ABSTRACT

WORKING AS AN AGENT OF CHANGE: WRITING RAPIDLY AND ESTABLISHING STANDARDS IN WEB SOFTWARE DOCUMENTATION

by Sarah Elizabeth Burke

This report discusses my internship experiences at Fig Leaf Software in Washington, DC, where I worked as a technical writer during the summer of 2001. In the report, I describe the young, rapid-development environment in which I worked, my major tasks and projects, and a significant project that I completed during my internship.

During this project, I faced many challenges in developing the company’s first client installation guide, including staying within the allotted hours and budget, gaining access to technical information, and establishing standards for a new document type. After discussing these challenges, I examine my role and value as an agent of change at Fig Leaf Software and present an expanded organizational role for technical communication practitioners.
WORKING AS AN AGENT OF CHANGE: WRITING RAPIDLY AND
ESTABLISHING STANDARDS
IN WEB SOFTWARE DOCUMENTATION

An Internship

Submitted to the
Faculty of Miami University
in partial fulfillment of
the requirements for the degree of
Master of Technical and Scientific Communication
Department of English
by
Sarah Elizabeth Burke
Miami University
Oxford, Ohio
2003

Advisor ________________________________
Dr. Katherine T. Durack

Reader ________________________________
Susan E. Gertz

Reader ________________________________
Dr. Jean A. Lutz

Reader ________________________________
Dr. LuMing R. Mao
## Contents

### Chapter 1: Introduction

- Fig Leaf’s History, Organization, and Culture .................................................. 1
- About the Technical Writing Team ........................................................................ 3

### Chapter 2: Diverse Projects, Common Solutions ........................................... 5

- Two Significant Projects ....................................................................................... 6
  - Project 1: Compose Specification Process Guide .............................................. 6
  - Project 2: Create Instructional Text for an Application .................................... 8
- Common Solutions .................................................................................................. 11

### Chapter 3: Standard-Setting Project: Developing a Client Installation Guide .... 13

- Overview and Phases of the FLS Documentation Process .................................... 13
  - Phase 1: Planning ............................................................................................... 14
  - Phase 2: Development ....................................................................................... 16
  - Phase 3: Review ................................................................................................ 16
  - Phase 4: Delivery ............................................................................................... 20
- Evaluation of the Project ........................................................................................ 20

### Chapter 4: My Role and Value as an Agent of Change ................................. 22

- My Role as an Agent of Change ........................................................................... 22
- Candidate for Change: The FLS Documentation Process .................................... 23
- The Organizational Value of My Role as an Agent of Change ............................ 25

### References ........................................................................................................... 28

### Endnotes .............................................................................................................. 29

### Appendix A: Fig Leaf Technical Writer Skill Set Requirements ..................... 30

### Appendix B: Introduction from Specification Process Guide ............................ 35

### Appendix C: Fig Leaf Software Documentation Process .................................... 38

### Appendix D: Original Installation Steps for Client Installation Guide ............. 41

### Appendix E: Final Client Installation Guide ...................................................... 44
Chapter 1: Introduction

Vibrant chartreuse and plum walls, a flat screen monitor pumping out eye-catching multimedia spots, and unfamiliar acronyms, words, and phrases like CFUG, ColdFusion, and “Rock the Fig” greeted me in my life as a technical writer at Fig Leaf Software (FLS) in Washington, DC. I began full-time work for this small, yet dynamic, software company on April 9, 2001, and completed my internship for the Master of Technical and Scientific Communication (MTSC) degree at Miami University from May 21, 2001, to September 7, 2001.

When I began working for Fig Leaf in its cutting-edge office space, I did not expect to gain more than industry experience in creating software documentation. However, working in a young, rapid-development environment that was light on standards and heavy on politics taught me far more. During my employment and internship at FLS, I discovered that educating coworkers about my profession, establishing standards and processes, and negotiating organizational change were crucial to developing software documentation in this organization. More importantly, I learned that these skills, which I had considered outside of technical communication, were an integral, everyday part of practicing technical communication in the “real world.”

The main purpose of this chapter is to describe the history, organization, and culture of the company in which I worked and to explain how the corporate culture shaped the work environment at FLS. I also discuss the roles and goals of the Technical Writing team and the challenges I faced as a technical writer working in this new department and within a unique corporate environment.

Fig Leaf’s History, Organization, and Culture

Fig Leaf Software’s youth and reputation as a champion of technology permeates its history, organization, and corporate culture. Steve Drucker and Dave Watts founded Fig Leaf Software in 1992 as a custom software development firm specializing in desktop database applications. To take advantage of emerging technologies, FLS shifted its focus in 1995 to “Web-centric applications, data-driven Internet sites, and creative media” (Fig Leaf Software, n.d., n.p.). Its service offerings shifted as well and now include the following: software consulting, creative and interactive media, software training, and software product sales. At the time of my internship, FLS employed roughly 65 people in both the Washington, DC, corporate headquarters and at the Atlanta, GA, satellite office who together developed custom Web solutions for Fortune 1000 organizations, trade associations, and government agencies. Most of this development was completed using advanced technologies like Macromedia ColdFusion® and Macromedia Flash®.
The organization of FLS also reflected its youth as a company and its strong focus on technology. To stay abreast of rapid technological changes in the Internet and interactive media industries and meet the distinct needs of its clients, FLS operated like a matrix organization. It overlaid its fairly rigid organizational structure, organized departmentally by service offering (see Figure 1), with a more fluid project-team structure. On a project basis, FLS formed ad hoc teams from various departments that were more able to adapt to the unique demands of each client and project. Moreover, Fig Leaf’s Chief Technology Officer position and its executive placement in the organization revealed not only FLS’s love of technology but also its desire to put technology first.

![Figure 1. Fig Leaf Software’s organizational structure](image)

Similarly, Fig Leaf’s corporate culture demonstrated its youth as an organization and portrayed its image as a trendsetting, technology pioneer. Its office space provided some of the most distinctive examples of this corporate culture. For example, the ultra modern conference room with its chic industrial lighting, flashy marketing banners, and trendy Herman Miller® office chairs conveyed a cutting-edge, image-conscious culture. In addition to trendy and cutting edge, Fig Leaf fostered a culture that was both “fun” and “homey.” The toys strewn across employees’ desks, the personal stereos peeking from corners, and the company games room—complete with pool table, television, and punching bag—captured Fig Leaf’s culture of fun. Open work rooms with comfortable couches and complimentary soda, juice, water, and other refreshments helped Fig Leaf employees feel “right at home” in their workplace.

However, Fig Leaf’s corporate culture was more than trendy office space and fun times. Indeed, it was shaped by these characteristics but was predominantly a developer-focused, rapid-development culture where entire development projects were often completed in less than two months. And, born out of this culture was the Technical Writing team on which I worked.
About the Technical Writing Team

The Technical Writing team, a new group within the Consulting Services division, “provides its clients with high quality, expertly written technical documentation, including user guides, system administrator guides, specifications documents, online help, and more” (Fig Leaf Software, 2001, p. 18). This documentation work “arrived” for the Technical Writing team during the course of FLS’s consulting with clients. Typically, technical writers from the team served on cross-functional project teams to develop documentation for Web-based software applications, as well as to improve product design and usability. The Technical Writing team adhered to the following best practices:

Conformance with the Microsoft Manual of Style for Technical Publications and the internal Fig Leaf Software Style Guide ensures that all deliverables reflect the latest technical documentation standards. A strict review process further guarantees quality: senior writers review all documents for readability, style, organization, grammar, consistency, and usability; and cross-functional team members certify the technical accuracy of the documentation. (Fig Leaf Software, 2001, p. 18)

The team applied these practices toward its goals of understanding audience, maintaining technical accuracy, and producing high-quality documentation on time and within budget.

