Technical Report
CMU/SEI-95-TR-010
ESC-TR-95-010

Training Guidelines: Purchasing Training for a Software Organization

Maribeth B. Carpenter

Harvey K. Hallman

December 1995

Technical Report
CMU/SEI-95-TR-010
ESC-TR-95-010
December 1995

Training Guidelines: Purchasing Training for a Software Organization

Maribeth B. Carpenter
Harvey K. Hallman

Community Sector

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

This report was prepared for the

SEI Joint Program Office
HQ ESC/AXS
5 Eglin Street
Hanscom AFB, MA 01731-2116

The ideas and findings in this report should not be construed as an official DoD position. It is published in the interest of scientific and technical information exchange.

FOR THE COMMANDER

(signature on file)

Thomas R. Miller, Lt Col, USAF
SEI Joint Program Office

This work is sponsored by the U.S. Department of Defense.

This work was created in the performance of Federal Government Contract Number F19628-95-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 52.227-7013.

This document is available through Research Access, Inc., 800 Vinial Street, Suite C201, Pittsburgh, PA 15212. Phone: 1-800-685-6510. FAX: (412) 321-2994. RAI also maintains a World Wide Web home page. The URL is http://www.rai.com

Copies of this document are available through the National Technical Information Service (NTIS). For information on ordering, please contact NTIS directly: National Technical Information Service, U.S. Department of Commerce, Springfield, VA 22161. Phone: (703) 487-4600.

This document is also available through the Defense Technical Information Center (DTIC). DTIC provides access to and transfer of scientific and technical information for DoD personnel, DoD contractors and potential contractors, and other U.S. Government agency personnel and their contractors. To obtain a copy, please contact DTIC directly: Defense Technical Information Center / Attn: BRR / 8725 John J. Kingman Road / Suite 0944 / Ft. Belvoir, VA 22060-6218. Phone: (703) 767-8274 or toll-free in the U.S. — 1-800 225-3842).

Table of Contents

Acknowledgments

The authors wish to thank their colleagues for their review of this report, their helpful suggestions, and their contributions to the content:

- Ellen Ayoob, SEI
- Sandra G. Behrens, SEI
- Gary Coleman, CACI International, Inc.
- Marsha Pomeroy-Huff, SEI
- Nancy R. Mead, SEI
- Charlene Rauber, SEI
- Sheneui Sloan, California State University at Long Beach

Training Guidelines: Purchasing Training for a Software Organization

In 1994 U.S. organizations with 100 or more employees spent almost $10 billion on educational goods and services from outside suppliers. [1]

Abstract: This set of training guidelines focuses on many of the issues surrounding the purchasing of software engineering training. It includes general guidelines for creating partnerships between training vendors and training purchasers to meet real training needs and a template for use by subject matter experts in elaborating training requirements.

A Brief Quiz for Training Planners and Administrators

If you have responsibility for providing software engineering training for your organization, here are some questions for you to contemplate:

1. Is training in your organization done on an "as needed" basis or as part of an integrated, coherent training plan or curriculum?
2. Do you view training vendors with suspicion or as important partners in meeting training needs?
3. Do consumers of training within your organization perceive vendor training as being of higher or lower quality than training developed in house?
4. Is writing the justification for a training purchase difficult because you are uncertain whether the training actually will satisfy a real need?
5. Do you relinquish the responsibility for selecting vendor courses to the software engineers requesting the training because you have no knowledge of the subject matter of the course or because you don't understand the business issues that drive the training request?
6. How many training vendors have you actually talked to?

If considering the above questions raised your interest, or at least your sensitivity to training vendor selection issues, please read on.

1. "Forecast: Boom Times Ahead For Training Industry." *Training* (April 1995): 20.

1 Introduction

Training is a significant investment for most software organizations. Almost anyone in a software organization might find themselves in the role of purchasing training, either for themselves or for a few project members. Such incidental training acquisition is usually not done on a large scale, but it can still represent a large expenditure. Within many organizations there is a training group that looks after the training needs of the organization. Sometimes the training group function falls upon the shoulders of improvement initiative teams, for example, a Software Engineering Process Group (SEPG) or a training Process Action Team (PAT). These official or *ad hoc* training groups often do not have defined procedures for working with training vendors to acquire training which is well matched to the organization's needs. These guidelines offer suggestions for creating better partnerships with training vendors.

This document focuses on the decisions of whether to purchase training and which training to purchase. It provides a software organization with guidance for purchasing training and gives a training vendor suggestions that can lead to positive, long-term customer-vendor relationships.

The audiences for these guidelines are

- Members of the training group of a software organization.
- The manager of the training group.
- Software engineering subject matter experts who help specify training needs.
- Vendors of software engineering education and training.

The training group is responsible for the entire training process. Training selection and delivery are elements of the training process. To show those activities in context, Table 1 illustrates an abstraction of the training process with the ETVX paradigm,[2] [3] which was first used to document the IBM programming process architecture in the early 1980s. The model has four components:

1. (E) entry criteria
2. (T) tasks
3. (V) validation tasks or criteria
4. (X) exit criteria

---

2. Radice, R.A.; Roth, N.K.; O'Hara, Jr., A.C.; & Ciafella, W. A. "A Programming Process Architecture." *IBM Systems Journal 24, 2,* (1985):79-90.

3. Radice, Ronald A. & Phillips, Richard W. *Software Engineering: An Industrial Approach. 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 Training policy and objectives Resources	Create or acquire training products Pilot and deliver training Evaluate training	Needed training delivered Training objectives met
Organizational context	**Validation Tasks or Criteria** 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 before starting the training process. The training function needs management support and adequate resources to perform its tasks; it needs to have a written training policy and written 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 only on the tasks of acquiring and delivering training products from an outside source. It is assumed that the training group has conducted a training needs analysis, created an organizational training plan,[4] and designed a software engineering curriculum. The guidelines assume that the organizational training group is the focal point for training requests and serves as the requestor's agent in negotiating with the training provider.

---

4. Carpenter, Maribeth B. & Hallman, Harvey K. *Training Guidelines: Creating a Training Plan for a Software Organization* (CMU/SEI-95-TR-007, ESC-95-TR-007). Pittsburgh, Pa.: Software Engineering Institute, Carnegie Mellon University, 1995.

The training process validation tasks or criteria insure that the output of the tasks meet required standards and that the tasks were performed properly. The exit criteria define the output state resulting from the execution of the training process.

Assuring that scarce training funds are spent wisely is one of the responsibilities of a training group. Cost considerations affect all aspects of the training process. The training group is also responsible for assuring that training solutions, whether they are developed and delivered in-house or purchased from external sources, meet the needs of the organization.

The guidelines lead the reader through a series of questions:

- When is a training solution appropriate?
- When is it appropriate to purchase training?
- What information should a training vendor supply?
- How do you select a vendor?
- What should you provide a training vendor?

The guidelines then describe how a software engineering subject matter expert can specify the training needs for a requested topic. The training specification serves as a negotiating tool for working with a training vendor to provide training that meets the needs of the organization. If purchasing the training turns out to be infeasible, the same specification can be used for in-house development of the training.

Many organizations make limited use of commercial education vendors and universities for satisfying their software engineering education needs. Often cited reasons are high cost and the perceived difficulty of tailoring training material to the organization. Training vendors recognize these concerns and increasingly emphasize their willingness to tailor their products and to offer them on the customer's site to save student travel expenses. The most commonly used external source of training is vendors of software products who provide training on the use of their products[5, 6]. Of those software organizations who outsource training, a majority deal with a very limited set of training providers with whom they have established a relationship over time or who have developed an outstanding reputation within the software community.

This document suggests ways to build better partnerships between software engineering organizations and training vendors. Software organizations can do a better job of specifying their training needs. Training vendors can do a better job of describing their products in terms of training outcomes relative to the customer's needs.

