, "finger-tip" learning allows the student to interact and use the training while working on a job-related problem.

23. Stevens, S., Christel, M., & Wactler, H. "Informedia™: Improving Access to Digital Video." *interactions 1* (1995): 67-71.

24. Schatz, B.R., & Hardin, J.B. "NCSA Mosaic and the World Wide Web: global hypermedia protocols for the Internet." *Science. 265, 5174* (12 Aug 1994): 895-901. Mosaic was developed at the National Center for Supercomputing Applications (NCSA), University of Illinois, Urbana-Champaign.

25. Herbsleb, J.; Carleton, A.; Rozum, J.; Seigel, J.; & Zubrow, D., *Benefits of CMM-Based Software Process,* (CMU/SEI-94-TR-13, ESC-TR-94-013). Pittsburgh, Pa.: Software Engineering Institute, Carnegie Mellon University, 1994. This document contains data from 13 organizations on the costs and benefits of software process improvement.

Computer-Aided Software Engineering (CASE) tools are often not used because of lack of training. Big investments in CASE technology can go unrewarded when an organization provides training to a small group of leading edge software developers, but doesn't follow up with the training of the final users of the technology.

> *"A number of efforts to adopt tools have failed because of an inability to incorporate the tool into day-to-day activities and planning of the organization.... A common mistake is to provide training for a group of early users, followed by only minimal ongoing training. Unfortunately, it is the larger group of users who are not CASE tool 'pioneers' who ... require more training"[26]*

Negative costs also include lost business opportunities, e.g., the inability to be competitive on a bid because people are not aware of certain technologies. Worse yet, the organization may get the contract at a price that is too low to make a profit because people aren't trained to develop software in a cost-effective manner. The cost of doing business may be too high. Negative costs also occur when the knowledge, skills, and abilities of software practitioners are obsolete. The organization can not deliver the needed functionality at the quality required, or the software group cannot advance the state of the art as required by a unprecedented software system.

In an organization that has difficulty accepting its responsibility to train its software people, one of the most effective ways to justify training is to identify the software capabilities needed to be effective and competitive and to calculate the cost of not having that capability. This approach can frequently demonstrate a significant return on the investment in training.

5.8   Student Selection and Enrollment Procedures

The training plan establishes the criteria for selecting students for specific training, including course prerequisites and priority schemes for allocating training resources and seat assignments. The training group uses the student selection criteria to determine which enrollment requests will be honored.

5.8.1   Prerequisites

The student selection criteria should identify the prerequisites that must be met for each of the training opportunities. Course prerequisites can include having a specified level of knowledge and skill in a software engineering practice, having a minimum amount of experience performing a certain task, or occupying an organizational position for which the course is required.

---

[26]. Oakes, K., Smith, D., & Morris, E.,*Guide to CASE Adoption* (CMU/SEI-92-TR-15,ADA258852). Pittsburgh, Pa.: Software Engineering Institute, Carnegie Mellon University, November 1992, p. 9

The training plan should identify who is responsible for enforcement of the selection criteria and define the process necessary to obtain a waiver. The established waiver policy may allow managers to grant prerequisite waivers by attesting to the employee's ability. Possession of a technical degree or college transcripts indicating successful completion of applicable college courses may constitute acceptable waiver criteria. For a sample waiver form, see Appendix E.

Occasionally a self study accompanied by an entrance exam may be used in order to provide the student with prerequisite knowledge so that the instruction during the training can be more focused and less remedial in nature. For example, a course on metrics might require the mastery of algebra and simple graphing techniques, or a course on formal methods may require set theory. In environments where people can become software developers without having taken basic mathematics, an entrance exam may identify students needing remedial help. Some organizations have policies forbidding entrance exams but when there are more students than there are seats available, it is an effective method for establishing a uniform learning situation. The instructor doesn't have to cover materials that most of the students already know, and the test eliminates students who can't fully profit from the course due to gaps in knowledge needed for full understanding of the material. The IBM Software Engineering Institute[27] taught a software engineering course with three such self studies and associated entrance exams. Almost 10,000 students attended these classes. Performance on the entrance exams was a very good predictor of performance in the training. The prerequisite exams had the benefit of providing a class profile that could be used for pacing the presentation and assigning people to balanced teams.