I worked on a two-person team under my supervisor during my internship and throughout my full-time employment with Fig Leaf Software. Because we were a small team and because of the nature of consulting, we often worked independently of each other, producing documentation for different Web applications on different project teams. While we were never asked to co-write materials, we regularly shared tips and advice; we also reviewed and edited each other’s drafts as part of the standard review process.

As Documentation Manager, my supervisor was responsible for training and directing all technical writers, including scheduling and coordinating the team’s documentation activities to maximize productivity. She also assigned incoming projects based on availability of personnel, managed the project-hour allocations for the team, and ensured that progress and quality were maintained throughout the documentation life cycle. She joined Fig Leaf in December 2000 as the first manager of the Technical Writing team.

My primary role as the newer, more junior member of this team was to develop documentation rather than manage it. During my internship, I completed a variety of documentation activities—from planning and composing to revising and delivering installation guides, user guides, online instructional text, Web site content, and training materials. I also edited materials other employees produced, which ranged from specifications to Web site content. In addition to these client projects, I assisted my supervisor with internal department-building activities like developing a standard documentation development process, creating and
maintaining templates and style guides, promoting corporate documentation branding, and producing internal quick reference guides to help employees with their writing. Appendix A offers more detail about the skill set I needed to complete these projects and activities.

The remainder of this report discusses my experiences at Fig Leaf Software during the summer of 2001. In Chapter 2, I describe the tasks, activities, and roles of my position and my major projects at Fig Leaf. In Chapter 3, I discuss, in detail, the process I followed and the many challenges I faced in developing the company’s first client installation guide. I then follow with a chapter that examines my role and value in negotiating change at Fig Leaf and presents an expanded organizational role for technical communication practitioners.
Chapter 2: Diverse Projects, Common Solutions

During my internship at Fig Leaf, I worked on sixteen projects that varied greatly in their purpose, size, duration, and medium for both internal and external clients. Both the diversity and rapidity of my projects, which I saw simply as benefits of working on a small technical writing team, were, organizationally speaking, byproducts of Fig Leaf’s culture. This rapid-development culture effectively drove all documentation activities at Fig Leaf, resulting in the documentation schedule not extending beyond two months for me and for all writers. Consequently, I never worked on a project for long, and I almost always worked concurrently on multiple projects. Figures 2 and 3 illustrate the overlapping nature and timelines of my major projects, as well as the diversity of tasks I performed throughout my internship.

Figure 2. Timelines of major projects worked on during my internship
Two Significant Projects

The following sections highlight two significant projects that are representative of the quality and diversity of work I completed in Fig Leaf’s rapid-development environment.

**Project 1: Compose Specification Process Guide**

In addition to client projects like the client installation guide described in Chapter 3, I worked on internal projects, including a specification process guide, to benefit Fig Leaf Software. At the time of my internship, Fig Leaf wrote specifications to win development work, and if this work was won, Fig Leaf became contractually bound to produce the applications described in these specifications. Indeed, these were critical documents, yet Fig Leaf’s specifications were notorious for not only being inconsistent from project to project but also for being difficult to understand and use. My divisional manager recognized these weaknesses and the need to streamline and standardize the entire specification development process so that Fig Leaf could produce more useful, more complete specifications to guide its projects. My task in this project was to develop a document describing the specification process that Fig Leaf could use during specifications development.

However, I soon learned that developing this document would require much more than composition and revision: it required me to gain a thorough understanding of the current specifications development process, develop and describe an improved process, and solicit the feedback and “buy-in” of team leaders across the company. In short, I had to facilitate the development of an improved process before I could document it.
To gain an understanding of the current process, I studied existing notes about the process and interviewed employees who had participated in the process. What came out of these informal employee conversations was a need for more clearly defined steps and roles, as well as a template for the document itself. With this information, I sketched the steps of an improved process and developed a first draft of the specifications document, which became known as the Fig Leaf Software Specification Process Guide (SPG).

After submitting this draft for editorial review and incorporating my supervisor’s changes, I had a draft ready for the divisional manager. This draft, when submitted, contained all of the elements of the process that I had discovered were needed but missing: a purpose and goals statement; a process overview; clearly defined steps; descriptions of the divisional roles at each step (for example, the roles of Consulting Services, Creative Media, and Project Management); the format and contents of each section of a specification, including the group responsible for authoring the section; and a roles and responsibilities matrix for the entire process.

The SPG project moved into the review phase when the divisional manager circulated the working draft of the specification process guide to the team leaders within the Consulting Services division. This phase was the most challenging part of the project because I had to facilitate decision making about an organizational process using the process document itself. The reviewers submitted their individual comments about the guide to me, and I compiled them into a comments document organized by chapter and page number. Each entry contained the suggested changes, a reason for the change and any commentary, and the name of the person who submitted the change. This comments document was designed for use in a meeting and enabled reviewers to decide as a group what changes to adopt. Soon after I circulated the comments document, the team leaders met to review all of the feedback and refine the document. On my behalf, my supervisor facilitated the group review, which focused on clarifying the organizational process in the document.

Following the team leaders’ meeting, I met with my supervisor to go over the implications of change for the document. Based on the team leaders’ decisions, I reorganized the SPG in the following manner to better reflect the organizational process and to make the document easier to read:

- Added sections describing the roles of the different teams (technical writing, quality assurance, and development) in the specification process
- Made each step in the process its own chapter
- Developed a roles and responsibilities matrix for each step of the process. The matrix highlighted each team’s roles, which would facilitate the guide’s use in the organization.
Before this new process could be implemented across the organization, the divisional manager had to solicit “buy-in” from the other divisions and teams, namely Project Management and Creative Services. Appendix B contains the introduction from the draft that was circulated to the other divisions for comments and additions. The other divisions received this draft coolly at first because they were surprised at having been left out of the initial review for this new organizational process; however, after reading through the document, they saw how efficient the new specification process was and agreed to add their role descriptions to the document. Therefore, after three months of iterative development, reviews, and negotiations, my divisional manager received word that all divisions had accepted the document.

**Project 2: Create Instructional Text for an Application**

An external project that I completed during my internship was to develop screen-level instructions for a content management application. This application enabled the client to create, edit, manage, and publish press releases and related Web page content, as well as make press releases available to interested reporters and the general public through its new service Web site. The client requested instructions describing the various tasks on each screen of the application, as well as technical reference and installation guides, which were completed during a later phase of the project. That phase involved coordinating the documentation with a contract technical writer the client had hired to write a user guide and online help.

The challenge of developing the screen instructions was their timing in the project: I was assigned to develop the screen instructions before the application had screen templates, graphics, or a user interface. This unusual timing for the screen instructions occurred because the graphic designer for the project was on a delayed schedule. This delay obligated me to develop the instructions without knowing how much space there would be for the instructions, where they would be placed in relation to other objects, and how they would be integrated into the screens. Because of these unknowns, my role in the project expanded from documentation specialist to documentation specialist plus usability advocate. To complete this project, I had to familiarize myself with the application, report usability issues, and convince the project team that fixing these issues would result in a more usable application.