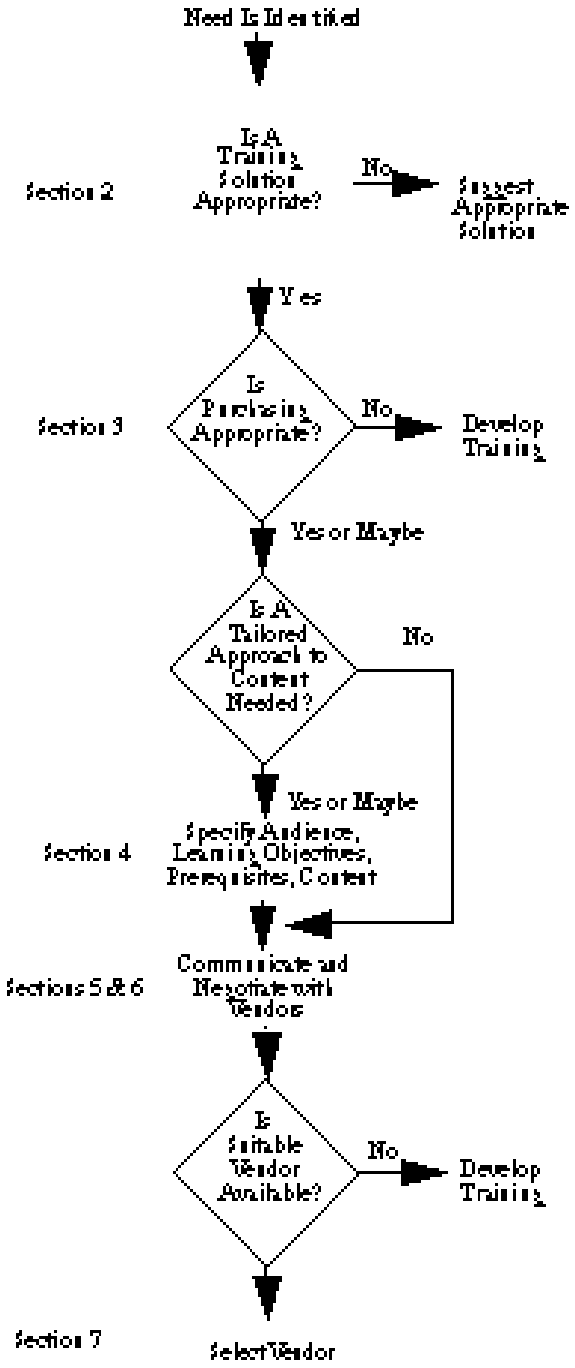
---

5. Early in 1995 the SEI conducted structured interviews with 10 software-intensive industry and government organizations to ascertain the degree of usage of outsourced training and what problems are encountered in purchasing training.

6. Jones, Capers. "How Software Personnel Learn New Skills." *IEEE Computer* (December 1995):88-89. Software Productivity Research, Inc., estimates that 500,000 software professionals will take at least one software product course as compared to 300,000 who will take at least one other outsourced software training course.

Figure 1 provides a temporal perspective on the material discussed in this document.

Figure 1: Mapping Training Acquisition Steps to Guidelines



2 Determining Whether a Training Solution Is Appropriate

Before you start rooting through your files for that course brochure on executive basket weaving, have you asked yourself the basic question, “Is the problem behind this request for training one that is amenable to a training solution?” Do the executives need a new hobby or is the new boss into baskets? What is the real problem? Maybe the policy on executive assignment rotation needs re-examining. There are many problems that no amount of training can solve, yet they are laid at the feet of the training department. Don’t let the organization put inappropriate monkeys on your back. Learn to question and to say no when appropriate.

Some requests for training should be questioned. These include requests for which the reason for the training is to provide motivation, reward employees, enhance political positioning, or react to the latest new trend. The only reason to provide training is that something new needs to be learned in order to maintain or enhance work performance.

2.1 Motivational Classes

Consider the situation where the software practitioners have the knowledge and skills needed for their jobs but job performance is disappointing. Can you teach them to be more motivated? Training can improve motivation, but before you propose a training solution to the performance problem, find out the answers to the following questions.

- Do high-level performers get punished by getting more work assigned to them?
- Do low-level performers get the same rewards as high-level performers?
- Are projects poorly planned with inadequate time, money, and tool resources?

Training won’t solve problems that are rooted in communications, management, or policy issues. Good training, when needed, helps people do their job well and this enhances motivation.

2.2 Training As a Reward

Suppose that the software project manager presents the following request: “Sam Smith, a senior software engineer, did an exceptional job of getting the last project out of the ditch and smoothing over a potentially ugly customer relationship. Can you provide him a class that will make him feel like the very important person he is?”

In considering this request you may recall some good reports on the “software industry of the future” courses at a university in the next state. Sam would fit right in with the elitist audiences those courses attract. But Sam is burned out after that last project. His family hardly knows him. Before you register Sam in the university course, discuss the following issues with Sam and his manager.

- Will he feel rewarded by a week of business travel and the demands of high-energy classroom interactions?
- Can Sam or his manager suggest some professional development opportunity that will contribute to Sam's stature within the software engineering community? This might be the opportunity to write a paper during business hours for presentation at an upcoming professional conference or to select an educational offering that expands Sam's breadth of expertise and thus his career options.
- Is the timing right to present Sam with additional short-term impact to his work load and calendar? Perhaps an easing of schedule demands would be more positively received.

Training is not a reward. Obtaining needed learning, however, can be very rewarding.

2.3 Training As Politics

Suppose that the organization has new leaders who want to supplant the old culture and make a fresh start. The training program is an important aspect of how people perceive the organization. Therefore, you receive a request to replace the old courses in software quality improvement developed under the old regime with new courses using "name brand" instructors. How might you respond?

Training decisions based upon non-training issues should be examined in light of training effectiveness and cost effectiveness. Present the requestor with data on the effectiveness of the current training and attempt to elicit desired changes in the training outcome. Present data on the costs of developing the current training and the projected costs of developing new training. Perhaps the requestor is right in suggesting needed improvements. Try to turn the request into a dialog that will enhance the credibility of the training group by suggesting a reasoned approach. Try to avoid replacing a perfectly good existing course with a new one just to highlight new management or cultural change. This wastes resources already invested and most likely will not improve the training outcome.

2.4 Reactive Buying

Good marketing makes us perceive that we have a need for the product being sold. The advertising may appeal to social needs, our feeling of self worth, our competitive spirits, etc., or to social responsibility, to protection of the environment, and to our future financial security. Writers of books and course vendors sell ideas and methodologies, often motivating us with the promise of competitive advantage and cost savings. What might you do if one of the software managers wants the whole organization trained in the latest software architecture concepts?

It is time to work with the software manager to determine what motivates the request. Don't let the organization be sold a training solution that has no obvious corresponding problem. Discuss with the manager questions that will help focus the training request so as to maximize training effectiveness, for example:

- Who within the organization is best able to benefit from learning the latest thinking on software architectures?
- How will this new knowledge be used in current and upcoming projects?
- What measurable benefits does the manager anticipate from applying the new theories?
- What will be the time frame of application of the knowledge?

Be sure to thank this manager for providing insight into present and future training needs and for having the foresight to fund needed training for the organization. You want to maintain the manager's enthusiasm and sponsorship.

3 Determining Whether It Is Appropriate to Purchase Training

Once you have decided that training is an appropriate solution to the real need behind a training request, you have to weigh the options between developing the training in-house, hiring a training vendor to develop the product, purchasing an off-the-shelf training product, or buying and tailoring an off-the-shelf training product. The factors to consider boil down to demand, cost, time, and available expertise.

The decision whether to "make or buy" is seldom clear cut. The training group rarely has all the information and foresight to be fully confident of their decision. Purchasing seems to be most clearly indicated in any of the following situations:

- Demand is low.
- Development costs would be high compared to the cost of purchasing training for the projected number of students.
- The training is needed quickly and doesn't currently exist in-house.
- In-house expertise is not available.

