The Documentation Diet

Neil Potter and Mary Sakry
The Process Group

We hate to do project and process documentation and are upset when it only seems like irrelevant paperwork that is done to please some external party. However, when we do not have any documentation, we can feel left in the dark because of a lack of direction and clear communication among colleagues. If your project is experiencing either too much or too little documentation, or your company improvement program is really no more than a documentation exercise, there is an effective middle ground; one that allows you to be productive in your projects, but not taxed with irrelevancy. This article discusses strategies you can use to make documentation work for you. The examples contain a mix of project and process documentation scenarios.

During the 15 years we have been observing companies and helping them improve, a common cause of irrelevant or overwhelming stacks of paper has been a lack of purpose or objective for each document. When we ask the average project team member why he or she has so much documentation, a common answer is, "Because my organization requires me to fill out the templates." If we ask about the usefulness of such documentation, a typical reply is, "I guess it will provide a trail of what has happened so that management can study my project later."

In hindsight, few people ever go back and plow through *the stack*. With purposes as unclear as these, it is not surprising that people merely *fill out the templates*. What is the purpose of documentation? Here are two examples:

- Project documentation is a method of capturing and sharing critical project concepts, plans, and information as they are developed so that impacted parties can share this information, make informed decisions, and move the project forward.
- Process documentation is a method of capturing and sharing engineering and management practices so that an organization can remember, reuse, and refine its skills and avoid reinventing lessons learned and best practices for each new project. Process documentation can include templates, procedures, and checklists.

Note that we did not include in our definition "evidence to please managers or auditors." Defining the information required to manage a project effectively creates *natural* documents that provide ample evidence of certain practices occurring. For example, if we plan a project correctly and capture the details so they can be communicated to others, the natural document that results (the plan) should be ample evidence that planning took place. Evidence is free when good practices are followed.

Strategies for Making Documentation Practical

There are several techniques to make documentation concise and practical. They include the following:

> *"Defining the information required to manage a project effectively creates natural documents that provide ample evidence of certain practices occurring ... Evidence is free when good practices are followed."*

- Focus documentation on the organization's needs.
- Merge duplicate work products.
- Remove redundancy in templates.
- Simplify "best practice" (process) documentation.
- Consider one representation.
- Consider one page per process or sub-process.
- Merge documented procedures and related work product templates.
- Use process descriptions as audit checklists.

The following sections describe these techniques in greater detail.

Focus Documentation on the Organization's Needs

Answer the following questions to determine the organization's needs for each document:

- What goal are you trying to achieve; what role does this document play with respect to this goal?
- What problem are you trying to solve with this document?

These questions cause you to focus on the specific purpose of each document; your responses scope the document and provide you with an end point.

In one software development group, 50 percent of each requirements document contained information describing how the product was going to be built instead of focusing on what the product was going to do for the end user. The lack of a clear goal allowed the specification to become a *catchall* document with no end point. The following is an example of a goal for a requirements document:

> Capture the needs of our customers by defining the tasks they need to perform and expectations they must have met in the solution we deliver (e.g., performance and reliability targets).

An example of a requirements template that captures user needs and other expectations is shown in Figure 1 (see page 22) [1].

Merge Duplicate Work Products

When project documents contain similar information and there is little benefit in keeping them separate, consider merging them. For example, if there are three documents to complete – *Statement of Work*, *Product Requirements*, and *Contractual Requirements* – and each will contain the same information, consider one docu-

Requirements Template	
Instructions for Template Use ~~~~~~~~~~~~~~~~~~~~~~~~~ ~~~~~~~~~~~~~~~~~~~~~~~~~ ~~~~~~~~~~~~~~~~~~~~~~~~~	4. External Interface Requirements 4.1 User Interfaces 4.2 Hardware Interfaces 4.3 Software Interfaces 4.4 Communications Interfaces
1. Introduction 1.1 Purpose 1.2 Intended Audience 2. Business Requirements 2.1 Business Requirements 1.N 2.2 Product Scope (Context Diagram) 3. Overall Description 3.1 User Classes and Characteristics 3.2 Operating Environment 3.3 Assumptions and Dependencies	5. Use Cases (UC) 5.1 UC Name 1 5.1.1 UC Details 5.1.2 Business Rules For UC 1 5.1.3 Functional Requirements for UC 1 5.x UC Name x 6. Other Nonfunctional Requirements 6.1 Performance and Reliability Requirements 6.2 Business Rules (Global) Appendix A: Glossary Appendix B: To Be Determined List