To familiarize myself with the application, I read the specifications and drew rough screen flows. These screen flows were helpful in understanding the long, complex workflow and in envisioning the application graphically. In addition, I experimented with the application on my own. I applied what Hughes (2002, p. 281) calls *critical reverse engineering* to review the usability aspects of the application. As I experimented with the application, I asked myself the following questions:
Why would users need or want to do this task?

Would users understand how to do this task?

Does the naming and placement of the task or object in the application make sense?

It was through this experimentation that I noticed a major usability issue related to the application’s “workflow” buttons that would affect my documentation and its users. These workflow buttons (see Figure 4) were used to assign standard tasks and move press releases through the entire publication process. In this workflow, a user was assigned a task, which he or she had to successfully complete before the press release could proceed to the next user and step in the publication process.

As originally defined, the buttons managing this workflow were labeled and functioned as follows:

- **Accept Task**—Clicking the Accept Task button means that the user is comfortable with being assigned to the task. It also enables users to put a task that they have begun working on but have not completed on hold. Clicking Accept, in this case, means that the user will continue to work on the task. The task also shows up in the user’s task list as a task “In Progress.”

- **Decline Task**—Clicking the Decline Task button means that the user refuses, for whatever reason, to perform the task. When the user clicks Decline, the workflow item passes back to the owner of the last completed task for reassignment.

- **Finish Task**—Clicking the Finish Task button indicates that the user has completed the task (rather than what I thought—that a user needed more time to finish a task). When the user clicks Finish, the task is submitted to the next owner in the workflow.

- **Terminate Press Release**—Clicking the Terminate Press Release button, which only appears to users with Manager access, ends the workflow for a press release and removes it from the production queue. The application effectively abandons the press release.
The issue I noticed was that the buttons controlling the workflow were not intuitively named, ordered, or placed on the screen. In fact, on many screens, these buttons appeared adjacent to buttons controlling lower-level screen tasks; this placement was very confusing because the workflow buttons functioned independently from the tasks on a given screen.

Before reporting this issue to the application’s developer, I consulted with my supervisor about how unintuitive these buttons were and how difficult it was to understand their functionality from the labels. I explained the functionality of the buttons in the workflow, and she and I brainstormed new labels for the buttons, as well as repositioned them in relation to each other. We based the positioning of the buttons on the common industry practice for button placement: placing them either in order (from left to right) from most frequently used to least frequently used or in order according to when they are used functionally within an application. We combined these practices and repositioned the buttons with both frequency of use and functionality in mind. We also considered alternative labels for the buttons, including Create New Content for Accept, Submit for Finish, and Delete for Terminate.

With these alternative names and positions in mind, I approached the developer and asked if we could meet about a usability issue I wanted to report. He agreed, and I presented to him the usability issues I saw with the button labels, their order, and placement on the screen and my alternatives. The developer responded by insisting that the names were very intuitive and that he was not open to changing them.

After the developer expressed such resistance, I turned to a more neutral source for help, the project manager. I was still convinced that my alternatives would be more intuitive for users and would be worth any potential conflict. I explained the usability issue to the project manager, as well as the difference of opinion with the developer. I also asked him if he would sponsor a meeting so that the entire project team could provide input on the issue and make a decision together.

Before this project team meeting, I met once more with the developer and tried a second time to convince him that the workflow buttons were counterintuitive; this time he was at least willing to consider alternatives. Perhaps the most valuable insight to come out of this meeting with the developer was to make the buttons graphically, as well as textually, convey their function. Like VCR buttons or software toolbars, which express their functions using easily recognizable icons, the workflow buttons would also associate icons with function. Our modified workflow buttons are pictured in Figure 5.

![Figure 5. Modified workflow buttons in the application after second meeting with developer](image-url)
When the project team met for the meeting, the project manager explained that he was unhappy with the workflow buttons in the application and that other team members had expressed similar concerns. He called the meeting as an opportunity for the entire project team to provide input on the usability of the buttons. After the developer reviewed the functionality of the original workflow buttons with the team, I presented our proposed modifications. While the team liked the iconic button concept, they suggested further modifications to the labels and to the graphical images. As a project team, we came up with the button names shown in Figure 6.

![Figure 6. Workflow buttons in the application suggested by the project team](image)

In the end, the graphic designer refined the buttons further, removing the button “look” but retaining the icon/label combination. She also removed the Back to My Tasks button because its function was already being covered by a navigational menu (the “jump” menu) in the upper left corner of the screen. Figure 7 depicts the final workflow buttons on one screen of the application.

![Figure 7. Final workflow buttons in the application](image)

**Common Solutions**

The problems I encountered in each of these projects, while representative of the types of problems I faced during my internship, varied widely. In the first project, I was faced with defining and documenting an organizational process, whereas in the second project, I focused on product usability and intuitive button naming and placement. Although I faced diverse problems in these projects, I found common solutions in completing them. To successfully complete each documentation project, I had to first negotiate organizational change. In other words, before I
could write a specification process guide for the company, I first had to facilitate meetings where team leaders discussed the process and circulated several drafts of the process guide for company-wide review. Before I could write intuitive labels for application buttons, I had to prove to the project team that renaming the buttons in a more active, standard way would be beneficial for users. And, as Chapter 3 similarly shows, before I could develop a new document type, I had to establish standards for that document. In these ways, my diverse projects consistently required the common solution of first negotiating organizational change. I discuss this solution further in Chapter 4 when I examine my role as an agent of change.

Chapter 3 describes one of the main projects that I worked on during my internship. Through this project, I developed and delivered the company’s first client installation guide. This chapter discusses in detail the process I followed and the many challenges I faced in developing this guide. In writing about this project, I also discuss how the FLS documentation process guided me through development and helped me not only to deliver a successful installation guide but also to negotiate and establish standards for developing future installation guides at Fig Leaf Software.
Chapter 3: Standard-Setting Project: Developing a Client Installation Guide

One of the significant projects I worked on during my internship was developing an installation guide for a client’s Web application. This guide was designed as a backup reference in case the client had to reinstall the application at a later date. While not without its setbacks, this project was an exemplary one both for Fig Leaf and for me: the company delivered its first client installation guide, and I not only completed my first documentation project from start to finish but also established lasting standards for the company’s future installation guides. This chapter describes the FLS process I followed, the challenges I faced, and the solutions I developed to deliver this award-winning, standard-setting project.3

Overview and Phases of the FLS Documentation Process

The FLS documentation process, which I followed in developing the client installation guide, structures the documentation activities of the project life cycle to “ensure accurate, high quality documentation for its [Fig Leaf’s] clients” (Fig Leaf Software, 2001, Fig Leaf Software Technical Writing Department Development Process, p. 1-2). It includes four phases described below (see Appendix C).

1. **Planning** — The planning phase focuses on developing the Documentation Content Plan. This plan offers technical writers a starting place for identifying and analyzing the documentation task, audience, technical content, and project dependencies and for developing a preliminary document outline. Information-gathering activities also take place during the planning phase, including reading project materials, interviewing subject matter experts, and attending meetings.