Even in these situations, the training group needs more information about what is available for purchase and how well it will meet the training needs.

3.1 Demand

Purchasing training is more cost effective than custom development if only a few individuals need training or if training for a larger group is nonrecurring. In scoping the demand for training, consider these questions.

- How many people in the current organization need the knowledge and skill the proposed training will provide?
- Will the current training need recur in the future? Consider the organization's personnel turnover rate and the frequency of job rotations.
- Does the training support a competency that the organization wants to maintain; that is, will there continue to be a demand for the in-house expertise being taught, or in the future will the organization have no need for the work or outsource the work? The level 3 key process areas of the People Capability Maturity Model[SM 7 8] describe the activities of identifying the business processes in which the organization must maintain competence and planning workforce development to meet future needs.

---

7. The Capability Maturity Model is a service mark of Carnegie Mellon University.

8. 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.

Even when demand is high, it may be advantageous to outsource training development and/or delivery. Devoting the organization's training personnel, software managers, and software engineering subject matter experts to satisfying long-term training demands may not be the best use of human resources. Correspondingly, the individuals involved may have career objectives incompatible with the support of sustained training delivery.

3.2 Cost

In order to weigh the cost advantages of purchasing training, one needs to know basic financial information about the organization's training function. Basic estimates or rules of thumb are needed, for example, for

- the cost of developing training in-house, measured in some consistent unit (such as an hour of instruction).
- the cost of tailoring vendor materials.
- the costs of administration for a course offered in-house, whether taught by the organization or a vendor. Administrative costs include support for registration, facilities, standard supplies, instructional materials, training management, etc.
- student labor costs.
- representative travel expenses for students attending off-site courses.

Armed with estimates based upon experience, it is possible to compare the cost of developing the training to the cost of purchasing and tailoring the training. In calculating when it is more cost effective to purchase or license training, consider these questions.

- At what level of demand does it become more cost effective to send students to a vendor's course rather than to offer the training in-house?
- Does the vendor offer a group discount? If so, is there sufficient demand to take advantage of the group discount?
- Do the vendor's licensing terms limit usage of instructional materials to a specified population, for example, your organization's employees only, or to a specified geographical area? Are extra fees charged for use otherwise?
- Is tailoring of the vendor's instructional materials by your organization allowed by the licensing agreement? Are there third-party copyrights involved for which your organization will owe usage fees?
- Is the vendor's fee based upon the number of students who are taught using the vendor's materials, and, if so, how are records maintained and verified? Will the organization incur additional administrative costs to maintain records and pay fees?

We also need an understanding of the organizational cost priorities. If delivery costs are more critical than development costs (perhaps because of high student labor costs), it may be better to suffer a higher development cost to customize and optimize the delivery of externally developed courses than to purchase them as is.

3.3 Time

If the training is needed in the near future and you don't have the training available in-house, purchasing may be your only option because of the development time required for a quality training product. For the current training need, determine the relative priority of timing, cost, and quality.

3.4 Expertise

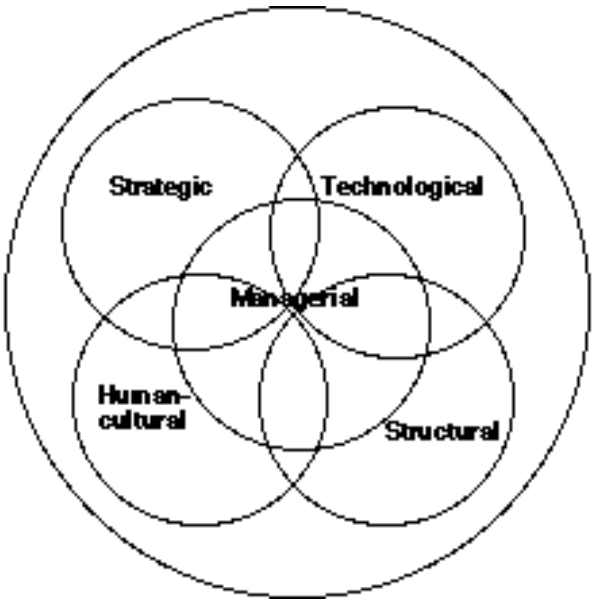
For the training topic requested, determine whether there is a qualified team available to develop a course in-house. Team members needed are subject matter experts to specify and assist in the creation of training materials, instructional design expertise to develop or tailor the training materials, media experts, and qualified in-house instructors to deliver the training. To obtain a qualified team, you must plan when particular talents are needed within the training development cycle, identify specific individuals, and negotiate for their time.

The desired style of delivery may impact the purchasing decision. The creation of computer-based or video-based training requires a suite of talent than may not be available to you in-house. Many companies specialize in certain media productions and can achieve significant improvements in production efficiency and quality over what less experienced people can do. Computer-based training and video-based training is often purchased because of the need for continuing maintenance by specialists and because of the quality expectations of trainees who are accustomed to professionally done productions and computer-generated special effects.

For some topics you may have no in-house expertise. Lack of internal resources and expertise is a strong argument for purchasing training. On the other hand, if the topic is highly organization specific, purchasing is probably not an option.Even when in-house expertise is available, it is sometimes worthwhile to conserve internal human resources and bring in outside viewpoints. External training may be of more or less value based upon the content and the desired outcome of the training.

Gareth Morgan[9] discusses a view of an organization as a system composed of interacting subsystems. See Figure 2 for an abstraction of this organizational view.

Figure 2: Organizational Subsystems



Training is often used to help an individual perform more effectively in one of the subsystems. We can use this organizational view to rank the value of external training. See Table 2 for an example of ranking of training associated with organizational subsystems

9. Morgan, G.*Images of Organization.* Beverly Hills, Calif: Sage Publications, 1986, p. 49. Original source was *Contingency Views of Organization and Management* by Fremont E. Kast and James E. Rosenzweig, Science Research Associates, Inc., 1973.

.

Table 2: Value of External Training in Organizational Subsystems

Organizational Subsystem	Value of External Training
Strategic:	
Planning methods	High
Organization-specific strategy	Low
Technological:	
Tools	High
Methods	High
Industry standards	High
Structural	Low
Human-Cultural:	
Human skills	High
Organization-specific culture	Low
Managerial:	
Awareness/concept	Medium
Experience/application	High
Organization-specific practices	Low

4 Specifying the Subject Matter To Be Taught

In some training situations, it is advisable to let the training vendor decide what subject matter needs to be covered in the training. This is true for basic courses on the use of software tools or systems when there are manuals available to document the way the software works. For example, a course on using PowerPoint® to create documentation and presentation materials doesn't require a specification of what features of the tool are to be taught, although the trainer should be aware of how the tool will be used. The training vendor can supply such courses effectively without the receiving organization describing the course in detail. However, you should be prepared to explain to the vendor what methods and tools are currently being used by prospective students.

When the training involves a less straightforward, more complex subject, such as how to create a software architecture for a system, the organization should determine what topics are important before talking seriously with a training vendor. For example, if the systems developed by your organization are primarily real-time embedded systems, an architecture course that discusses the architecture of financial systems is of little use to you, but one that concentrates on architectures with concurrent and distributed systems may be appropriate. These requirements must be documented. The more specific the training is to the needs of the learners, the more effective it will be.

4.1 Developing the Specification

A subject matter expert skilled in instructional knowledge identification and specification develops the specification of the needed training. The subject matter expert is typically not the same person as the requestor of the training, although this may be the case if the request is for the training of others. The subject matter expert may need to work with the requestor to clarify the need motivating the request.

For a specific topic area, the subject matter expert works with an instructional designer to identify the learning objectives that training can accomplish, the prerequisite knowledge and skills needed by the student, and the topics within the subject to be covered by the training. These objectives, prerequisites, and topics need to be identified for each category of student needing the training. The most efficient format for display of this information is a table.

