Abstract
Many experienced users are faced with the challenge of developing a course to train others. Although experienced with the software, these users may be bewildered by a course development project. This paper describes a structured approach to course development, an approach similar to structured programming familiar to most programmers.

Following a course design methodology helps ensure that the training accomplishes desired outcomes. Courses written without a development strategy may (or may not) provide the desired training. These courses are frustrating to develop and frustrating to attend.

The techniques described here can be applied to the development of any training: a one-hour lesson or a three-day course, training to use a software package or improve writing skills, or training developed in different formats such as instructor-led or video-based training.

The examples and terminology used in the paper describe a course that is led by an instructor and targeted toward an audience with minimal computer usage skills. The example course teaches trainees to write simple SAS programs.

Introduction
Did your boss say something like, “You’re a natural teacher. You explain things so well to other users. We need a course to teach people how to really use the software. Why don’t you write it?”

Is that how you got your course development assignment? After that praise, did you zip into your office, sit down at the trusty word processor, and write page 1 of the course textbook? Or perhaps you remembered what Mrs. Fitzsimmons pounded into your head back in high school English class and you wrote an outline first.

How successful was the course the first time you taught it? Were you happy with the instruction? Were the students? Did you end up rewriting the course?

There are several problems with this hit-or-miss approach to course development. You do not have a known starting point. (Starting with page 1 doesn’t count!) You risk floundering in a sea of potential course topics since you have no criteria for deciding which topics belong in the course and which do not. You have no plan for how topics are sequenced. And you will probably try to incorporate everything you know into the course.

The Course Development Plan
With a structured development technique, you know where to begin, you determine course content based on new skills students should acquire from the training, and you make decisions about course structure before you begin writing. You also design courses that work the first time you teach them, you develop the course in small units or topics, and you can update topics easily when necessary.

There are five course development steps. Each of these five steps has a distinct purpose and a desired outcome.

1. Investigate needs and resources. Collect enough data about the trainees’ needs so that you can plan the course goals and objectives. Then write a student profile and the global course objective.

2. Analyze tasks to be learned. Develop skills yardsticks and use them to conduct an analysis of the tasks to be included in the course. Identify subtasks within each task to produce a task list.

3. Design course structure and content. Identify instructional strategies for each subtask. Develop instructional units (topics) from subtasks. Organize topics in an instructional tree. Verify the tree diagram with a structured walk-through.

4. Develop course materials. Write materials to guide your classroom presentation and provide notes for the students. These materials might be visual aids, a student textbook, job aids such as reference cards, or case study examples.

5. Polish course materials. Conduct a test-teach to check the design and accuracy of the course. Evaluate the effectiveness of the materials in guiding your presentation. Identify problem areas and correct them. Conduct a second test-teach if necessary. Publish the course materials.

Investigate Needs and Resources
During investigation, you
• conduct needs assessment
• conduct research
• define your audience
• define the global course objective.

These activities help you determine the course scope and audience.

Conduct needs assessment
Needs assessment identifies whether a training need does indeed exist, who should receive the training, and what the training should include. During this assessment phase, interview the person(s) requesting the course, users of the software that is the subject of the training, future users of the software (potential course attendees), and supervisors of those users. You can
interview people in person or over the telephone, send out questionnaires, and observe users while they work with the software.

Too often, a course developer is given an assignment to develop a course that is not really needed. Ask questions such as these to determine the need for the course:

- Who asked that the course be developed?
- Why does this person think a course is needed?
- What problem should the training solve?
- What new product or system is being installed that generates the training need?
- Are there other training resources to meet the need?
- Is more than one course necessary?

Determine the potential audience for the course. The person requesting the course may have a particular audience in mind. There may also be others who would benefit from the course. Locate people who use the software. Some of these people may need additional training, or there may be new employees who need training.

Find others who do not currently use the software but would benefit from using it. Ask why they do not use the product. Interview potential students to learn what they want to be able to do after they have completed the training. Find out what skills they already have that can be applied to using the software.

Interview supervisors of potential students. The supervisor knows what new skills the employee should acquire from the training and has specific departmental objectives that require use of the software.

If you are developing a course that is part of a larger curriculum, consider where the new course fits into that curriculum. What courses exist that provide appropriate prerequisite skills? What courses provide follow-up training?

As a result of needs assessment, you confirm that there is a need for the course, identify the intended audience, determine the skill levels of the trainees — both before the course and after — and focus on the specific tasks to be included in the training.