5.8.2 Priorities

The student selection criteria should identify the priorities that will be used in placing students in the classes, especially if there are more students than there are classroom seats. Certain projects will have higher priority based on their mission and organizational goals. It may be desirable to give certain software professional levels priority over other professional levels.

Another method of student selection is to have seats allocated geographically. Seating quotas for high-demand training may be the only way to fairly distribute the available training. Seating allocation can be as simple as identifying a particular class for all of the people in Group X. It can be as complex as a mix of students from a number of projects with "m" seats for Project X, "n" seats for Project Y, and "o" seats for Project Z and spreading them across the course offerings.

[27.] Carpenter, M. & Hallman, H. "Quality Emphasis at IBM's Software Engineering Institute." *IBM Systems Journal, vol 24, no 2* (1985):121-133.

5.9 Course Delivery Standards

The training plan should document or reference course delivery standards. Ultimately the quality of the training delivery will speak volumes to the students about the organization's commitment to training objectives. There should be standards for training facilities. The training environment should meet minimum standards for comfort, visibility of materials, and noise control. The training development process should provide adequate quality control on the materials so that students are not distracted by poorly designed instruction, irritating spelling errors, or amateurish visuals.

Instructor qualifications are an important factor in training quality. This is especially important for internally developed training. The prospective instructor should be good at presenting ideas and be interested enough in the subject matter to have a good knowledge of its usefulness and concepts. The instructor becomes a salesperson for the benefits of using the material being taught, and thus a spokesperson for the organizational goals being supported by the training. Instructors should have the respect of their peers in the organization. A subject matter expert may not be effective as an instructor; a balance between technical knowledge and teaching ability is key to selecting instructors.

With training that is acquired, the reputation of the vendor and the qualifications of the individual instructors should be reviewed. Review instructor resumes for level of subject matter knowledge, experience both teaching and as a software developer and manager, and the amount of time the instructor has spent teaching and using the subject matter. Interview previous clients of the vendor. In order to assure the quality of instruction, the organization should retain the right to reject instructors who have performed unfavorably in the past.

5.10 Training Evaluation and Tracking Procedures

The training plan includes a section on how the effectiveness of the training program will be assessed and what measures will be used to gauge success. Questions to consider include

- How effectively did the training meet its objectives?
- How effective was the training program in providing training to the right people as it was needed?

The training plan needs to consider a blend of at least two perspectives: an instructional design perspective, as represented below by the Kirkpatrick model, and an assessment based on the CMM model.

5.10.1 Kirkpatrick Model Guidance

Training evaluation is most conveniently discussed within the context of Donald L. Kirkpatrick's four levels, which he described in 1959 and 1960 in the *Journal of the American Society of Training Directors (ASTD).*

1. Reaction: Participant opinions regarding the training, its processes, and outcomes. Satisfaction.

2. Learning: The extent to which learning took place during the training. Achievement of skill and knowledge objectives.

3. Behavior: Changes in actual on-the-job performance. Transfer of learning to the work setting.

4. Results: Positive effects of training on the organization. Improvement in quantity and/or quality of output, cost savings, profit, etc.[28]

Training evaluation is often viewed as being subjective and therefore has been looked upon as unfounded. Most organizations perform Kirkpatrick level 1 evaluation, the *"happiness sheets"* that are collected at the end of all classes. Course evaluations tend to reflect short-term emotional reactions to the training experience more than the long-term utility of the training. Nonetheless, from them one can learn what adjustments to delivery style can make learning more effective and collect statistics on general student satisfaction.

Because of pressures to identify the benefits of a course, the training community has evolved to the use of *pre-tests and post-tests* for Kirkpatrick level 2 evaluation. The test questions should map to the measurable learning objectives for the course, e.g., "The student shall be able to cite eight characteristics of good requirements." or "The student shall be able to name the four methods for testing code." Pre-tests and post-tests are not difficult to administer. At the start of the class, the students are given an exam to see how much they know about the content listed in the objectives. At the end of the class, the students are given a similar or the same exam to see whether they learned what the objectives intended. The differences between pre-test and post-test scores are a measure of how effectively the material was taught. The students receive positive reinforcement on their progress and the results can often be tallied for immediate feedback before the student leaves the classroom.