Table 3 is an example of such a table. The rows contain in outline form the topics that might be taught as part of the training. Sources of topics may be your in-house expert on the subject, a consultant, textbooks on the subject, outlines of courses, articles on training published in the proceedings of conferences, or even the manuals for the product that caused the training to be required. Be sure to save the background information acquired while preparing the table. It will be useful in negotiating with training developers if a specific training needs to be developed, whether vendor or in-house. See Appendix B for an example of the use of the table for the topic of software architecture.

Table 3: Subject Matter

STUDENT TYPE	A	B	C	D	E
LEARNING OBJECTIVES - for the student					
PREREQUISITES					
COURSE TOPICS					

COLUMN - **STUDENT TYPE**

A:

B:

C:

D:

E:

APPLICABILITY

S: Suggested

R: Recommended

M: Mandatory

CODE

[applicability], [level of learning]

LEVEL OF LEARNING

K: Knowledge

C: Comprehension

Ap: Application

An: Analysis

S: Synthesis

E: Evaluation

The “Student Type” columns indicate the types of student and the rows designate the specific learning objectives, prerequisites, and topics that apply to each type of student. The need for training differs by student type. Set up a column for each population segment the training will address.

In the learning objectives section of the table, use X’s to indicate which learning objectives apply to each student type. In the prerequisites section of the table, use S’s to indicate suggested prerequisites, R’s to indicate recommended prerequisites, and M’s to indicate mandatory prerequisites.

In the Course Topics section, use two-value codes **[Applicability],[Level of Learning]** to indicate course topics needed for each student type. A blank cell indicates that the topic is not needed. The first value of the code indicates how applicable the topic is to the group of individuals identified in the column. The second value projects the level of learning the group should achieve on the topic.

The values used for **Applicability** are

- “S” - It is suggested that the topic be included in the training for the group identified. Knowledge or experience with this topic isn’t absolutely necessary in the performance of their job.
- “R” - it is strongly recommended that the topic be included in the training for the group identified.
- “M” - Some topics will be deemed mandatory, as the training will not be acceptable without them. When this occurs, the code should be “M” to indicate that the training must include the topic.

The values used for **Level of Learning** are explained in detail in Appendix A: Bloom’s Taxonomy. Each level includes and builds upon the levels above it. A short summary of these is provided here:

- “K” - Knowledge - The student needs to know terminology and facts.
- “C” - Comprehension - The student needs to be able use the material or ideas without necessarily relating them to other ideas or seeing the fullest implications.
- “Ap” - Application - The student needs to be able to apply abstractions in particular and concrete situations.
- “An” - Analysis - The student needs to be able to identify the constituent elements of a communication, artifact, or process, and be able to identify the hierarchies or relationships among those elements.
- “S” - Synthesis - The student needs to be able to combine elements or parts in such a way as to produce a pattern or structure that was not clearly there before.
- “E” - Evaluation - The student needs to be able to make qualitative and quantitative judgements about the value of the methods, processes, or artifacts.

Different amounts of training are required to bring a student to the designated level of learning. A software executive-level manager may need the comprehension level of training in order to talk intelligently with the individuals that are using the subject matter. The software practitioner may need the analysis level, while the subject matter expert may need the synthesis level in order to perform their jobs. Each higher level of learning requires more time to attain. To achieve the higher levels of learning, students may need to enter the training with a higher level of prerequisite knowledge, skills, and abilities.

When the table has been developed, it should be reviewed by the people requesting the training for agreement on the content and intensity of the training. Changes can be made and additional mandatory topics identified. Occasionally the table can be used interactively while soliciting the training needs to help identify the different levels of students for the proposed training.

4.2 Acquisition Alternatives

Once the table has been filled out for a specific type of needed training and has been reviewed with the requestors of the training, it can now be used to acquire the training. Using the table, compare the training available to the needs of the organization. If courses are available for purchase, they can be evaluated for applicability using the table. Several situations might exist.

- Training is available on the subject matter with all of the recommended and mandatory topics covered.

  This situation seldom exists, but if it does, the training solution may have been identified. The training product still needs to be evaluated for demand, cost, time, and available expertise, as discussed in Section 3.

- Training is available on the subject matter, but not all of the recommended and mandatory topics are covered.

  If a specific course is appropriate except that it lacks one or more specific required topics, negotiations with the vendor are necessary in order obtain the course needed. As an alternative to having the vendor expand the course, a second course can be developed in-house to add the missing training after the student takes the selected course. The second course can also provide the in-house culture desired for the use of the technology that is usually missing from vendor training.

- Vendors may provide similar training but not the specific training needed.

  A desirable training vendor may be able to develop the required training to the specifications of your organization. The table and the backup materials obtained during the development of the table will provide the technical content needed for the training.

- No vendor currently provides or is willing to create the needed training or similar training.

If no qualified vendor can be found to provide the training, it may be necessary to develop the training in-house. The table and backup materials give the developer direction on what needs to be included.

All courses, whether purchased or developed in-house, will likely contain additional objectives and topics. As the course developer realizes that certain background topics are necessary in order to develop the mandatory topics, further enabling objectives and their supporting topics will be included. Conversely, cost and time constraints may dictate that some topics be omitted. The table will be helpful during content negotiations among training developers and training requestors to ensure that needed background material is included and to prevent omission of mandatory topics.

5 Information a Candidate Training Vendor Should Supply

A common complaint is that the information provided by the vendor on their training product is inadequate to understand the content of the course, what students will really learn, which topics will be covered in depth, and how useful the ideas will be to the students' environment. Too often the course brochure or course catalog is the sole source of information used by the purchaser in deciding whether to acquire the course. The premise of this section is that coming to an understanding of the training product is the shared responsibility of the training vendor and the purchaser.

The course brochure is not the purchaser's only recourse for information. There is almost always a phone number to call for more data. It may take some detective work to reach a person who is knowledgeable about the course (usually the instructor) but it is worth the effort. Training is a substantial investment: tuition, travel, student time, and lost work opportunity. Getting value in return for your investment is worth further exploration of what you'll get for your training dollar.

If you don't have the following information, ask the vendor for it.

- A description of the intended audience for the course. A course can't be all things to all people. The audience should be targeted to those with certain prerequisite knowledge and experience and those with certain job responsibilities or organizational objectives. Nothing dooms an educational experience to failure more quickly than having a mismatch between the audience expected by the course author and the actual participants.
- A syllabus, course outline, schedule by topic. What content will be covered and in what depth? On which topics are greatest emphasis placed? What is the balance between student doing and student listening? Consider whether the schedule is too intense or too relaxed for the planned audience. There should be time for reflection, application, and reinforcement of the content.
- Specific measurable learning objectives. Are the learning objectives reasonable given the expected audience (content covered versus anticipated prerequisite knowledge) and the course structure (length, depth, types of activities, opportunities to test learning)? Will the course itself attempt to measure achievement of the objectives through testing or demonstration of skill? If the objectives are not met, will the vendor provide remediation? Consider whether the goals of the course are in concert with your organizational objectives.
- A list of materials the students will receive or have access to during the course. Copies of course visuals and notes are the most common student handout. Does the student receive a textbook, case studies, exercises, tests, videotapes, a voucher for follow-up consultation, a course completion certificate, continuing education units? Does the course include appropriate hands-on use of computer-based tools or practice sessions on the skills being taught?

- References. Course brochures often list the names of prestigious organizations as clients. The vendor should be willing to provide you with a few names of individuals from those organizations whom you can contact for their retrospective on the value of the course. Don't be embarrassed to talk to the references. You may learn some important tips on how to make the course more effective through pre- and post-course activities.

  Has anyone from your organization (other branches or locations) taken this course or other courses from this vendor? If so, did that person feel the course lived up to its advertising and was worth the cost of the training? Did this person have training needs similar to the current ones?