Conduct research

You need to discover what resources are available and what constraints control the course development project. Check to see if the training is already available. You may find that it is offered by the software vendor or a training vendor. Even if it is not cost-effective for your company to use vendor training, you may find the vendor’s course descriptions helpful.

Determine where the course will be taught. Is the course always offered in the same facility, or must the course be transportable to other facilities that may have different equipment? Decide what media will be used to present the course. Determine the time factors that affect course development and delivery.

Consider course materials such as student textbooks, reference cards, and instructor’s guides. What materials are you expected to produce? How will the materials be prepared and printed? What support services, such as clerical help, editing, and graphic design, are available to you?

Collect as much information as possible related to the course subject. Look for user guides that show how to use the software, reference manuals that describe the rules of the language, primers or introductory guides designed for beginning users, and reference cards.

Find a subject matter expert. The expert should have credibility in the subject matter, enthusiasm for the subject, and a desire to involve others in the subject. Select someone who knows a little bit more about the subject than what is intended to be covered in the course.

Perhaps you are the expert and were selected for this course development assignment because of your expertise and communication skills. If possible, locate another expert for advice and review.

Talk to current users of the software. Watch these users work with the software and find out how it applies to the performance of their jobs. Ask how these users learned the software. Find out what resources (manuals, other users, and so on) they consider valuable. Check to see if there are trouble spots that new users should be warned about. Make sure you talk with users whose skill level is similar to the post-course skill level you plan for the course.

Talk with potential users of the software. Find out what they expect from the course. Ask how they expect to use new skills in performing their jobs.

Select an advisory team. Although you have the major responsibility for course development, you need a team to objectively review your plans and materials and to advise you. The team should consist of one or more subject matter experts, other trainers, and an editor.

Define your audience

Once the research is completed, define the course audience. Put your description in writing so you can refer to it later. Use job titles specific to your company or industry, note the general skill level trainees should have acquired prior to the course, and list overall benefits trainees gain.

Avoid trying to create one course for everybody. In that case, some trainees are bored and others struggle to keep pace with the class. Define a single audience for the course. If a different audience needs similar training, develop a separate course.

See figure 1 for a sample audience description.

This course is designed for users with limited computer experience who want to analyze data and produce reports using the SAS System.

Figure 1. Sample Audience Description
Define the global course objective

The global objective is a description of what the student will be able to do upon successful completion of the instruction. Objectives must be measurable and observable. Write objectives that describe actions, using verbs such as generate, issue, list, print, or submit. Avoid using words that do not describe measurable actions, such as understand, comprehend, or know.

Make a single statement of new skills or benefits students gain from the course. Use terminology appropriate for someone considering attending the course.

See figure 2 for a sample global course objective.

After completing this course, you will be able to

- use the SAS System to read a data file that is column-aligned
- create a SAS data set
- produce these reports:
  - a column-style listing of the data
  - a sorted listing
  - a report of the average, total, minimum, and maximum values for each numeric variable in the data set
  - a count of the number of times each different value occurs for a given variable.

Figure 2. Sample Global Objective

The result of the investigation step is the written audience description and the global objective. With this information you move to the next development step.

Analyze Tasks to be Learned

During this step you

- define skills yardsticks
- conduct a task analysis.

These activities help you define the content or scope of the course.

Define skills yardsticks

There are two yardsticks you must develop to evaluate course topics: one measures pre-course skills and the other, post-course skills. The pre-course yardstick defines skills students should have prior to the course. The post-course yardstick describes skills students will have acquired upon completion of the course. The course goal is to provide the training required to step the student from the pre-course level to the desired post-course level.

To ensure the training meets this goal, you measure potential tasks to be taught against these yardsticks. If the task is a pre-course skill, it should not be included in the training. Likewise, if the task is beyond the post-course skills, it is not included.

For your pre-course skills yardstick, write a detailed list of skills students must have before entering the course. Be realistic and consider how students will have acquired those skills. If you are creating a course that is part of a curriculum, consider skills students gain in courses preliminary to the one you are developing.

Make certain that you list skills you can observe or test. If you list pre-course skills as measurable activities, it is easy for potential trainees to evaluate their readiness for the course.

See figure 3 for a sample pre-course skills yardstick.

Prior to taking this introductory SAS software training course, you should be able to

- log on and off the computer system
- use the computer terminal and keyboard to enter commands or information
- use a text editor to enter and edit data in an existing file.

Figure 3. Sample Pre-Course Skills

Use the global objective to write a list of new skills students acquire during the course. The list of post-course skills is more detailed than what you indicate in the global objective.

See figure 4 for a sample post-course skills yardstick.