A problem with pre- and post-testing that requires the recitation of short answers is that the pre-test gives the student the impression that the course is a mere rote course and doesn't challenge his or her thinking. Sometimes students assume that the pre-test questions are an indication of what is important in the course and take the stance that they don't need to pay attention to the other content of the course. This type of testing can also provide an environment where the instructors are motivated to get high scores in these tests since it is also perceived as a measure of their ability to teach. The less qualified instructor may take short cuts in other parts of the materials that are essential to the students' ability to use the technology. The challenge is to include content in the objectives and tests that will help the student learn the proper material and to do this in a measurable way. Software professionals need to learn concepts and how to adapt the concepts to their domain. This ability to use the material in each student's unique environment is not something that level 2 evaluation can easily measure.

---

28. Medsker, Karen L. & Roberts, Donald G., *Evaluating the Results of Training,* American Society for Training and Development, San Diego, Calif.: Pfeiffer & Company, 1992, p. 1.

Some organizations have stopped using pre- and post-testing because they could not prove in court that the tests were not used for job qualification and were not culturally biased. The legal issue is sometimes managed by making the tests anonymous or otherwise assuring that test results are not available to the student's manager.

Using the *managers evaluation* approach to evaluation of the training is an effective method. A Kirkpatrick level 3 approach is to have the employees' managers evaluate their effectiveness after taking the training or to track the students' careers to make a comparison of those who had the training versus those who did not.

Periodically the students' managers may be sent evaluation forms to indicate their perspective on the students' ability to use the materials taught in the training. This may be done at the three- and six-month periods after the class. The evaluation form asks for measures of the student's ability to apply the concepts taught or to use a tool or method effectively. Normally, the manager is asked to rate the performance on a scale similar to their annual personnel performance review. The scale may be numeric, e.g., 1 to 5, or text based. Again the use of Bloom's taxonomy (see Appendix F) is an effective scale for such evaluation. What is being reported indirectly is, "How effective was the course for the individual?" The scale needs to have an additional category added to indicate that the student didn't have an opportunity to use the content of the course. There may have been a job change, a change in direction, or many other reasons for not using the materials taught. The problem with this form of training evaluation is that it assumes the manager is sufficiently knowledgeable about the content of the course to evaluate its use. In most organizations this is not the case.

An alternative approach to manager evaluation is for the student to respond to the evaluation questions. Although many feel this gives biased answers, it may be the most accurate way to get the evaluation information if it is done in a non-threatening environment.

For certain kinds of training, *career tracking* can be used to get a global view of a curriculum that is taught for many years. This can be applied when the training is of long duration, a month or more, and is made available consistently over a number of years. This evaluation can be approached in two ways. One way is to examine the progress of the students up the career ladder versus the progress of their peers who have not had the training. In most organizations this approach is difficult, if not impossible, to do. The second way is to look at the individuals who have reached the higher levels of their career ladders and determine what percentage of them have taken the curriculum early in their careers. In one industrial setting, a three-month curriculum on systems development was given to a number of students over a 25-year period. An evaluation of the people at the highest career levels in software indicated that over 70% had participated in the systems development curriculum early in their careers. There could have been many other influences, but it inspires confidence in the training.

Kirkpatrick level 4 evaluation involves determining whether training contributed to the organization's making progress toward meeting its strategic goals. While it is executive management's responsibility to report improvements related to strategic goals, the training program should provide executive management with reports on how well the training resources are being applied toward meeting strategic goals.

Cost savings is one aspect of corporate strategy. A measurement worth tracking is the difference in turnover of trained versus untrained staff. Some studies have indicated that trained staff are more satisfied and tend to stay on the job longer and that untrained staff may be early candidates for "rightsizing" when times get tough. Staff turnover has a high cost, which can be used to offset the expense of training.

No matter how effective the training itself is, the right students must be trained at the right time for maximum effectiveness to be achieved. Quite often training is requested and students are enrolled, but, when the course is offered, the seats are empty because of last minute project pressures. The training plan should establish a procedure for maintaining records of training delivered and who received the training and a reporting procedure for tracking requests for training, enrollments, and actual attendance and course completion. Reporting these statistics to the management team reminds them that training is provided to support their objectives and that when students don't get the training planned, organizational objectives suffer.