2. **Development** — During this phase, the technical writer composes a working draft of the document. While writing is the primary activity of this phase, other activities may include querying developers and testers about application functionality, reporting usability issues, and alerting developers about any bugs encountered that may affect the documentation.

3. **Review** — Document reviews occur periodically throughout the document development process, but there are four that are scheduled and mandatory: the first editorial review, the technical review, the second editorial review, and the final document proofing. The first editorial review focuses on organization and presentation, as well as conformance to the style guides. The technical writer submits the edited draft to the project team for technical review and then to the senior writer a second time for a more thorough editorial review covering document readability, style, organization, grammar, and consistency. For the last review, the technical writing team conducts a final proofreading session to ensure that the document has the correct formatting and is free of surface errors.

4. **Delivery** — The delivery phase consists of updating the document to reflect the changes suggested during the final proofreading session, preparing the document in the requested electronic format, and printing and binding two hard copies for client delivery.
Figure 8 depicts the FLS documentation process graphically.

<table>
<thead>
<tr>
<th>Phase 1: Planning</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Develop documentation content plan</td>
</tr>
<tr>
<td>− Describe documentation task</td>
</tr>
<tr>
<td>− Describe target audience</td>
</tr>
<tr>
<td>− Describe format and delivery medium</td>
</tr>
<tr>
<td>− Describe technical content contributors</td>
</tr>
<tr>
<td>− Describe project dependencies</td>
</tr>
<tr>
<td>− Describe preliminary document outline</td>
</tr>
<tr>
<td>2. Gather information</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Phase 2: Development</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Develop working draft</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Phase 3: Review</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Complete first editorial review</td>
</tr>
<tr>
<td>2. Complete technical review</td>
</tr>
<tr>
<td>3. Complete second editorial review</td>
</tr>
<tr>
<td>4. Complete final document proofing</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Phase 4: Delivery</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Update document to reflect changes from final proofreading session</td>
</tr>
<tr>
<td>2. Generate document in requested electronic format and print and bind hard copies for client</td>
</tr>
<tr>
<td>3. Deliver to client</td>
</tr>
</tbody>
</table>

In the following sections of this chapter, I describe how I followed this process in developing the installation guide throughout June 2001.

**Phase 1: Planning**

*Gathering information:* I participated in planning activities before receiving content or beginning development work on the client installation guide. While I did not create a formal Documentation Content Plan, I did gather information about the installation guide from a
number of sources. Because project managers at FLS serve as client liaisons, I first asked my project manager about the client’s expectations for the guide. Unfortunately, the client did not have specific requirements for the guide except that it should describe the steps of the installation procedure and include the necessary configurations. However, I also knew that this client could be a demanding one who, at some point, would assert preferences about the guide, as the developers indicated had been done during the application development. Additionally, I spoke with the lead developer to gain a deeper technical understanding of the procedure I would soon be documenting. These information-gathering activities were critical in planning and developing the guide, namely because the application had already been installed in the client’s environment and was no longer available to me before I began work on the installation guide. As a result, information was harder to obtain on my own.

**Considering the audience:** Although I did not have direct access to and never met the client, I planned for and considered the guide’s audience and its most likely context for use. Because FLS installs its applications as part of delivery, I knew that the client would not be using the guide to install the application for the first time; instead, the guide would be set aside and used only to reinstall the application in an emergency or to migrate it to a different server environment in the future. It was not clear at the time whether the guide’s users would be general staff members or trained system administrators. Therefore, it also was not clear whether I would need to write instructions for novice or expert users. This lack of audience definition would become problematic in both the development and review phases of the project.

**Planning documentation content:** After completing pre-development information gathering and audience analysis, I met with my supervisor for an oral documentation content planning session. In this session, we brainstormed ideas for the content and structure of this new document type. We also discussed the allotted hours for the documentation component for which I did not provide input because I was assigned to the project in the middle of the application’s development. According to the project plan, there were 44 hours allotted to develop and deliver the guide—8 hours for the developer to document the installation procedure, 24 hours for the technical writer to write the installation procedure and develop the guide, 4 hours for the testers to verify the accuracy of the steps, and 8 hours for the technical writing manager to perform an editorial review. During this session, we did not, however, establish an official philosophy or scope for the document, an oversight that would jeopardize the content, structure, time line, and development of the guide.
Phase 2: Development

With no access to the server and no ability to work through the steps of the procedure, I had to rely on the developers who designed the application for the guide’s content and all aspects of its technical accuracy. I gathered content for the guide from the initial installation procedure that the lead developer (Developer 1) submitted to me. This procedure, shown in Appendix D, contained 23 steps, was written for a highly sophisticated audience, and was missing screen shots. From this procedure, I developed the first draft of the guide, which included the following improvements to Developer 1’s original steps:

- I chunked the developer’s long list of action-packed steps into steps containing a single action and grouped like steps into small procedures. For example, the first procedure of the installation process was “Creating Connections to the Servers,” and the steps of that procedure described how to connect the servers that support the application to each other.
- I created separate chapters to describe the application environment, the servers, and the software and hardware requirements rather than weaving this information into the steps themselves. These chapters eventually merged into one and became a standard chapter in future installation guides.
- I added “Step Number X” to the headings of each short procedure to differentiate the chronology of the procedures from the numbered steps themselves.

I submitted this draft to my supervisor for an informal first editorial review.

Phase 3: Review

Gaining access to the developers for technical content and reviews proved more difficult throughout the rest of the documentation development process. The lead developer (Developer 1) soon left the company, and a new developer (Developer 2) who was understandably less familiar with the application and installation procedure was assigned to the project. As the application neared completion, the developers became increasingly busy, and it was harder for me to schedule time with them or obtain information from them. Because I could not easily query a developer for clarification, I approached the development and review of this guide differently and modified the FLS process in the following ways to accommodate this situation:

- To guard against introducing technical inaccuracies into the guide, I highlighted content that I had questions about or steps that were incomplete using the Microsoft Word® highlighter tool. While simple, this practice proved immensely valuable during the guide’s many reviews because it indicated visually the places where content was confusing, incomplete, or potentially inaccurate.
- I queried another equally capable team, the Quality Assurance (QA) team (testers), with my technical questions. This team helped me settle compatibility issues, as well as develop a standard statement about operating systems and browsers.
- I incorporated a second technical review into the process to “double check” that the information I had gathered from such diverse sources was correct.
Completing the first technical review: After incorporating minor changes from the first editorial review, I met with Developer 2 to review the steps of the installation procedure. This review meeting consisted of correcting steps, filling in missing steps, and checking the draft for technical accuracy. I also left him the first draft—complete with its highlighted content—to review independently, insert the appropriate screen references into the steps, and create needed screen captures. This exchange constituted the first technical review.

Facing challenge #1: Meanwhile, I faced the first major challenge in the project when I requested the help of the QA team in clarifying the operating systems and Web browsers compatible with the application. This seemingly ordinary request generated much discussion and resulted in the first philosophical and legal decisions about the guide. Instead of returning a complete list of compatible operating systems and browsers, the lead tester questioned me about a phrase that appeared repeatedly in the section. The section contained the phrase, “or later,” following the oldest browser compatible with each stated platform. In other words, for the Windows 2000 platform, the guide stated that “Internet 4.0 or later browsers” were compatible with the client application. The lead tester pointed out that this phrase was not only inaccurate but also carried with it legal ramifications for the company. Fig Leaf Software does not license its software as many larger, mass-market software producers do because it develops custom applications for a single client. Therefore, Fig Leaf’s ownership and control over an application ends at delivery unless a client chooses to pay for maintenance support services.