Upon completion of this introductory course, you will be able to

- invoke and exit the SAS Display Manager System
- use display manager to enter, modify, and submit program code
- write and execute SAS jobs to solve simple data analysis problems
- create a SAS data set from an external data file that is column-aligned
- sort and print the information in the data set
- determine the minimum, maximum, average, and total values for each numeric variable in the data set
- count the number of occurrences of each data value for a given variable
- detect and correct errors reported on the SAS log
- print hardcopy of your reports
- enhance reports with titles
- save a SAS program from display manager to a disk file and include a program from a disk file to display manager.

Figure 4. Sample Post-Course Skills
Conduct a task analysis

A task analysis helps you define the learning that is to take place and limit the instruction to that which is needed. In task analysis, you recognize the present skill level of the trainees, determine what skills must be gained to raise that level, and focus on the sequence of what is to be learned.

A task is a series of actions that accomplishes a goal. A task is performed in an observable manner; that is, something happens or is changed as a result of the task. In task analysis, you create a list of tasks trainees should be able to perform at the conclusion of the training.

To begin, brainstorm with the subject matter expert. List all tasks that are (or seem to be) between the pre-course and post-course skills yardsticks. Also, consider the global objective and the tasks necessary to meet that objective. During this brainstorming session, do not take time to filter information or to evaluate tasks as you add them to the list. At this point, the important thing is to create the list. List tasks that describe actions, using words such as exit, interpret, invoke, issue, and so on.

When you have listed all tasks, evaluate each one. Compare each task to the skills yardsticks. If the task is defined in the pre-course yardstick, remove it from the list; it should not be taught in your course. If the task is beyond the scope defined by the post-course yardstick, drop it from the list.

Once you have compared the task list to the skills yardsticks, there is one final measure to apply: need-to-know or nice-to-know. Consider whether students with appropriate prerequisite skills need to know each task in order to meet the post-course skills. If not, strike the task from the list. You do not want to spend valuable class time on topics that are simply nice to know.

Beware — a task can be nice-to-know in one course for one audience and need-to-know in a similar course for a different audience. For example, a need-to-know task in an introductory course for programmers is to describe how the SAS System constructs its program data vector and uses it to read and write SAS data sets. This task is only nice-to-know in an introductory course for end users.

When you have evaluated all tasks, the list defines the scope of the course. Each task, however, consists of multiple subtasks. To define course topics, you must identify the subtasks.

Consider each task and define its subtasks. Think of the steps a user must go through to accomplish the task. You can also observe someone performing the task and write down each step necessary to perform it.

Evaluate subtasks by applying the same measures: the pre- and post-course skills, and need-to-know or nice-to-know.

See figure 5 for a sample task list.

Objective: Create a SAS data set from a raw data file

Tasks:
- locate a record layout for the file (pre-course skill)
- log on to the computer (pre-course skill)
- invoke display manager
- assign a fileref to the raw data file with a system command
- begin a DATA step
- point to the external file with the fileref
- use the record layout to design an INPUT statement to read data
- read the raw data with column input techniques
- describe how the SAS System creates an internal program data vector (nice-to-know)
- mark the end of the DATA step
- submit the step for execution
- review the SAS log and interpret messages, including errors
- recall program code and correct errors
- exit display manager
- log off the computer (pre-course skill).

Figure 5. Sample Task List

When you have evaluated all tasks, the list defines the scope of the course. Each task, however, consists of multiple subtasks. To define course topics, you must identify the subtasks.

Consider each task and define its subtasks. Think of the steps a user must go through to accomplish the task. You can also observe someone performing the task and write down each step necessary to perform it.

Evaluate subtasks by applying the same measures: the pre- and post-course skills, and need-to-know or nice-to-know.

See figure 6 for a sample subtask list.

Task: Begin a DATA step

Subtasks:
- define purpose of DATA step
- select data set name following syntax rules
- write a DATA statement and name the data set.

Task: Point to an external data file

Subtasks:
- remember fileref previously assigned with system command
- write an INFILE statement
- use correct syntax.

Figure 6. Sample Subtask List

See figure 5 for a sample task list.
The result of the analysis step is the course content, listed by task and subtask. Note that the content is probably arranged in the sequence tasks are performed on the job, which is not necessarily an appropriate teaching sequence. At this point you know what the course should teach, but not how. Decisions about how to organize the course content are made in the next step.

Design Course Structure and Content

Next you finalize course topics and sequence. This step consists of three phases:

- define course content
- organize teaching sequence
- verify the course design.