5.10.2 CMM Training Program KPA Guidance

A further checklist for training evaluation and tracking is found in the CMM level 3 Training Program KPA in the common features *measuring and analysis* and *verifying implementation.*

> *Measurement and Analysis*
>
> *Measurement 1: Measurements are made and used to determine the status of the training program activities.*
>
> *Measurement 2: Measurements are made and used to determine the quality of the training program.*
>
> *Verifying Implementation*
>
> *Verification 1: The training program activities are reviewed with senior management on a periodic basis.*
>
> *Verification 2: The training program is independently evaluated on a periodic basis for consistency with, and relevance to, the organization's needs.*
>
> *Verification 3: The training program activities and work products are reviewed and/or audited and the results are reported.*[29]

Measurement 1 focuses on tracking the status of the training activities. Part of this is normal project management tracking: training development milestones, course offering dates, budget data, staffing profiles, etc. The CMM suggests some additional measurements specifically related to training: training attendance levels and waivers approved. Both of these measures are indicators of the growth or decline of training needed on particular topics, which may trigger the reallocation and resizing of training resources for future work. These measures may also indicate an erosion of management commitment to the training program, thus triggering further communication with managers or promotion of the training program.

It is appropriate for the training group at the organizational level to consolidate the status data from other training groups within the organization and report on all training activities to senior management. A key part of the organization-level training group's mission is to make the delivery of training efficient, eliminate unreasonable redundancies, and present recommendations for improving the training capability of the organization.

Measure 2 focuses on the quality of the training program. These measures are the Kirkpatrick measures discussed above.

---

[29]. Paulk, M.C.; Weber, C.V.; Curtis, B.; & Chrissis, M.B. *The Capability Maturity Model for Software: Guidelines for Improving the Software Process.* Reading, Mass.:Addison-Wesley Publishing Company, 1995, pp. 221-222.

Verification 1 states that the training program must be reviewed with senior management on a periodic basis. Both the status and quality measures must be reviewed at an appropriate level within the organization. Furthermore, future plans must be reviewed. The training plan itself can serve as the vehicle of that review, as it is revised on a periodic cycle and represents organizational training needs and the plans to address them. Verification 2 requires that the training program be linked with organizational needs and goals. Verification 3 requires that the training program follows its own documented procedures as recorded in the training plan:

- Those responsible for the training plan create it and revise it periodically (see Responsibility for the Plan).
- Course development and acquisition procedures are followed (see Course Development and Acquisition Process).
- Course quality standards are met (see Course Development and Acquisition Process and Course Delivery Standards).
- Student selection and enrollment procedures are followed (see Student Selection and Enrollment Procedures).
- Training is evaluated and tracked (see Training Evaluation and Tracking Procedures).

The CMM further suggests that training records be properly maintained and that individuals designated as requiring specific training complete that training. In several of the software engineering curricula shown in section 5.5 of this document, some training is designated as mandatory, some optional. Within project training plans there are training plans for each employee, showing specific training planned. The organizational level training group can serve as an auditor to

- verify that the training in individual training plans is completed
- verify that individual training plans contain required software engineering curriculum training

A typical human resources function is related to this audit function, overseeing career path progression, a part of which is the training required for an individual to move through a career path. The training group must work in partnership with the human resources group in fulfilling this career tracking function. Significant activities within the career path management function are:

- defining software engineering career paths to be recognized within the organization
- tracking individual progression within career paths
- maintaining training records for employees

The training plan must state how these activities will be shared among the training groups, the human resources group, and other groups such as SEPGs within the software organization.

6   Summary

With a CMM level 3 defined process, a software organization has its management and engineering activities documented, standardized, and integrated across all projects. In order to accomplish this level of maturity, good training practices are critical and need to be defined in the form of an organizational training plan.

The desirable entry criteria for creating a training plan include

- management support through established training policies that promote the development of key skills and competencies
- assignment of resources, including qualified people to create the plan
- consensus on the organizational scope for the training program, ideally including all projects in the software organization
- results of a training needs analysis

The training plan must describe the scope and objectives of the training program and identify responsibilities for funding the program and executing the plan. The plan must identify the strengths and weaknesses of the software organization, as elicited during a training needs analysis. The plan must specify the training to be offered, the schedules of those offerings, and how training opportunities will be allocated across the organization. It must document procedures for evaluating the quality and effectiveness of the training developed or purchased against stated objectives for the training program.