What I did not realize was that the phrase inadvertently obligated Fig Leaf to continue supporting the application indefinitely. By stating that the application was compatible with x browser or later, Fig Leaf would have had to resolve, at no additional cost, any future problems that might have developed if the application were running in that browser. As a result, I revised the operating systems and browsers section to list only those operating systems and browsers that Fig Leaf had tested to be compatible with the application and was therefore prepared to support. In addition, I crafted the following statement that clearly defines and effectively limits Fig Leaf’s responsibility in supporting the application: “<Names of most recent browser versions> are the most recent browser versions compatible with <name of application>.” This statement, which has since become a standard note in FLS installation guides, follows the table of compatible operating systems and browsers on page 3 of Appendix E.

Facing challenge #2: Soon after resolving the browser issue, I confronted the second major challenge in the project: fundamental content change. After the application had been installed, the client, for undisclosed reasons, decided to change hosting providers for its Web site and the application. This change required Fig Leaf’s Chief Technology Officer (CTO), who installed the application the first time, to migrate the application to the new hosting provider’s servers.
Migrating the application to this new server environment required a different configuration and installation procedure and thus a different installation guide to document it. For this reason, the CTO (Developer 3) became my third source for technical information. As I would soon learn, dealing with such significant content changes late in the process not only challenged the technical accuracy of the content but also the integrity of the document as a whole.

**Completing the second technical review:** Following the migration, Developer 3 published extensive notes detailing the steps he performed in migrating the application (68 steps total). Both Developer 2 and I reviewed these migration notes and agreed that the drastic changes in the server environment and installation procedure were going to require equally massive changes in the installation guide. To develop the next draft, I merged the steps in the migration notes with the steps of my current draft and incorporated the edits from Developer 2’s first technical review. However, instead of submitting this draft for its second editorial review, as the FLS documentation process indicated, I submitted the draft to Developer 3 for a second technical review. Along with this draft, I submitted specific questions to guide Developer 3 in his review and to ensure that my remaining questions were answered.

While my decision to submit the draft for a second technical review and to indirectly set standards about the scope for this and future installation guides ultimately resulted in a more accurate, higher quality installation guide, I soon learned that doing so cost the project both time and money. The day after I submitted the guide for a second technical review, the project manager announced that installation and documentation components of the project were over budget. The client’s mid-project switch to a new hosting provider, which required additional time to migrate the application, and the slow development of the installation guide had resulted in the extra costs. With this announcement, the pressure and momentum were on to complete the documentation as soon as possible.

**Facing challenge #3:** In addition to this deadline and the knowledge that the documentation component was over budget, a third major challenge developed from Developer 3’s strong reservations about the draft. He expressed several philosophical issues with the guide; in short, he felt that there was not a guiding philosophy for the document or an understanding of scope. He pointed out to the project manager, divisional manager, lead developer, and technical writing lead that the document’s scope was not clearly defined and posed a number of questions about what details an installation guide should cover (for example, installing an FLS application, setting up supporting services, or describing the current server environment). He also recommended that the scope of the document be limited to Fig Leaf’s applications only. He later added that it would be in the company’s best interest to assume an audience of system administrators, remove all extended references to Fig Leaf’s recommended “best practices” for securing publicly
accessible Internet servers, and focus on creating a document that described how to install the application in any server environment.

Realizing that defining the audience and scope of the guide had been a challenge from the start, I agreed with Developer 3’s recommendations and solicited him as an advocate for the guide’s audience and scope. (Because the content of the guide was dependent upon Developer 3’s understanding of the installation procedure, it was beyond my power as technical writer to make an executive decision about audience and scope by myself.) On my behalf, Developer 3 met with the project manager and Fig Leaf’s senior executives about the overall philosophy for the installation guide. This important meeting resulted in the decision to deliver an installation guide directed at system administrators that covered only the steps needed to install Fig Leaf’s application and whose installation procedure was server independent. Delivering this type of guide, Fig Leaf reasoned, would increase the portability of the procedure (from one server environment to another), as well as release Fig Leaf from having to troubleshoot supporting products it did not develop.

With a clear audience and scope in place, Developer 3 finished his technical review of the guide. He answered the questions I had posed and rewrote the draft he reviewed so that it was generic enough to work in any server environment; it only contained information relevant to installing Fig Leaf’s application. When he submitted this updated draft, he also included a generic installation procedure that I could use as a guide in writing future installation guides. I then reviewed his updated draft and reinforced the new philosophy of the guide with several additional statements. For example, to clarify the intended purpose of the guide and define the level of user expertise, I added the following statement to the introduction: “This guide is intended for system administrators who may need to reinstall the [name of application] application. The guide assumes that system administrators installing this application are familiar with Microsoft Windows NT/2000, Microsoft Internet Information Server (IIS), Microsoft SQL Server, and ColdFusion Administrator.” I also inserted a statement at the beginning of the installation chapter, which transferred the responsibility of securing publicly accessible Internet servers to the guide’s users: “Before installing the application, configure both the Web and database servers according to best practices for securing publicly accessible Internet servers.” After making these changes, I submitted the draft to my supervisor for its long-awaited second editorial review. When my supervisor returned the guide with her comments, I incorporated her revisions and we then proofread the document and promoted it to the delivery phase.
Phase 4: Delivery

Before delivering the document to the client, I had to complete one more component: its presentation. Because the guide was the first one of its type that Fig Leaf had produced, the Technical Writing team did not have a template to govern its presentation and formatting. I saw this lack as an opportunity and developed a template that would also serve well for future installation guides. To save time, I modified the team’s user guide template, which contained most of the needed styles and already had the “look and feel” the Technical Writing team had established for Fig Leaf documentation. I then used this template to reformat the client installation guide. To satisfy the requests of the client, I also generated the guide in Adobe Portable Document Format®. After completing these preparations, I submitted the completed installation guide via e-mail to the project manager for delivery to the client. Appendix E contains a copy of the final installation guide.

Evaluation of the Project

A month after the project ended, the project manager scheduled a “washdown meeting” for the team to evaluate the project as a whole. Prior to this meeting, the project manager sent out a list of questions to help team members review all facets of the project. To prepare myself for the meeting, I met informally with my supervisor to discuss the project’s high points and low points. We focused on the issues and developments that affected the documentation, namely the change in hosting providers, lack of philosophy for the guide, and difficulty in obtaining technical information. We also composed answers to the project manager’s questions, which I referred to during the meeting. The project manager facilitated the washdown meeting, which lasted an hour and included three discussion prompts—what went well, what went poorly, and how the project could have been better.

The project team came away from the meeting with a deeper understanding of the project, as well as a list of best practices to apply to future projects. The meeting was also valuable for me because it gave me the opportunity to review the documentation effort and my role in the project. I was also able to express my frustrations about the lack of support I received from the project team and how that affected the documentation for the project as a whole, which the project team received with understanding and admitted that the client’s demands had perhaps diverted their attention from documentation needs.