- Instructor credentials. The vendor should provide you a short biographical sketch of the course authors and instructors. Look for indications that the course materials and the course delivery will relate to the instructor's credentials. Does the instructor have current experience in the technical domain, the culture, the business environment of the students? The relevance of that harmonious perspective will vary with the course content, but is always there to some degree. While it is possible to learn a software design method from a generic viewpoint, it is far better if the instructor can show how the method expresses the design concerns relevant to your organization's application domain.

While the course brochure can't tell you everything you need to know, it can offer some important clues to the vendor's perspective toward the product and its delivery.

- Teaching versus learning. Which is being marketed: what will be presented or what will be learned? The course content should be described in simple terms. Be wary of courses described in arcane vocabulary designed to impress you with the superior knowledge of the instructor. Be wary when the name recognition of the instructor takes undue precedence over the concerns of the learners.

- Teaching versus marketing. Generally speaking, course content should be based upon more than one source, more than one perspective. Here again you need to look at your organizational objectives. If you are looking for training in the use of Product X, you don't expect the vendor to teach you about the features of their competitor's products as well. However, if you are looking for training in software cost estimating, you should expect to learn more than one tool or method.

6 What You Should Provide Training Vendors

When you bring a vendor in-house to deliver training, you have a large mutual commitment of resources. Both you and the vendor want to maximize the probability of a successful training experience.

6.1 A Collaborative Stance

Throughout the information gathering and negotiation stages, the purchaser should initiate a collaborative stance with prospective vendors. It is in the interest of both parties to share appropriate information so as to create the best possible solution to the organization's training needs. Here is a list of what the organization should provide vendors.

- A willingness to negotiate. Don't saddle training vendors with unrealistic constraints: unpaid course tailoring, inadequate facilities, too little time for students to master what they are supposed to learn.
- Results of your training needs analysis. Presumably, if you are purchasing a course for in-house delivery, you have analyzed the training needs of the organization, and that analysis has led to the selection of this course. The vendor should know what needs are to be satisfied by the course. (See "Specifying the Subject Matter To Be Taught" on page 14.) What performance gaps are to be addressed? What knowledge and skills are lacking? What new systems and procedures are being initiated? This information will help the instructor relate to the students more readily and tailor both the materials and the emphasis placed upon them to the students' context.
- Organizational knowledge. The instructor needs to know what business you are in. A corporation's annual report, an organization's mission statement, statistics on number of branches and locations, identification of major clients, descriptions of significant projects and organization charts can be helpful cues for building relationships with students and helping the instructor understand the issues the students face.
- The organization's training program curricula. The instructor needs to know what other training is being offered to the students. What related courses have the students taken or will they be taking? How is this course expected to fit into the curriculum?
- A description of the audience. You should minimize the number of surprises for the instructor. How many people are going to attend? What are their job functions and responsibilities? How will they be applying the course material back on the job?
- A preview of the facilities. The instructor needs access to the classroom facilities before the day of the class. He or she needs to become familiar with and to test the visual capabilities: projectors, boards, flip chart positioning, VCR and monitor. The arrangement of student seating, if flexible, should be

designed by the instructor. Any computer software to be used should be tested for compatibility with classroom computers. The instructor should be informed about the location of rest rooms, food service, smoking facilities, and emergency exits.

6.2 Professional Courtesy

Whether you are sending a single individual to a vendor's course or bringing the course in-house, there are certain professional courtesies to which the vendor is entitled. The purchaser should provide

- Respect for the vendor's copyright. Significant investment goes into the design and development of educational materials. It is wrong to abuse the vendor's rights by copying materials or re-teaching the course using the materials without express agreement with the vendor. People sometimes fall into the trap of thinking that "educational use" should be free. You can get into serious legal trouble, as well as cause organizational embarrassment.
- Enrollment of students appropriate to the course. The students enrolled should meet the criteria for the course's intended audience and its prerequisites. Most vendors rely upon the enrollees to "self select." In rare cases, the students are asked to apply for enrollment and are accepted if they have the appropriate credentials. Hopefully, every course is designed for an intended audience and will likely not be effective for others. If the course is meant for managers, don't enroll a technical staff member. Be aware that managers like to delegate, but they can't delegate their own learning.
- Distribution of prerequisite assignments. If the students are to do pre-class reading or bring data with them to class, verify that each student has received the assignment in sufficient time to be adequately prepared. Provide an indication of the effort required to complete the assignment, so that the student won't discover on the airplane that she needs more time or more data.
- Students who are ready to learn. Prior to participating in a course, the students should know why they are going to class and what business decision motivated their enrollment. They should receive basic information about the course: the course description, the intended audience description, learning objectives, instructor credentials, and class schedule. There should be as few surprises as possible. The instructor shouldn't have to play the management role of explaining the relevance of the training to the student's job and why the student is there.
- Management support. For an in-house offering, a manager within the reporting structure of the students should be invited to welcome the students, express his or her view of the value of the training as it relates to organizational goals, explain how the knowledge and skills acquired in the training will be utilized back on the job, and introduce the instructor. This lets the students know that this training is relevant to the organization and that learning is supported by management. Management support also helps the instructor by clarifying ambiguity that the students may be feeling toward the class.

- Uninterrupted student participation. It should be made clear to the students and their managers that, during the class, the students are expected to be present and attentive to course activities. Managers should be discouraged from pulling students out of class and should relieve them of normal work responsibilities during class hours. Students should be discouraged from returning to their offices during breaks or doing their regular work in the classroom.

7 Selecting a Training Vendor

After you and the prospective vendors have exchanged a lot of information, you are left with the decision of which vendor to select. Collecting the data suggested in the prior sections will give you some data points upon which to base your decision, but there are other considerations.

- Value. Which vendor gives the most "value" for the money? Cost is only part of the equation. Consider what tangible and intangible benefits might tip the scale in favor of one vendor over another. These might be
  - first-hand knowledge of the organization's business or application domain
  - a willingness to tailor the product
  - participation in a long-term relationship with the organization through which the vendor offers other products and services, for example, consulting, follow-on workshops, an integrated curriculum approach rather than a point solution, etc.
  - superior credibility with the audience
- Relevance. How relevant is the vendor's solution to the organization's needs? Can the course be used as is or will it require extensive tailoring? If tailored, do the tailoring costs offset other requirements, such as timeliness?
- Geography. A local vendor may be able to offer more post-class support to students, although electronic communications may minimize instructor access issues.
- Availability. The timeliness of training is crucial. How well does the vendor's time table match your needs?
- Past performance. Each time your organization purchases a vendor product, data on the product's quality and effectiveness should be kept in the organizational "memory." Does the vendor have a track record for delivering high-quality results? Has the product under consideration been used within the organization before? It is usually wise to send one or two students from the target audience to the class before purchasing it for a large group. Be sure that the "scout" is typical of the target population, not someone who is already an expert in the subject matter. Perhaps in collaboration with a subject matter expert and instructional designer, query the student about the product's depth, relevance, and value to the organization.

8 Summary

Once a legitimate training need has been identified, factors such as expected demand and resource circumstances point to an initial decision as to whether purchasing training can be a cost-effective solution to training needs. If purchasing may be appropriate, unique aspects of the training needs are specified. Then begins a process of information sharing with vendor candidates. Training success can be optimized if the vendor relationship is considered a partnership by both parties. An atmosphere of open communication and the exchange of vital information is needed if training requirements are to be well matched to training solutions. The dialog and negotiation with vendor candidates provides the training group with the information that will form the basis for the selection of a vendor who will satisfy the training need.

Appendix A 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[1] has defined a 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 other ideas 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 concrete situations. Technical principles, techniques, and methods can be remembered and applied. The mechanics of the use of appropriate tools within a given context 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.

Evaluation

---

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

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.