The result is a schematic tree diagram of the entire course.

Define course content

Courses are a series of topics, strung together in a sequence that is logical to the learner. A topic is the smallest unit of instruction, a single discussion to convey a point.

Related topics are grouped and presented as an instructional unit, often called a section, segment, or lesson. (This paper uses the term section.) If necessary, related sections can be grouped into chapters or modules. (This paper uses the term chapter.)

To make the transition from the subtask list to course topics, you must identify instructional strategies for each subtask. Before you show students how to perform a subtask, you have to build a conceptual framework for the instruction. In other words, you have to prepare the student to receive the instruction.

Instruction about subtasks has two stages: enabling and performance. The enabling stage prepares the student with facts, figures, terms, and concepts. In the performance stage, you illustrate how to accomplish the subtask by presenting rules, discussing procedures, or demonstrating processes.

Identify course topics that you can use to present each subtask. Remember to include topics that enable students to receive the instruction. You may notice that the same topic is included in several subtasks. Make a note each time the topic occurs since you do not yet know the order in which topics will be presented.

See figure 7 for a sample of topics derived from subtasks. Notice that terminology and concepts are enabling topics that should be presented prior to rules, a performance topic.

Subtask: Use correct syntax

Topics:
- terminology — define syntax
- concept — why correct syntax is necessary to communicate with the software
- rules — end statements with a semicolon, how to name variables, and so on.

Figure 7. Sample Topics from a Subtask

Continue identifying course topics for each subtask on your task list. When you have completed this phase, you have defined the course content, but not the flow or sequence of presentation.

Organize teaching sequence

For a group of course topics, numerous teaching sequences can be designed. Some topics build on others and must be presented in sequence. For example, the definition and concept of syntax should be discussed before you state the syntax rules. Other topics, such as the purpose of the CONTENTS and MEANS procedures, are independent and can be presented in any convenient sequence.

To develop a sequence, arrange your task lists into a logical sequence for presentation. Look through the topics for overlap. You want to present a topic as early as necessary and avoid too much repetition. Find the first instance that a topic must be presented. After that, check off the topic whenever it occurs on the task list (or jot an R beside it if you feel a brief review might be necessary).

Look for topics that are closely related. Cluster these at an appropriate point on the task list.

Next look through the topics and consider teaching them in the sequence they are listed on the task list. Rearrange if the order of topics seems inappropriate. Continue until you are satisfied with the sequence of topics.

There are many strategies for sequencing topics. Students find learning easiest when information is presented in simple-to-complex order and in a logical order with one topic leading directly to the next.

You may employ different instructional methods effectively. You can teach in the order that tasks are performed (create a data set and print it). Show a task and then its opposite or counterpart (how to log on and how to log off). Illustrate simple tasks before complex ones (how to read column-aligned data before formatted data). Move from familiar to unknown (the purpose of the SORT procedure is fairly obvious, but the purpose of the FREQ procedure may not be). You can also organize topics to give students a quick sense of accomplishment (have them use procedures on existing data sets before you present how to create new data sets).
Most course topics are grouped into higher structural categories: sections and chapters. Use a tree diagram to structure the topics. Design the tree from the bottom up: cluster topics into sections, and sections into chapters.

To design your tree, read through the topics, starting at the beginning. Look for the first logical conclusion or break point. Group topics from the beginning to the break point as a section. Sketch a diagram of the section, using an appropriate section title as the top node of the tree. If you have difficulty selecting a section title that seems appropriate for all topics, reconsider the topics you have clustered. You may need to move some topics into other sections.

Define the next section, working from the first break point to the next logical conclusion. Name and diagram this section. Continue until all topics are clustered into sections.

Review each section. Does each section have a beginning and a logical conclusion? Consider the time that you think will be required to present the material. Most courses allow breaks every hour or so. Can you present the section in that time frame? If not, try to subdivide it into smaller sections.

After you identify and review the sections, cluster them into chapters. A chapter often presents one task or several related tasks. Determine logical groupings of sections into chapters and name the chapters. Add the chapter names as the highest nodes of the tree diagram.

Connect the chapters on the tree to one top node that contains the course title. The tree now defines the course structure.

See figure 8 for a sample tree diagram.

Chapter 5 Running Programs Interactively

- Section 5.1 Submitting programs
  - Topics:
    - show how to issue SUBMIT command
    - discuss alternate method of submitting with function key
    - define function of output screen
    - note function keys have new definitions since output screen is active
    - discuss how to scroll output and move to top or bottom
    - show how to end from output screen and return to program editor and log screens.