The following chapter examines my organizational role and value as an agent of change at Fig Leaf Software. Using examples from my internship projects, including the client installation guide project, Chapter 4 highlights the many ways in which I served as an agent of change and
worked to improve fledgling business and documentation processes. The chapter concludes by defining the important organizational role of the technical communication practitioner.
Chapter 4: My Role and Value as an Agent of Change

As Chapter 2 highlights, negotiating change in a creative environment that was both young and light on standards played an important role in my job as a technical writer at Fig Leaf Software. In fact, it was this role as an agent of change that shaped not only my job and internship experience, but also my projects, my department, and the organization itself. This chapter focuses on my organizational role as an agent of change and on the value of the role to Fig Leaf Software. In the end, it presents an expanded organizational role for technical communication practitioners, which I learned through my internship to be a necessary part of practicing technical communication in the “real world.”

My Role as an Agent of Change

As a technical writer at Fig Leaf Software, I played a key role, as did the other member of the Technical Writing team, in providing clients with high quality, expertly written technical documentation. This documentation described Fig Leaf’s applications and made them more usable by explaining their functions, instructing users on how to perform tasks, and providing rationale for why users should adopt a new tool. In this way, my role as a technical writer was to support, educate, and sell Fig Leaf’s applications to clients and to develop documentation that trained users on these applications. I performed a similar role for internal documents like the Fig Leaf Software Specification Process Guide and the Fig Leaf Software Style Quick Reference Guide.

In addition to this departmental role, I served in an organizational role— as an agent of change. In all three documentation projects discussed in this report, I had to negotiate organizational change in order to accomplish my goals as a technical writer and complete my documentation work. For example, I had to facilitate decision making about an organizational process, convince a project team to consider alternate application button labels, and establish standards for a new document type before I could develop an effective process document, improve a product’s usability, and compose the company’s first installation guide. Moreover, at Fig Leaf Software, I was considered a valuable technical writer in part because, through my documentation work, I established standards and processes for a young company lacking them and created organizational tools that helped other employees do their jobs better. In short, I effected organizational change.

Yet, my ability to effect this change was tempered by the environment in which I worked. As Chapter 1 describes, Fig Leaf was a young organization that was highly creative and technologically savvy. It was also an environment that was, at times, distracted from the “bottom line” by its very creativity and skill. Because of this lack of focus, I looked for ways to improve
the existing business process while at the same time deliver quality documentation for Fig Leaf’s products. I found a solution in the MTSC problem-solving model (Anderson, 1984) that I had studied during my graduate program.

To improve the business process at Fig Leaf Software, I applied elements of the five-step MTSC problem-solving model to this “real world” context. Likewise, I found myself incorporating elements of the model in order to standardize and further develop Fig Leaf’s fledgling business process. I often incorporated these elements, like additional testing or evaluation, outside of Fig Leaf’s existing processes as a way to ease the organization into new ways of doing things. For example, I conceived of, designed, and developed labels for our documentation products delivered on CD in hopes that the organization would recognize the value in this step and, at some point, incorporate it into the process. I also experimented with Fig Leaf’s applications as a test user to uncover usability weaknesses and to demonstrate the value of usability testing during the software development process. Over time, I believe that my efforts as an agent of change, while small, would have helped Fig Leaf to evolve its incipient business process into a model (or at least closer to one). The utility of a model is in the flexible framework it provides; in Fig Leaf’s case, it would have provided a heuristic for producing and delivering software products that was applicable to almost any business context. However, my ability to facilitate this evolution was limited by many factors: Fig Leaf’s organizational structure, its creativity and technology-focused environment, and perhaps most significantly, my short tenure with the company.5

Another process upon which my role as an agent of change had a significant impact was the Fig Leaf Software documentation development process introduced in Chapter 3. Because the FLS documentation process was a fairly new one in the company, I was able to play the role of technical writer in following it to complete my documentation projects. However, by testing out the process through my projects, identifying weaknesses, and evaluating its effectiveness, I was also able to serve as an agent of change. Using illustrations from the client installation project, the following section points to weaknesses in the FLS documentation process and offers process improvements that, given more time, I would have suggested to bring the FLS documentation process closer to a model.

**Candidate for Change: The FLS Documentation Process**

Although the FLS documentation process was helpful in structuring the activities I performed to complete the client installation guide, it fell short in resolving some of the challenges that arose in this project.
The greatest challenge I faced in this documentation project was in staying within the allotted hours and budget. Because this project was the first one in which FLS delivered an installation guide, the company did not have any metrics to use in estimating the hours to complete this project. FLS also lacked a standard for developing client installation guides and, therefore, could not draw upon an existing template or document philosophy. Moreover, someone outside of the Technical Writing team estimated the hours for the documentation and did so without consulting the Technical Writing team. Because of these subversions of the standard process for estimating hours, the documentation estimate did not factor in adequate time for “first time” tasks like template development or for scope changes out of the company’s control, such as the client’s mid-project switch to a new hosting provider. For all of these reasons, Fig Leaf was unable to develop a realistic project plan for the installation guide. I found out the hard way—by serving as the technical writer for this project—that the FLS process does not include hours allocation in its planning phase. Adding hours allocation to Phase 1 would not only ensure more accurate documentation estimates (because the Technical Writing team would be completing the estimates and relying on past project metrics to do so) but also reduce the number of future projects where technical writers have to complete documentation within poorly estimated numbers of hours.

One of the other challenges I faced was in gaining access to needed information. As Chapter 3 illustrates, I had difficulty obtaining the technical content and reviews I needed in a timely manner. Not having easy access to the developers, who held the technical knowledge, or to the server and installation procedure itself, stymied the progress of the guide. Moreover, without having direct access to the client, it was difficult to plan, develop, review, and deliver a document to meet the client’s needs. Although Phase 1 of the FLS process calls for identifying team members who will contribute to the technical content and calls for information gathering, it falls short in not requiring that team members provide technical content or in guaranteeing that technical writers will have access to these team members. Phase 3 of the FLS process could be improved by requiring that these subject matter experts provide content to technical writers in a timely manner and by guaranteeing channels for technical writers to access needed content.

The third way in which the FLS process fell short was in not requiring a document philosophy and standards for this new document type from the start. I discovered when trying to develop and have drafts reviewed how detrimental it was to lack standards from the beginning of a documentation project. I did not know when developing the guide what audience to address—general staff members or system administrators—or whether to include server-specific instructions. To complete the installation guide, I first had to help Fig Leaf develop a philosophy and standards for the installation guide, which slowed the guide’s progress and required a major, last minute overhaul of its content. This in turn led to the late delivery of the guide and the
documentation component of the project going over budget. For these reasons, it is important that Phase 1 of the FLS process require that the philosophy and standards for a document be established at a project’s start.

The challenges of the client installation guide project clearly suggest places where the FLS process can be improved. Through the shortcomings of the process in developing the installation guide, I learned the value of being proactive, both as a technical writer and an agent of change, during document development. While the proactive practices I implemented may have been small, in the aggregate they can have a considerable impact on keeping a project on time and in budget. Therefore, the FLS process could be further developed by incorporating more proactive practices into its phases, such as using metrics to estimate project hours, requiring that subject matter experts provide content to technical writers in a timely manner, guaranteeing channels for technical writers to access needed content, and requiring that the philosophy and standards for a document be established at the beginning of a project.