Note: Adapted from IEEE Standard 830-1998

Figure 1: *An Example Requirements Template*

ment. In the document, cross-reference the other two templates that this document satisfies. If there are differences in the three documents but considerable overlap, write one set of requirements and label those items that are *Statement of Work* as deliverables, and those that are *Contractual* as requirements.

If you are using an improvement framework such as the Software Engineering Institute's (SEI) Capability Maturity Model® (CMM®), consider merging work products together to implement specific practices. For example, a *Software Configuration Management* (SCM) plan, *Software Quality Assurance* (SQA) plan and *Software Development Plan* (SDP) can be merged. Milestones and activities for SCM and SQA might be listed on the master schedule in the SDP.

If your project is required to write maintenance documentation so future teams can understand the program's internal workings, consider the overlap between this manual and the other work products that have been created. A current design document already states the architecture, data flow, algorithms, methods and interfaces, so there should be no need to duplicate this information in a maintenance manual.

A current requirements document states the functionality and behavioral characteristics of the product, so the maintenance manual should not need to repeat this information either. Determine what unique information is needed for a maintenance person and scope the document to just this need. Cross reference existing work products to make the maintenance manual complete.

Remove Redundancy in Templates

Closely examine sections within each template that are redundant. The template might have looked sound when first created, but during use you might find that some of the sections contain the same information. Each use of the template is an opportunity to put it on a diet. For example, the requirements template on the left in Table 1 can be slimmed down to the template on the right in Table 1 when it is realized that everything said in sections 1 and 4 have already been said in sections 2 and 3.

Figure 2 describes an example of a design template. In this example, items 5.6 and 5.7 are potentially overlapping and could be merged.

Simplify Best Practice (Process) Documentation

Process documents can also suffer from a lack of purpose clarity. For example, suppose you are on the Best Practices Definition committee, or your group is using the SEI CMM, and you have been chartered to develop a process for creating project schedules [2]. You might be tempted to build the world's greatest and most comprehensive schedule creation process, with all known bells and whistles. The document could restate the CMM text, include references to numerous books on the subject, and refer to Critical Chain Analysis (whatever that is!). The appendix could include three pages of cross-references to other models and standards.

Alternatively, ask the first question, "What goal are you trying to achieve; what role does this document play with respect to this goal?" For example, the goal could be to determine which product features could be completed by the established delivery deadline given the available resources. This process describes how to develop a schedule to help achieve that goal.

The second question, "What problem are you trying to solve with this document?" enumerates needs for the document to address. An example of a need is to prevent your project from chronically over-committing, causing financial loss to the company. Now write a small process to accomplish these two items. An example is shown in Table 2.

When do you stop defining this process? Stop when your goal has been achieved (e.g., scoping the project) and your problem solved (e.g., avoiding over-commitment). Refine the document further when it no longer meets the need.

Consider One Representation

Write processes using one representation. For example, if you are creating a process for risk management, it would be redundant to have one file of presentation slides, the same process formatted using a word processor, a version in HTML for

Table 1: *Redundant Template Sections Are Removed*

Requirements Specifications	Revised Requirements Specifications
1. Product Objectives 2. Business Requirements 3. Product Advantages 4. Value Proposition	1. Business Requirements 2. Product Advantages

Figure 2: *Example Design Template*

Design Template	
Instructions for Template Use ~~~~~~~~~~~~~~~~~~~~~~~~~ ~~~~~~~~~~~~~~~~~~~~~~~~~ ~~~~~~~~~~~~~~~~~~~~~~~~~	4.0 Software Architecture 5.0 Module Attributes 5.1 Module purpose 5.2 Performance requirements 5.3 Communication protocol 5.4 User interface (if applicable) 5.5 Local data structures 5.6 Algorithm 5.7 Flow diagram 5.8 Error handling
1.0 Purpose 2.0 Global Data Dictionary 3.0 Design Method 3.1 Function-oriented design 3.2 Data-oriented design 3.3 Real-time control-oriented design 3.4 Object-oriented design	

® Capability Maturity Model and CMM are registered in the U.S. Patent and Trademark Office.

browsing, and the same information again using a flow diagramming tool.