Appendix B Software Architecture Development: Training Specification Example

This appendix contains an example of a training specification as it might appear for the training subject Software Architecture Development.

0.1 Subject Matter

In this example, the subject matter was collected from a number of sources. (See Appendix D.) Some of these sources were actual courses covering some of the subject matter. Others were textbooks on the subject or related subjects. Others sources could have been local experts in the area, specialists with the subject matter, or individuals who are influential in determining what tools and standards should be used by the organization.

These inputs were merged in a logical order into an outline of the materials, one that perhaps could be used to teach a course. This appears in Appendix C. It also contains cross-reference numbers to the source of the materials for use later in understanding and preparing a course. The higher levels of the outline were added to the table below which were then used to determine the applicability and level of learning needed for each type of student.

When this type of table is created by a training organization, it would then be reviewed by the target organizations for agreement on the subject matter and intensity of training needed by each student type. In this review, Appendix C and D materials are useful in understanding the intended subject matter. During this review, it will become obvious that some of the topics are mandatory if the prospective course is to be successful. They should be so indicated.

0.2 Student Types

In the “Example: Software Architecture Development,” the columns are set up for:

- “A” - Executive Management
- “B” - Other Management
- “C” - Subject Matter Specialist
- “D” - Software Practitioner
- “E” - Support Personnel

Executive management is the top level of management that has software engineers reporting to their lower level managers.

Other Management is the lower level middle management and first-line management in the organization. In some organizations, it may be advisable to divide this group into more specific areas.

Subject Matter Specialist is a person who has become an in-house expert on the subject matter being acquired. This person often acts as an in-house consultant in solving problems in the use of the subject matter or in installing related practices within the organization.

Software Practitioner is that person who will use the subject matter of the training in everyday work. Software practitioners are typically the largest audience for the training.

In some organizations, *Support Personnel* perform more narrowly defined tasks and may not need the same level of training in the subject matter as the software practitioners. Examples of support personnel are the people who collect the software metrics on the project, prepare the status charts, support the entry of the design of the software system into the computer, or access the system to prepare reports.

Example: Software Architecture Development

STUDENT TYPE	**A** - Exec.	**B** - Mgt.	**C** - Spec.	**D** - Pract.	**E** - Supt.
LEARNING OBJECTIVES - for the student					
Describe what a software architecture is and how it is created.	X	X	X	X	X
Make management decisions regarding the architecture development process.		X	X		
Create a software architecture, given the requirements and objectives.			X	X	
Define what methods and tools will be used to define the architecture. Guide others on how to use the methods and tools.			X		
Use the tools to enter the architectures of others.					X
PREREQUISITES					
Experience in software development	S	S	R	R	
Set theory, algebra, statistics			R	R	
Computer science knowledge			R	R	
Use of a personal computer	S	S	R	R	R
COURSE TOPICS					
1. Software Systems Architecture - What is it?					
1.1. Introduction What is a software architecture? Architecture and design within the system life cycle framework Content of an architecture	R,C	R,C	R,Ap	R,Ap	S,C
1.2. Architecture-Driven System Characteristics	S,K	R,C	R,S	R,Ap	S,C
1.3. Architectural Strategies for Software Systems Data flow architectures Communicating process architectures Call-and-return architectures Virtual machines Data-centered systems (repositories) Heterogeneous architectures Distributed systems architectures Concurrent systems architectures Mixed use of idioms in software architectures	S,K	S,K	R,E	R,S	S,C
1.4. Standards The role of architecture standards System standards	R,C	R,Ap	R,E	R,An	R,Ap

2. Creating the software system architecture - How is it done?					
2.1. Introduction Principles of good architecture development Elements of a software system architecture Architecture development process Architecture verification and validation Iterative process System integrity - requirements & configuration	R,K	R,C	R,S	R,An	S,Ap
2.2. Entry Criteria System requirements analysis completed Organizational objectives Acceptance criteria defined	R,K	R,C	R,S	R,An	S,C
2.3. Process Step-System Feasibility Analysis					
2.3.1. Concept Development and Analysis		S,K	R,E	R,S	
2.3.2. Technical Analysis - Prove new architectures can work Models Simulating Prototyping		S,K	R,E	R,S	
2.3.3. Schedule Analysis		S,Ap	R,An	R,Ap	
2.3.4. Cost Analysis		S,Ap	R,An	R,Ap	
2.3.5. Tradeoff analysis - Trade Studies - fit, balance, compromise Performance vs cost Schedule vs risk Efficiency vs flexibility Simplicity vs options Other tensions	S,K	R,Ap	R,E	R,S	
2.4. Step-System Concept definition					
2.4.1. Objectives Development Product-driven objectives Process-driven objectives			R,S	R,An	
2.4.2. Environment Definition			R,E	R,S	
2.4.3. Operations Concept Development			R,E	R,S	
2.4.4. Constraint Definition			R,E	R,S	
2.4.5. Testing Objectives			R,E	R,An	
2.4.6. Systems Engineering Process Definition		R,S	R,E	R,An	

2.5. Process Step-System Architecture Development					
2.5.1. Conceptual Architecture Development		S,C	R,E	R,S	
2.5.2. Functional Analysis and Decomposition		S,C	R,E	R,S	
2.5.3. Boundaries and Interfaces			R,E	R,S	
2.5.4. Architectural Models Hatley-Pirbhai architectural model &method			R,E	R,An	S,Ap
2.5.5. Requirements Allocation and Tracking			R,S	R,An	S,Ap
2.5.6. Architectural Standards			R,S	R,An	R,Ap
2.5.7. Testing Architecture Development			R,E	R,S	
2.6. Process Step-System Design					
2.6.1. Introduction - What is Design?	S,K	R,C	R,S	R,An	S,C
2.6.2. Design Methodologies			R,E	R,S	
2.6.2.1. Objectives and Concerns of a Design Methodology			R,E	R,S	
2.6.2.2. Dimensions of Design Methodologies			R,E	R,S	
2.6.2.3. Taxonomy of Design Methodologies Tool Kit Function oriented Design language oriented Abstract data type Object-oriented Data structure oriented Data flow oriented Control flow oriented Structure oriented Interconnection model			R,E	R,S	S,C
2.6.2.4. Methods of Distributed System Design			R,E	R,S	S,C
2.6.2.5. Methods of Concurrent System Design			R,E	R,S	S,C
2.6.3. Requirements Allocation and Tracking		R,Ap	R,S	R,An	R,Ap
2.6.4. Design Documentation			R,S	R,An	R,Ap
2.6.5. Design Standards and Rules			R,S	R,An	R,Ap
2.6.6. Testing System Design		S,C	R,E	R,S	
2.6.7. Tools - Design Aids		S,C	R,S	R,Ap	R,Ap
2.6.8. Summary					
2.7. Process Step-Managing the Life-Cycle					
2.7.1. Managing Architecture and Design Progress	S,C	R,S	S,An	S,An	
2.7.2. Integrity Maintenance - Architecture, Configuration, etc.	S,C	R,S	R,E	R,An	

2.7.3. Negotiate Changes in Requirements and Acceptance Criteria	S,Ap	R,S	S,C	S,An	
2.7.4. Acquisition and Design Issues	S,An	R,S	S,C	S,An	
2.8. Process Step-System Architecture Validation and Verification					
2.8.1. Validation - Did we build the right system?		S,C	R,E	R,An	
2.8.2. Verification - Did we build the system right?		S,C	R,S	R,An	
2.8.3. Acceptance Criteria Demonstration		S,C	R,S	R,An	
2.9. Exit Criteria Architecture and design documentation ready for implementation System architecture validation and verification complete	R,K	R,C	R,An	R,Ap	S,C

COLUMN - **STUDENT TYPE**

A: Executive Management

B: Other Management

C: Subject Matter Specialist

D: Software Practitioner

E: Support Personnel

APPLICABILITY

S: Suggested

R: Recommended

M: Mandatory

CODE

[**applicability**], **[level of learning**]

LEVEL OF LEARNING

K: Knowledge

C: Comprehension

Ap: Application

An: Analysis

S: Synthesis

E: Evaluation

Appendix C Software Architecture Development: Detailed Content Outline

Note: The numbers in braces refer to the references in Appendix D.