For software organizations that don't know where to start, examples of parts of training plans from other organizations are provided in the guidelines. These may give some ideas on data representation and content. Training plans must be easily understood by those within the organization who will approve and fund them, as well as by the software managers and practitioners who will receive the training.

A good training plan provides a cost-effective approach to developing improved capabilities in the software organization and institutionalizes a process for continuing training improvement.

Appendices

Appendix A   Ratios of Technical Specialists to General Software Population

Jones, Capers. "Software Specialization." *IEEE Computer* (July 1995): 81-82.

Software Specialty	Small (100) Enterprise	Medium (1,000) Enterprise	Large (10,000) Enterprise	Specialists to Generalists
Architecture			X	1:75
Configuration control		X	X	1:30
Cost estimating		X	X	1:100
Customer support	X	X	X	N/A[a]
Database administration	X	X	X	1:25
Education and training			X	1:250
Function point counting		X	X	1:50
Human factors			X	N/A
Information systems			X	N/A
Integration			X	1:50
Maintenance/enhancement	X	X	X	1:3
Measurement		X	X	1:50
Networks	X	X	X	1:50
Package acquisition			X	1:150
Performance			X	1:75
Planning			X	N/A
Process improvement			X	1:200
Quality assurance	X	X	X	1:25
Reusability			X	1:100
Standards			X	1:300
Systems software support	X	X	X	1:30
Technical writing	X	X	X	1:15
Technology (OO, GUI, ...)			X	N/A
Testing	X	X	X	1:8
Tool development			X	N/A

a. N/A indicates data not available.

Appendix B   Training Needs Analysis Study Types

Study Type	Description	Data Collection Methods
Needs Survey	In this simplest method of gathering data, subject matter experts and managers are asked what training is required.	• Management requests • Focus groups • Interviews
Problem Analysis Study	Data is gathered on specific performers to determine whether the problem is a performance problem or another type of problem (such as an attitude problem or a matter of conflicting direction).	• Focus groups • Interviews • Skill tests • Climate questionnaires • Observation • Performance questionnaires • Performance appraisal reviews
Task Analysis Study	Individual tasks are analyzed to determine the knowledge, skills, and abilities necessary to perform the job. Subject matter experts are interviewed and observed.	• Focus groups • Interviews • Skill tests • Performance documents • Observation • Performance questionnaires • Performance appraisal reviews • Exit interviews
Competency Study	Subject matter experts are asked for information on the knowledge, skills, and abilities necessary for broad competencies associated with a job.	• Management requests • Focus groups • Interviews • Performance questionnaires
Performance Analysis Study	This most complex method determines the desired process to complete the job, the job output, the tasks required, and the knowledge, skills, and abilities necessary to do the job.	• Focus groups • Interviews • Skill tests • Climate questionnaires • Observation • Performance questionnaires • Performance appraisal reviews • Performance documents • Exit interviews

Adapted from the American Management Association course, "How to Conduct an Effective Training Needs Analysis," 1994.

Appendix C   Software Engineering Training Plan Sample Outline

1   Scope of the Training Plan

1.1   Organizational Entities

1.2   Software Personnel Categories

1.3   Software Topics and Performance Areas Addressed

2   Responsibility for the Plan

2.1   Management Ownership

2.2   Authorship and Revision Ownership

2.3   Stakeholder Ownership

3   Training Objectives

3.1   Training Policy

3.2   Organizational Goals Supported

3.3   Training Benefits

4   Technical Strengths and Weaknesses of the Software Organization

4.1   How Training Needs Are Identified, Verified, and Prioritized

4.2   Training Needs To Be Addressed

5   Software Engineering Curriculum

5.1   Curriculum Architecture and Funding Structure

5.2   Course Sequences by Job Category

5.3   Course Descriptions

5.4   Planned Course Development/Acquisition Strategy

5.4.1   Estimated Development and Acquisition Costs

5.4.2   Course Development and Acquisition Processes

5.4.3   Course Quality Standards

5.5   Planned Delivery Schedule

5.6   Student Selection and Enrollment Procedures

5.7   Training Evaluation and Tracking

Appendix D   Definition of Software Roles[1]

The following is an excerpt from PRC Inc., Systems Integration Segment Training Resources Guide, 1995:

"The Software Engineering Training Program is a formal course of study designed to develop the skills and knowledge of individuals so that they can perform their roles as software professionals. The training program is based on standard PRC software engineering practices and provides employees with the foundation necessary to achieve improvement in the software process.