Instead, determine how the process document will be used (e.g., online use by developers during project execution, or in a classroom setting with 100 people being trained). Then consider one representation that can suit all needs. For example, a presentation slide format can be printed for reading, e-mailed for sharing, presented for teaching, and uploaded for browsing.

Consider One Page Per Process or Sub-Process

There are approximately 60 lines on a page and 10 words per line. That is quite a lot of information. So consider keeping process documentation to one or two pages (at least at the beginning).

Processes can be kept to one or two pages by limiting how much detail you allow yourself to write. Unless you plan on writing forever, you have to put some limit on the document, so start with one page. When you are tempted to add more explanation and detail, refine what you have defined; do not necessarily add more sections.

Merge Documented Procedures and Related Work Product Templates

Organizations using process improvement frameworks such as the SEI CMM and ISO 9001 might be tempted to write procedures *because the framework states that they are needed*. Creating a template to assist the procedure user (for example, a template for an SCM, SQA, or project plan) often follows procedure creation. Creating both a procedure and template can lead to redundancy. An alternative approach is to embed the instructions for completing a template in the template itself. The procedure and the template are the same document. For example, the CMM practice "Create an SCM plan according to a documented procedure," can be implemented by developing a lightweight template with embedded instructions for use (see Figure 3).

Use Process Descriptions as Audit Checklists

If you have a process assurance function that audits projects for process compliance, use the process descriptions that the projects use; do not write a separate audit checklist. Write processes (for example, estimation, schedule creation, and change control) in a style that can be used for both project and audit purposes. It might be necessary to provide auditors with some additional guidance in conducting the audit and reporting the results. It is unnecessary to duplicate the same process information in a different format.

Schedule Creation Process to Scope a Project and Avoid Financial Loss Due to Over-Commitment

1. Determine project tasks.
2. Determine project task dependencies:
   a. For each pair of tasks (A+B), ask, "Must task A complete before task B starts, or can both tasks execute in parallel?"
   b. Draw dependency between tasks.
3. Add effort estimates for each task (uninterrupted time).
4. Add resources to each task (people, equipment, resource assumptions).
5. Add resource availability, i.e.:
   a. Planned percentage each resource will be allocated.
   b. The dates each resource is available.
6. Overlay desired project completion deadline. If the deadline is impossible, ask:
   a. What features fit within the deadline? Is this a satisfactory list?
   b. What options are available to achieve the deadline (e.g., make/buy tradeoffs, adding resources to the critical path, simplifying features, subcontracting work out, reusing existing code).
   c. What features should be demoted for later release?
7. Present data, schedule options, and risks to management and the customer. Agree on a schedule that has acceptable customer satisfaction and acceptable risk of failure.

Table 2: *A Schedule Creation Process*

Knowing When You Are in Trouble

An organization is in *document trouble* when project team members create documents that have little use or value. This can occur when either the team members are unclear about a document's purpose or when a document is created to satisfy the needs of an external auditor or assessor.

> *"An organization is in document trouble when project team members create documents that have little use or value."*

In the first scenario, a committee is typically formed to define a specific phase of the software life cycle. The template is the committee's deliverable. The template is successfully used on a few projects and is then made standard operating procedure. When the template contains more sections than needed, and when the larger audience is not trained in the template's purpose, too many project teams fill out the template with redundant information. At this point, the resulting document can be viewed as unnecessary.

In the second scenario, project team members believe they have to create additional documentation to prove to an external auditor or assessor that the project is being managed correctly. Memos capturing meeting discussions, statements-of-work documents summarizing product requirements, and design documents that are created after the code has shipped are produced to *pass the audit*. The team views documentation as an activity unrelated to building the product. It is a *keep management happy* tax.