1. Software Systems Architecture - What is it?

1.1. Introduction

What is a software architecture? {5}
- The nature of system architectures {10}
- Assessing the architectural merits of systems {10}
- Innovative architecture {10}

Architecture and design within the system life cycle framework {10}
- Historic basis of system architecture {10}
- Computer architecture {10}
- Architectures of familiar systems {10}

Content of an architecture {1} {14}
- Identification of system elements {1}
- Identification and specification of internal and external interfaces {1}
- Each system element of a systems architecture is defined by the {1} {15}
  - Functions to be performed {1}
  - Performance requirements each function is to meet {1}
  - Physical constraints, such as weight and power consumption {1}
  - Operational environment constraints, such as temperature and pressure{1}

1.2. Architecture-Driven System Characteristics {14} {16}

Underlying axioms {14}
Modular {14} {5}
Intellectual control {14}
Conceptual integrity {14}
Open systems, portable {14} {16}
Layered protocols {16}
Common network interfaces {16}
Common user services {16}
Extensibility, malleable {14} {16}
User interface tools {16}
Common security architecture {16}
Priority, preemption {16}
Domain specific architecture {16}
Application interoperability {16}
Common data dictionary {16}
Compatible analysis tools {16}

1.3. Architectural Strategies for Software Systems {5} {4}

Data flow architectures {4} {5}
- Batch sequential {4}
- Pipes and filters, etc. {4} {6}

Communicating process architectures {5}
- Message passing {5}
- Client-server architectures {5}

Communicating sequential processes {5}
Call-and-return architectures {4}
Event-based, implicit invocation {6}
Main program and subroutines {4}
Data abstraction and information hiding {6} {4} {5}
Object-oriented organization {6} {4} {5}
Layered architectures: network protocols {5} {4} {6}
Virtual machines {4}
Table driven interpreters {4} {6}
Rule-based systems {4}
Data-centered systems (repositories) {4} {6}
Transactional databases, blackboards {4}
Heterogeneous architectures {6}
Distributed systems architectures{11}
Characteristics of distributed systems {11}
Parallel or concurrent programs {11}
Networked computing {11}
Cooperative computing {11}
Concurrent systems architectures {11}
Mixed use of idioms in software architectures {5}
Systems are rarely developed based on a single architectural idiom {4}

1.4. Standards {10}

The role of architecture standards {10}
System standards {10}
Standards used in government system acquisitions {10}
Industrial standards {10}

2. Creating the Software System Architecture - How is it done?

2.1. Introduction

Principles of good architecture development {2} {13} {15}
Consistency {2} {13} {15}
Orthogonality (elements relative independent of each other) {2} {13} {15}
Propriety (proper to functions, no unnecessary function) {2} {13} {15}
Parsimony (no functional redundancy in different forms) {2} {13} {15}
Transparency (functions introduced in implementation not imposed on user) {2} {13} {15}
Generality (multipurpose) {2} {13} {15}
Open-handedness (alternate uses of needed function) {2} {13} {15}
Completeness (in solving needs and desires of user) {2} {13} {15}
Elements of a software system architecture {10}
Decomposition {10}
Architecture and design documentation {10}
Address system requirements {10}
Interface identification and definition {10}
Internal interfaces in systems {10}
Interfaces to external systems {10}
Interfaces to system operators and other users {10}
Interface standards {10}
Architecture development process {2}
System feasibility analysis {2}
System concept definition {2}

System architecture development {2}
System design {2}
Managing the life-cycle {2}
Architecture verification and validation {2}
System validation {2}
Iterative process {2}
System integrity - requirements & configuration {10}
Requirements traceability {10}
Configuration management {10}

2.2. Entry Criteria

System requirements analysis completed {10}
Qualitative requirements analysis {10}
Quantitative requirements analysis {10}
Organizational objectives
Acceptance criteria defined {13}

2.3. Process Step-System Feasibility Analysis {2}

2.3.1. Concept Development and Analysis {2}

2.3.2. Technical Analysis {2} - Prove new architectures can work {10}

Models {13} {14}
Formal models {5}
Abstractions {14}
Stimulus-response mechanisms: models for atomic actions {14}
Context models {14}
Tasks and control processes: models for program executions {14}
Models of event systems {5}
Models for distributed concurrent design {14}
Simulating {13} {10}
Prototyping {13} {10}

2.3.3. Schedule Analysis {2}

2.3.4. Cost Analysis {2}

2.3.5. Tradeoff Analysis - Trade Studies - fit, balance, compromise {3}

Performance vs cost {3}
Schedule vs risk {3}
Efficiency vs flexibility {3}
Simplicity vs options {3}
Other tensions {3}
Function vs form {3}
System requirements vs environmental imperative {3}
Performance specs vs strict acceptance criteria {3}
Human needs vs affordability {3}
Complexity vs simplicity {3}
New technology vs familiar technology {3}
Top-down plan vs bottom-up implementation {3}
Conservative design vs risk of overdesign {3}
Continuous evolution vs product stability {3}
Minimal interfacing vs tight integration {3}
Process characterization vs process revolution {3}
Avoid complexity vs manage complexity {3}
Low-level decisions vs strict process control {3}
Specialized manufacture vs flexible manufacture {3}

2.4. Step-System Concept Definition {2}
2.4.1. Objectives Development {2}
Product-driven objectives {10}
High reliability or error-free operation {10}
System availability {10}
Maintainability {10}
Compatibility with existing systems or interfaces {10}
Ease of use {10}
System or personnel safety {10}
Long or warranted useful life {10}
Allow easy system growth or enhancement {10}
Additional product-driven objectives {10}
Process-driven objectives {10}
Cost as a design driver {10}
High-volume or specialized production {10}
Design for testing {10}
2.4.2. Environment Definition {2}
2.4.3. Operations Concept Development {2}
2.4.4. Constraint Definition {2}
2.4.5. Testing Objectives {10}
2.4.6. Systems Engineering Process Definition {2}

2.5. Process Step-System Architecture Development {2}
2.5.1. Conceptual Architecture Development {2}
2.5.2. Functional Analysis and Decomposition {2} {11}
Process communication and synchronization {11}
Process behavior {11}
Refinements of a process {11}
Problems of system composition {4}
Important ideas {4}
Common patterns for system structure {4}
Repeated decomposition {4}
Heterogeneity {4}
Independence {4}
Fit to problem {4}
2.5.3. Boundaries and Interfaces {13}
2.5.4. Architectural Models {5} {1}
Hatley-Pirbhai architectural model & method {1}
2.5.5. Requirements Allocation and Tracking {10}
2.5.6. Architectural Standards {10}
2.5.7. Testing Architecture Development

2.6. Process Step-System Design {2}
2.6.1. Introduction - What is Design? {2}
Logical design {2}
Functional design {2}
Physical design {2}

2.6.2. Design Methodologies {9}
2.6.2.1. Objectives and Concerns of a Design Methodology {9}
2.6.2.2. Dimensions of Design Methodologies {9}
Scope of applicability {9}
Information sharing {9}
Information organization {9}
Support of software life cycle {9}
2.6.2.3. Taxonomy of Design Methodologies {9}
Tool kit {9}
Function oriented {11}
Informal functional specification {11}
Predicate transformation {11}
Algebraic specification {11}
Stepwise refinement {12}
Weakest preconditions and conventional reasoning {12}
Design language oriented {9}
Abstract data type {9} {4}
Object-oriented {9} {11} {4}
Data structure oriented {11} {4}
Jackson structured programming (JSP) {11}
Warnier/Orr program design and construction method {11}
Entity relationship diagrams {11}
Data flow oriented {11} {4}
Control flow oriented {11}
Structure oriented {9}
Interconnection model {9}
2.6.2.4. Methods of Distributed System Design {14}
2.6.2.5. Methods of Concurrent System Design {14}
2.6.3. Requirements Allocation and Tracking
2.6.4. Design Documentation {10}
2.6.5. Design Standards and Rules {10}
2.6.6. Testing System Design {10}
2.6.7. Tools - Design aids {11}
Cohesion {11}
AD/Cycle {11}
SREM {11}
SDL {11}
Estelle {11}
LOTOS {11}
SADT
Hatley-Pirbhai {1}