The Organizational Value of My Role as an Agent of Change

To demonstrate the organizational value of the technical communicator as an agent of change, I use the framework Hughes (2002) outlines about the role and value of the technical communicator as a creator and disseminator of knowledge at the individual, group, and organizational levels of an organization:

Technical communicators make three important contributions as creators of knowledge within an organization:

- They help experts make their tacit knowledge explicit.
- They help design teams arrive at consensus about what the product is or does. In this regard, they are facilitating knowledge creation at the group level (or at a minimum, escalating individual knowledge to the group knowledge level).
- They create knowledge assets. That is, they transform tacit knowledge into explicit artifacts so that it can be accessed by others within the organization. (p. 280)

In fact, my efforts to effect change at the project, department, and organization levels, which are detailed in the following paragraphs, demonstrate the impact and value of my role to Fig Leaf Software.

In many ways throughout my projects, I helped experts make their implicit knowledge explicit. In composing the specification process guide, I had to interview not only the developer upon whose meeting notes the process document was based but also other employees who had participated in the specification development process in the past. Through these interviews, I probed for an understanding of the specification process, which I then transformed into a process document. Similarly, in the application instructional text project, I met with the lead developer several times so that I could capture his in-depth understanding of the application’s functionality.
And, for the client installation guide, I obtained content from three developers and converted their highly technical information into generic instructions for system administrators.

I also helped teams arrive at consensus about products, processes, procedures, and standards. In the specification project, I prepared a draft of the guide and distributed it to the team leaders for open review. I also developed a comments sheet to help facilitate their decision-making during meetings and served as an agent for the review process, soliciting “buy-in” from divisions who had not been participating in the process from the start. All of these actions helped the team leaders arrive at consensus about the organizational process of developing specifications. Similarly, in the application instructional text project, I facilitated a series of meetings with project team members, which culminated in a project team meeting about the workflow buttons. These meetings brought the issue to the team’s attention and resulted in a team decision about the button names. Last, during the client installation guide project, I submitted a draft of the guide for two editorial reviews and two technical reviews, as well as enlisted the CTO to help Fig Leaf executives set an organizational standard for the content and scope of all installation guides developed.

Beyond my role as a technical writer who develops documentation, I also developed what Hughes calls knowledge assets that could be accessed by others within the organization. For example, the specification project resulted in a policy document, the FLS Specification Process Guide, which detailed the organization’s process for developing specifications and the roles and responsibilities of those involved in the process. This document was posted to the company network and was easily accessible as an organizational reference. I also developed knowledge assets in conjunction with the installation guide. This project resulted in an installation guide template, as well as an organizational standard to guide the development of future installation guides. This template, while it would primarily be used by the Technical Writing team, was also posted to the company network.

In all three of these projects, I had to negotiate organizational change in order to do my job effectively; I could not sit by myself with my technical communication theories and writing skills and hope to be successful. Instead, I had to facilitate formal meetings and ad hoc discussions, convince (and in some cases change) the decision makers and project stakeholders, institute new processes, establish standards, and improve product usability. Through this professional interaction, I added value to the organization by creating knowledge at the front end of the business process, which Hughes calls development knowledge value (2002, p. 276), as well as on the more familiar end for a technical writer, the user end (called delivery-side knowledge value). In other words, my organizational role and true value as a technical writer for FLS was in making
knowledge more available, subjecting it to public scrutiny, and transforming it into lasting organizational “artifacts” (Hughes, 2002, p. 280).

Because of my role as an agent of change, I learned far more during my internship than simply how to create Web software documentation in a young, trendy, rapid-development environment. The projects I completed taught me that merely applying technical communication techniques learned in the classroom in an industry setting does not alone create effective documentation. Rather, I learned that skills like developing standards and negotiating organizational change, which prior to my internship I had considered as outside of technical communication, were actually part of an expanded organizational role for technical communication practitioners. In other words, practicing technical communication in the “real world” meant not only developing documentation but doing so by developing, in small ways, the organization itself.
References


Fig Leaf Software. (n.d.). *Fig Leaf Software: We’ve got you covered.* [Brochure]. Washington, DC: Author.


Endnotes

1 CFUG stands for ColdFusion User Group, a forum for Web application developers to learn more about Macromedia ColdFusion®, share tips, and network. ColdFusion is a Macromedia solution for developing Web applications. FLS employees used the phrase, "Rock the Fig," to encourage coworkers to work hard for the sake of the company.

2 Fig Leaf Software asked each department or team to develop a matrix defining the skill sets of its positions. These matrices covered job responsibilities, qualifications, education, work experience, and supervisory requirements. The Technical Writer matrix described a departmental career path through four levels—Technical Writer, Senior Technical Writer, Lead Technical Writer, and Documentation Manager. During my internship and employment at FLS, we had employees performing the duties of the Senior Technical Writer (me) and Documentation Manager (my supervisor) positions.

3 The Web application that the installation guide supported won a prestigious award in the information architecture category of a well-respected competition for interactive media (2002).

4 It is important to note that the Vice President of Consulting Services—not the Documentation Manager—estimated the hours for the installation guide. He did not consider that the Technical Writing team would write the guide when he estimated the hours; instead, he assumed that the developers would be preparing the guide. It was only after the client had signed off on the contract (when project plan hours are not negotiable) that he shared the final project plan with the Documentation Manager and assigned the work to the Technical Writing team.

5 Because of Fig Leaf’s ongoing financial difficulties, I was laid off from my position in January 2002, just nine months after I had starting working for the company.

6 The standard process for developing project plans at FLS began with the project manager setting up the schedule. Next, the project plan circulated from department to department, each department adding its tasks and the appropriate hours. After the project plan passed from development to quality assurance to technical writing to creative media, it was submitted to the FLS Vice Presidents for official review.
Appendix A: Fig Leaf Technical Writer Skill Set Requirements
## Technical Writer