- Section 5.2 Reviewing results

Figure 8. Sample Tree Diagram (Partial)

Verify the tree diagram

The tree structure shows you visually the entire course and helps you work with the overall structure and content. By tracing paths on the tree, you can discover overlaps and uncover gaps. It is easy to determine where to add a new topic or move a topic you feel is misplaced.

Look at the tree diagram and trace each branch from the top node (the course title) to the bottom (topics). Think of the topics along the branch as stepping stones across a river. At the beginning, the student is on one side of the river. At the end, you want the student on the other side.

Does the sequence of topics lead the student directly across the river? Are there gaps in the information? Is there too much overlap so that you present a topic more than once? If you find topics that are out of place, move them. Add or delete topics as necessary to provide a smooth path across that river.

Conduct a structured walk-through with your advisory team. Send copies of the audience description, the global objective, the skills yardsticks, the task/subtask list, and the tree diagram to the team members. Call a meeting to review the course structure and content.

During the meeting focus on problems, not solutions. Trace the intended flow of the course using the tree diagram. Note problems with structure, omissions, overlaps, and so on. Do not attempt to solve these problems during the meeting. That is the course developer's job after the meeting is over.

Once any problems are resolved (and the team reviews the course again if necessary), you are ready to write course materials from the tree diagram.
Develop Course Materials

With the preliminary work complete, you can confidently produce course materials. In this step you
- write course materials
- review course materials.

Write course materials

It is easy to write materials using the tree as a map. You can work from top to bottom, in sequence from left to right. Or you can develop materials working in a random order, writing what you feel like writing, and checking off topics until you are finished.

Develop instructional materials to correspond to the lowest levels of the tree, the topics. Write materials for the higher nodes of the tree (sections and chapters) to connect topics and help guide the student. Provide introductory lead-ins at the beginning of each chapter and each section to tell students where the course is headed.

Include summaries at the end of sections or chapters to remind students what was covered and relate new material to that previously covered. Help students organize their learning by providing checkpoints along the way. One instructional design axiom is

Tell them what you are going to tell them, tell them, tell them what you told them.

In other words, prepare, present, and summarize.

Review your materials

After writing the materials, review and edit them several times. Each time, focus on different aspects of the course. On the first pass, focus on the overall design and organization of the course. Do the topics flow smoothly from one to the next?

On the second pass, concentrate on each chapter. Are the topics in that chapter logically related? Consider the time frame. How long will it take to present the chapter?

Finally, review each topic as a single entity. Check the materials closely for spelling, grammar, completeness, and so on. Make any necessary corrections.

When you are satisfied with the materials, have the advisory team review them. The subject matter expert concentrates on technical accuracy, the trainer looks for instructional quality, and the editor checks for grammar and formatting consistency.

Produce a few copies of the course materials for the final test, the classroom.

Polish Course Materials

During the last step of course development you
- conduct a test-teach
- finalize course materials.

Conduct a test-teach

A test-teach helps you find any problems with the course design or materials before you sign off on the project. To hold the test-teach, gather five to ten students who meet the course prerequisites and who are willing to be guinea pigs. Also include a subject matter expert and a trainer.

Conduct the test-teach as a standard course with a few differences. Ask students to evaluate the course often, perhaps at the end of each chapter. Ask someone to record the beginning and ending time for each section. Have someone record questions that are asked and make note of any course topics that seem especially confusing.

Be sure that you use class time to discuss questions related to the course subject. Discuss questions related to course design at the end of the session.

Ask attendees to complete an overall evaluation at the course conclusion. Be sure to make your own course evaluation.

Finalize course materials

Based on student comments and your evaluation, revise the course as necessary. Because of the careful planning that went into the course design, you should not have extensive revisions.

Often the test-teach reveals that you have too much material for the time allotted. To help you decide what material to cut (if necessary), re-apply the need-to-know or nice-to-know measure to course topics.

Avoid falling into the trap of perfectionism. Revise the materials, check your revisions, and print appropriate quantities of the materials. Then sign off the project.

Conclusion

Course development can be a frustrating assignment, or a rewarding one. Experienced users are often asked to create courses, yet have no formal instructional design training and may feel overwhelmed by the magnitude of the assignment.

This structured approach to course development divides the project into five manageable steps. Each step has an explicit purpose and a desired outcome. The result of one step becomes input to the next. Reviews are included so that design problems are uncovered early and corrected. The result is less frustration and better courses.

Improving the process improves the product.

SAS is a registered trademark of SAS Institute Inc., Cary, NC, USA.