2.7. Process Step-Managing the Life-Cycle
2.7.1. Managing Architecture and Design Progress {10}
2.7.2. Integrity Maintenance - Architecture, Configuration, etc. {2}
Architectural integrity {2}
Development process maintenance {2}
Control of system configuration data {10}
2.7.3. Negotiate Changes in Requirements and Acceptance Criteria {13}
2.7.4. Acquisition and Design Issues {10}

Subcontracting decisions: make or buy? {10}
Competitive system markets {10}
Single-buyer markets {10}
System design for speculative sale {10}

2.8. Process Step-System Architecture Validation and Verification

2.8.1. Validation - Did we build the right system? {18}

Conceptual validation {2}
Prove new architectures produce desired result {10}
Functional requirements validation {17}
Operational validation {2}
Performance requirements validation {17}

2.8.2. Verification - Did we build the system right? {17} {18}

Performance requirements verification {17}
Functional requirements verification {17}
Design requirements verification {17}

2.8.3. Acceptance Criteria Demonstration {13}

2.9. Exit Criteria

Architecture and design documentation ready for implementation
System architecture validation and verification complete

Appendix D Software Architecture Development: Sources of Outline Materials

{1}
Ramchandani, C.; Maier, M.; & McKendree, T. “Toward a Comprehensive System Architecture Representation Model,” *657-664. Proceedings of the Fifth Annual International Symposium*. St. Louis, Missouri, July 22-26, 1995. Seattle: National Council on Systems Engineering, 1995.

{2}
Smith, T.B. “Systems Architecting During the Client Interaction Cycle,” 665-673. *Proceedings of the Fifth Annual International Symposium*. St. Louis, Missouri, July 22-26, 1995. Seattle: National Council on Systems Engineering, 1995.

{3}
Rechtin, E. “Systems Architecting and Engineering,” Lecture 3, *Software Design*. Pittsburgh, Pa: Software Engineering Institute, Carnegie Mellon University, 1994.

{4}
Shaw, M. “Architectures for Software Systems,” Lecture 7, *Software Design*. Pittsburgh, Pa: Software Engineering Institute, Carnegie Mellon University, 1994.

{5}
Shaw, M.; Garlan, D.; & Galmes, J. *Experience with a Course on Architecture for Software System: Part II: Educational Materials* (CMU/SEI-94-TR-20), Pittsburgh, Pa: Software Engineering Institute, Carnegie Mellon University, 1994.

{6}
Garlan, D. & Shaw, M. *An Introduction to Software Architecture* (CMU/SEI-94-TR-21), Pittsburgh, Pa: Software Engineering Institute, Carnegie Mellon University, 1994.

{9}
Badami, S.H. “Toward A Generic Description Of Software Design Methodologies And Environments: Motivation, Issues And Form.” Clemson, S.C.: A Masters Thesis Presented to the Graduate School of Clemson University, December 1988.

{10}
Beam, W.R. *Systems Engineering: Architecture and Design*. New York, N.Y.: McGraw-Hill Publishing Company, 1990.

{11}
Fleischmann, A. *Distributed Systems Software Design and Implementation*. New York, N.Y.: Springer-Verlag, 1994.

{12}
Leathrum, J. F. *Foundations of Software Design*. Reston, Va: Reston Publishing Co., 1983.

{13}
Rechtin, E. *Systems Architecting: Creating and Building Complex Systems*. Englewood Cliffs, N.J.: Prentice Hall, 1991.

{14}
Witt, B.I.; Baker, F.T.; & Merritt, E.W. *Software Architecture and Design; Principles, Models, and Methods*. New York, N.Y.: Van Nostrand Reinhold, 1994.

{15}
Blaauw, G.A., von. “Computer Architecture.” *Electronische Reckenanlagen 14*, 4 (1972): pp 154-159.

{16}
Howe, D.M. “Information System Trends Influencing Systems Engineering,” 969-974. *Proceedings of the Fifth Annual International Symposium*. St. Louis, Missouri, July 22-26, 1995. Seattle: National Council on Systems Engineering, 1995.

{17}
Bellagamba, L., “Testing Knowledge of Systems Engineering Process Fundamentals,” 73-80. *Proceedings of the Fifth Annual International Symposium*. St. Louis, Missouri, July 22-26, 1995. Seattle: National Council on Systems Engineering, 1995.

{18}
Azzolini, J., “Essential Systems Engineering: A Lifecycle Process,” 831-838. *Proceedings of the Fifth Annual International Symposium*. St. Louis, Missouri, July 22-26, 1995. Seattle: National Council on Systems Engineering, 1995.

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION
Unclassified

1b. RESTRICTIVE MARKINGS
None

2a. SECURITY CLASSIFICATION AUTHORITY
N/A

3. DISTRIBUTION/AVAILABILITY OF REPORT
Approved for Public Release
Distribution Unlimited

2b. DECLASSIFICATION/DOWNGRADING SCHEDULE
N/A

4. PERFORMING ORGANIZATION REPORT NUMBER(S)
CMU/SEI-95-TR-010

5. MONITORING ORGANIZATION REPORT NUMBER(S)
ESC-TR-95-010

6a. NAME OF PERFORMING ORGANIZATION
Software Engineering Institute

6b. OFFICE SYMBOL (if applicable)
SEI

7a. NAME OF MONITORING ORGANIZATION
SEI Joint Program Office

6c. ADDRESS (city, state, and zip code)
Carnegie Mellon University
Pittsburgh PA 15213

7b. ADDRESS (city, state, and zip code)
HQ ESC/ENS
5 Eglin Street
Hanscom AFB, MA 01731-2116

8a. NAME OFFUNDING/SPONSORING ORGANIZATION
SEI Joint Program Office

8b. OFFICE SYMBOL (if applicable)
ESC/ENS

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
F19628-95-C-0003

8c. ADDRESS (city, state, and zip code))
Carnegie Mellon University
Pittsburgh PA 15213

10. SOURCE OF FUNDING NOS.

PROGRAM ELEMENT NO	PROJECT NO.	TASK NO	WORK UNIT NO.
63756E	N/A	N/A	N/A

11. TITLE (Include Security Classification)
Training Guidelines: Purchasing Training 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	December 1995	26

16. SUPPLEMENTARY NOTATION

17. COSATI CODES

FIELD	GROUP	SUB. GR.

18. SUBJECT TERMS (continue on reverse of necessary and identify by block number)
software architecture, software engineering, subject matter specification,training acquisition, training vendors

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

This set of training guidelines focuses on many of the issues surrounding the purchasing of software engineering training. It includes general guidelines for creating partnerships between training vendors and training purchasers to meet real training needs and a template for use by subject matter experts in elaborating training requirements.

(please turn over)

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT
UNCLASSIFIED/UNLIMITED ■ SAME AS RPT □ DTIC USERS ■

21. ABSTRACT SECURITY CLASSIFICATION
Unclassified, Unlimited Distribution

22a. NAME OF RESPONSIBLE INDIVIDUAL
Thomas R. Miller, Lt Col, USAF

22b. TELEPHONE NUMBER (include area code)
(412) 268-7631

22c. OFFICE SYMBOL
ESC/ENS (SEI)

ABSTRACT — continued from page one, block 19