In this second scenario, the cause can be due to poorly trained auditors who look for *paperwork* but do not really understand the fundamental engineering or management practices that are desired of the project teams. For example, the auditor looks for a statement-of-work document even though a detailed set of requirements exists that covers the same information. Minutes of meetings are examined even though the project is six months behind and none of the corrective actions during those six months have been implemented. Here, the auditor needs education, and the documents required of the team need tailoring.

Pleasing an auditor can also occur when the project team members have not ana-

Figure 3: *Combined SCM Plan Creation Procedure and Template*

SCM Plan Template	Instructions
Step 1: List Configuration Items – x, y, z	~~~~~~~~~~~~~~~~~~~ ~~~~~~~~~~~~~~~~~
Step 2: Establish File Naming Conventions – File-x<n>.doc	~~~~~~~~~~~~~~~~~~~ ~~~~~~~~~~~~~~~~~
Step 3: Establish Baseline File Structure – ~~~~~~~~~~	~~~~~~~~~~~~~~~~~~~ ~~~~~~~~~~~~~~~~~

lyzed why an engineering or management practice (and its associated document) is required, and how they could benefit from it. In such cases, the team *reacts* to the process requirement without understanding why the requirement is there. Here, the team needs training on the purpose and correct use of each specific document. If the project is performing the required practices correctly, the natural documents produced should be essential for team operation and be adequate for any auditor or assessor. The goal is no *extra paperwork*.

Summary

Software development is not about documentation. Software development is about creating solutions that help meet customer needs. Process improvement is not about documentation. Process improvement is about fixing critical problems in the organization and capturing the solutions for reuse and refinement.

Excess documentation is often the result of poor clarity of purpose and inadequate understanding of how each document should be used. When documents are written with a clear business goal and need in mind, they become important and useful.◆

References

1. Institute of Electrical and Electronics Engineers. Recommended Practice for Software Requirements Specifications. Piscataway, NJ: IEEE Computer Society Press, 1998 <http://shop.ieee.org/store/product.asp?prodno=SH94654>.
2. Software Engineering Institute's Capability Maturity Model (V1.1) Software Project Planning Activity 12.

Additional Reading

1. Potter, N., and M. Sakry. Making Process Improvement Work – A Concise Action Guide for Software Managers and Practitioners. ISBN 0-201-77577-8. Reading, MA: Addison-Wesley, 2002.
2. Paulk, Mark C., Charles V. Weber, and Bill Curtis. The Capability Maturity Model: Guidelines for Improving the Software Process. Reading, MA: Addison-Wesley, 1995.

About the Authors

Neil Potter is co-founder of The Process Group, a software-engineering process improvement consultancy. He has 19 years of experience in software and process engineering. For six years, Potter was a software engineer at Texas Instruments (TI). For two years, he managed a TI Software Engineering Process Group spanning America, England, and India. Potter is a Software Engineering Institute-authorized assessor for CBA-IPI and SCAMPI℠ appraisals and an "Intro to CMMI" instructor. He has a Bachelor of Science in computer science from the University of Essex (United Kingdom) and is co-author of "Making Process Improvement Work – A Concise Action Guide for Software Managers and Practitioners."

The Process Group
P.O. Box 700012
Dallas, TX 75370-0012
Phone: (972) 418-9541
Fax: (972) 618-6283
E-mail: help@processgroup.com

Mary Sakry is co-founder of The Process Group, a software-engineering process improvement consultancy. She has 28 years of experience in software and process engineering. For 15 years she was a manager and software engineer at Texas Instruments (TI). In 1989, she led TI's worldwide process assessment effort and was a member of a Software Engineering Process Group. Sakry is a Software Engineering Institute-authorized assessor for CBA-IPI and SCAMPI℠ appraisals and an "Intro to CMMI" instructor. She has Bachelor of Science in computer science and a master's degree in business administration. She is co-author of "Making Process Improvement Work – A Concise Action Guide for Software Managers and Practitioners."

The Process Group
P.O. Box 700012
Dallas, TX 75370-0012
Phone: (972) 418-9541
Fax: (972) 618-6283
E-mail: help@processgroup.com