Definition of Roles

The **Software Practitioner** is the programmer, analyst, or software engineer working as an individual contributor on any phase of a software project's life cycle. The software practitioner is primarily responsible for developing the work products associated with product engineering and takes direction from a software team leader.

The **Software Team Leader** is the front line supervisor of product engineering activity and manages a group of software practitioners. The software team leader is an experienced software practitioner and is assigned responsibility to plan and direct the work of subordinates in all phases of a software project's life cycle, following standard processes developed for use on the project. The software team leader takes direction from either a supervisory team leader or a software project manager.

The **Software Project Manager** is ultimately responsible for the delivery of technically compliant work products within the resource and schedule constraints established in the software development plan. The software project manager is an experienced practitioner and team leader and directs the work of product engineering teams as will as that of support staff engaged in planning, monitoring, configuration management, and quality assurance activities. The software project manager is delegated authority from senior management to make the required decisions and to deliver products of high quality using the organization's approved processes.

The **Project Management Specialist** is a support staff member, employed on large software projects, who assists the software project manager and engineering staff in the planning and estimating of project activity and the monitoring and control of performance against cost and schedule goals. The project management specialist may take direction directly from the project manager or may be a member of a larger support staff organization detailed to the software project.

---

1. See also "Organization Roles" in Paulk, M.C.; Weber, C.V.; Curtis, B.; & Chrissis, M.B. *The Capability Maturity Model for Software: Guidelines for Improving the Software Process*. Reading, Mass.:Addison-Wesley Publishing Company, 1995, pp. 59-61; and Curtis, Bill; Hefley, William E.; & Miller, Sally. *People Capability Maturity Model* (CMU/SEI-95-MM-02). Pittsburgh, Pa.: Software Engineering Institute, Carnegie Mellon University, 1995, pp. O-62 - O-65.

The **Configuration Management (CM) Specialist** is a support staff member of a software project's CM group, responsible for conducting activity necessary to manage the project's configuration baseline. The CM specialist's activity may be directed by the software project manager on small projects or by a product assurance manager on larger projects.

The **Quality Assurance (QA) Specialist** is a support staff member of a software project's QA group, responsible for conducting activity necessary to verify and audit all technical activity and work products. The QA specialist's activity may be directed by the software project manager on small projects or by a product assurance manager on larger products.

The **Software Engineering Process Group (SEPG) Member** is a member of a SEPG at any level of PRC's software process improvement infrastructure. SEPG members are drawn from all product engineering and support staff groups."

Appendix E   Waiver Form

The following training waiver form is adapted from one used by PRC Inc., Systems Integration Segment Training Resources Guide, 1995.

Project Tracking and Oversight
Training Waiver

Training Objectives

Upon completion of the Project Tracking and Oversight course, the student will

- Demonstrate a good understanding of the PRC Standard System Life Cycle, the PRC Software Process Improvement Program, the SEI Capability Maturity Model (CMM), and the project tracking process.

- Apply the three sub-processes within the project tracking process and list the main tasks of each.

- Given a software development plan, identify the software commitments, staffing, and schedule.

- Select the appropriate measurements for cost, software size, and schedule.

- Describe how the project tracking process is integrated with other CMM key processes and procedures described in PRC manuals.

- Support project groups in applying the project tracking process and in making process improvements in software projects.

I certify that _____ is able to perform the objectives listed above.

_____   _____
Manager/Instructor   Date

_____   _____
Employee   Date

Appendix F   Bloom's Taxonomy

Adapted from:

Ford, G.; Gibbs, N.; & Tomayko, J. *Software Engineering Education: An Interim Report from the Software Engineering Institute.* (CMU/SEI-87-TR-8, ADA182003). Pittsburgh, Pa.: Software Engineering Institute, Carnegie Mellon University, 1987.

Bloom[2] has defined the taxonomy of educational objectives that describes several levels of knowledge, intellectual abilities, and skills that a student might derive from education. We found it beneficial to use this taxonomy to help describe the objectives, and thus the style and depth of presentation, of software engineering topics. The six classes of objectives below are presented in increasing order of difficulty; each requires education beyond the previous class for the student to achieve the objective.

Knowledge

> The student learns terminology and facts. This can include knowledge of the existence and names of methods, classifications, abstractions, generalizations, and theories but does not include any deep understanding of them. The student demonstrates this knowledge by recalling information.

Comprehension

> This is the lowest level of understanding. The student can make use of material or ideas without necessarily relating them to others or seeing the fullest implications. Comprehension can be demonstrated by rephrasing or translating information from one form of communication to another, by explaining or summarizing information, or by being able to extrapolate beyond the given situation.

Application

> The student is able to apply abstractions in particular and concrete situations. Technical principles, techniques, and methods can be remembered and applied. The mechanics of the use of appropriate tools have been mastered.

Analysis

> The student can identify the constituent elements of a communication, artifact, or process, and can identify the hierarchies or relationships among those elements. General organizational structures can be identified. Unstated assumptions can be recognized.

Synthesis

> The student is able to combine elements or parts in such a way as to produce a pattern or structure that was not clearly there before. This includes the ability to produce a plan to accomplish a task such that the plan satisfies the requirements of the task, as well as the ability to construct an artifact. It also includes the ability to develop a set of abstract relations either to classify or to explain particular phenomena, and to deduce new propositions from a set of basic propositions or symbolic representations.

[2.]Bloom, B, "Taxonomy of Educational Objectives," *Handbook I: Cognitive Domain.* New York, N.Y.: David McKay Company, 1956.

Evaluation

The student is able to make qualitative and quantitative judgements about the value of methods, processes, or artifacts. This includes the ability to evaluate conformance to a standard and to develop evaluation criteria, as well as apply given criteria. The student can also recognize and suggest improvements to a method or process and suggest new tools or methods.

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION	1b. RESTRICTIVE MARKINGS
Unclassified	None

2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT
N/A	Approved for Public Release
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	Distribution Unlimited
N/A	

4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
CMU/SEI-95-TR-007	ESC-TR-95-007

6a. NAME OF PERFORMING ORGANIZATION	6b. OFFICE SYMBOL (if applicable)	7a. NAME OF MONITORING ORGANIZATION
Software Engineering Institute	SEI	SEI Joint Program Office

6c. ADDRESS (city, state, and zip code)	7b. ADDRESS (city, state, and zip code)
Carnegie Mellon University Pittsburgh PA 15213	HQ ESC/ENS 5 Eglin Street Hanscom AFB, MA 01731-2116

8a. NAME OFFUNDING/SPONSORING ORGANIZATION	8b. OFFICE SYMBOL (if applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
SEI Joint Program Office	ESC/ENS	F19628-95-C-0003

8c. ADDRESS (city, state, and zip code))	10. SOURCE OF FUNDING NOS.			
Carnegie Mellon University Pittsburgh PA 15213	PROGRAM ELEMENT NO	PROJECT NO.	TASK NO	WORK UNIT NO.
	63756E	N/A	N/A	N/A

11. TITLE (Include Security Classification)
Training Guidelines: Creating a Training Plan for a Software Organization

12. PERSONAL AUTHOR(S)
Maribeth Carpenter

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (year, month, day)	15. PAGE COUNT
Final	FROM TO	September 1995	52

16. SUPPLEMENTARY NOTATION
Appendix A-F 14 pages

17. COSATI CODES			18. SUBJECT TERMS (continue on reverse of necessary and identify by block number)
FIELD	GROUP	SUB. GR.	training program
			key process area at level 3
			guidelines, software projects

19. ABSTRACT (continue on reverse if necessary and identify by block number)

This set of training guidelines focuses on issues to be addressed by the training program of a software organization comprised of multiple software projects. While much of the content of the guidelines is equally applicable to training plans for individual projects, this document presumes a coordinated function providing training across software projects.

(please turn over)

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED/UNLIMITED ■ SAME AS RPT □ DTIC USERS ■	Unclassified, Unlimited Distribution

22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE NUMBER (include area code)	22c. OFFICE SYMBOL
Thomas R. Miller, Lt Col, USAF	(412) 268-7631	ESC/ENS (SEI)

DD FORM 1473, 83 APR	EDITION of 1 JAN 73 IS OBSOLETE	UNLIMITED, UNCLASSIFIED SECURITY CLASSIFICATION OF THIS PAGE

ABSTRACT — continued from page one, block 19