### Skill Set Requirements

<table>
<thead>
<tr>
<th>Job Responsibilities</th>
<th>Technical Writer</th>
<th>Senior Technical Writer</th>
<th>Lead Technical Writer</th>
<th>Documentation Manager</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uses Microsoft Word, Adobe FrameMaker, or similar word-processing tools to develop documentation.</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Takes screen captures, creates graphics, and produces printed copies of the documentation.</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Implements commonly used documentation concepts, practices, and procedures during documentation development and editing cycles.</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Accepts and learns from editorial feedback.</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Edits documents for format, grammar, accuracy, and organization.</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Demonstrates writing proficiency, specifically in the areas of document organization, clarity, quality, task orientation, completeness, style, and accuracy.</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Considers audience during documentation development and its impact on document organization, applied techniques, and tone.</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Works effectively with developers, quality assurance analysts, designers, project managers, and other personnel as part of a cross-functional team.</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Meets documentation deadlines.</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Stays up-to-date with tools, technologies, and trends in the technical writing profession.</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Gathers information from a variety of sources, including interviewing technical experts and experimenting with software applications.</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Mentors and trains other writers about internal documentation processes and Fig Leaf Software products.</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Job Responsibilities</td>
<td>Technical Writer</td>
<td>Senior Technical Writer</td>
<td>Lead Technical Writer</td>
<td>Documentation Manager</td>
</tr>
<tr>
<td>----------------------</td>
<td>------------------</td>
<td>-------------------------</td>
<td>-----------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>Actively participates in meetings to improve product design and consistency to increase application usability.</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Considers interface design as it relates to usability and the clarity of instructions.</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Works independently on a daily basis.</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Reviews technical specifications to accurately estimate the time required to complete all documentation tasks for a project.</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Effectively evaluates his or her own work on a project, as well as the overall work of a project team.</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Creates and maintains style guides and templates.</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Analyzes customer and audience requirements to determine the type, scope, and complexity of required documentation.</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reviews and evaluates technical information, projects, standards, and processes.</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>At a division level, participates in meetings to improve standards and processes while considering the impact on projects, the Documentation department, and the division as a whole.</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Prepares employee performance evaluations.</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Coordinates documentation review schedule with Development Team Leaders, Project Managers, and Creative Media.</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oversees all work done by documentation department.</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reviews technical specifications to accurately estimate and evaluate time estimates for documentation tasks across all projects.</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Job Responsibilities</td>
<td>Technical Writer</td>
<td>Senior Technical Writer</td>
<td>Lead Technical Writer</td>
<td>Documentation Manager</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------------</td>
<td>------------------</td>
<td>-------------------------</td>
<td>-----------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>Estimates and tracks multiple projects while meeting documentation schedules.</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td><strong>Qualifications</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Familiar with Microsoft Word or similar word-processing tools used to develop documentation.</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Ability to accept and learn from editorial feedback.</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Ability to edit documents for format, grammar, accuracy, and organization.</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Ability to meet deadlines.</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Ability to work successfully in a team environment.</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Experience writing and editing a variety of technical materials, including manuals, instructions, job aids, training guides, articles, reports, brochures, press releases, and/or online help.</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Ability to work on and prioritize multiple projects in parallel.</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Previous staff supervision experience.</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td><strong>PLUS:</strong> Experience documenting software applications, including Web applications and applications that require rapid documentation turnaround.</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td><strong>PLUS:</strong> Familiar with specialized and industry-standard technical writing software, including FrameMaker, RoboHelp, Acrobat, PaintShopPro, PageMaker, or other similar technology.</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td><strong>PLUS:</strong> Project/Resource Management experience in the software development field.</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>
**Skill Set Requirements**

**Technical Writer**

<table>
<thead>
<tr>
<th>Job Responsibilities</th>
<th>Technical Writer</th>
<th>Senior Technical Writer</th>
<th>Lead Technical Writer</th>
<th>Documentation Manager</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PLUS:</strong> Has technical writing certificates, training, or an advanced degree in a related field.</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td><strong>Education and Work Experience</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Requires a bachelor's degree in English, communications, or a related field, or has equivalent work experience.</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Requires 0-2 years of experience.</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Requires 2-4 years experience.</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Requires 4+ years of experience.</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td><strong>Supervisory Requirements</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Works under immediate supervision and typically reports to a Lead Technical Writer or Documentation Manager within the documentation department.</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>May be called upon to mentor another writer. Reports to a Lead Technical Writer or Documentation Manager within the documentation department.</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>May manage and mentor a small group of writers. Reports to the Documentation Manager.</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Leads and directs all members of the documentation department. Reports to the Vice President of Consulting or other member of a larger corporate department.</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>
Appendix B: Introduction from Specification Process Guide

Note: Out of respect for the sensitive nature of this internal process, this appendix contains only an excerpt from the 42-page document.
1. Introduction

1.1. Overview

Specifications are important documents at Fig Leaf Software (FLS) because they help us to define, shape, and manage our client projects. Generally speaking, specifications are documents that describe the building blocks of an application. At the beginning of a project, FLS specifications describe the intended features and functionality of an application, and they also capture the business needs of our clients. Specifications are living documents that we update regularly to reflect changes in the applications we develop. FLS specifications should be technical enough to be useful for the development team, yet simple enough for clients to understand.

1.2. About This Document

This document enables FLS employees to better understand the purpose of a specification, goals for writing a specification, the FLS development process, and their roles in developing successful specifications. It is intended for all FLS employees who are involved in the specifications development process, including project managers, developers, designers, quality assurance analysts, and technical writers.

The information in this guide is organized into the following chapters.

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The <strong>Introduction</strong> provides an overview about the specification development process at FLS and describes the organization of this document.</td>
</tr>
<tr>
<td>2</td>
<td><strong>Purpose, Goals, and the Development Process</strong> describes the purpose and goals behind FLS specifications and briefly describes each step of the development process.</td>
</tr>
<tr>
<td>3–14</td>
<td><strong>Chapters 3 through 14</strong> explain each step of the development process in depth, as well as the roles of specification team members at each step of the process.</td>
</tr>
<tr>
<td>15</td>
<td><strong>Specifications Document Format</strong> presents the structure for FLS specifications and describes the contents of each section of the document.</td>
</tr>
<tr>
<td>16</td>
<td><strong>Roles and Responsibilities</strong> (Appendix A) contains a chart depicting the role of each department at every step of the specification development process.</td>
</tr>
</tbody>
</table>
### Chapter Description

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>Specification Section Authors (Appendix B) lists the author or authors for each section of the specification in chart form.</td>
</tr>
</tbody>
</table>

#### 1.3. Where to Find More Information

[FLS Senior Staff: Please select specifications for the list below that are representative of the content described in this guide.]

To view the content of a typical specification, see the following sample specifications:

- Sample specification title goes here
- Sample specification title goes here

To view the structure of a typical specification, see the specification template located on the Fig Leaf Software network at [Add network location here].

**Note** The specification template is a document shell that specification development teams can start with and fill in as they develop the sections of a document. The template contains the styles and formatting of a standard specification as well as some boilerplate text that can be reused in every specification.
Appendix C: Fig Leaf Software Documentation Process
1. Software Documentation

This section describes the role of the Technical Documentation team from the onset of a project to its end.

- **Overview** presents a high-level description of the team and the process it follows during documentation development.
- **Documentation Development Phases** describes in detail each aspect of the FLS documentation process.
- **Recommended Project Documentation** provides detailed information about the types of documentation that are recommended for this project.

1.1. Overview

The Technical Documentation team at Fig Leaf Software provides its clients with high quality, expertly written technical documentation, including user guides, system administrator guides, specifications documents, online help, and more. Conformance with the Microsoft Manual of Style for Technical Publications and the internal Fig Leaf Software Style Guide ensures that all deliverables reflect the latest technical documentation standards. A strict review process further guarantees quality: senior writers review all documents for readability, style, organization, grammar, consistency, and usability; and cross-functional team members certify the technical accuracy of the documentation.

1.2. Documentation Development Phases

Fig Leaf Software follows a thorough documentation development process to ensure accurate, high quality documentation for its clients. There are four phases of the documentation development process: planning, development, review, and delivery.

1.2.1. Planning

The Planning phase focuses on the development of the Documentation Content Plan (DCP), which describes the documentation task, the target audience for the document, the format and delivery medium, the team members who will contribute to the technical content, project dependencies, and the preliminary document outline. The DCP enables the writer to assess whether enough information is available to begin work, and it allows cross-functional team members to provide input about the type of information that is presented in a document.

The Planning phase also includes preliminary information-gathering activities, such as reading existing project documentation, interviewing developers and testers, and attending project meetings.
1.2.2. Development
During the Development phase, the writer compiles all information gathered during the Planning phase into a working draft of the document. While writing is intensive, this step also includes additional activities, such as querying developers and testers about application functionality, reporting usability issues, and alerting developers of any bugs encountered that may affect the documentation. Typically during this phase, the writer also incorporates screen captures and other visual aids into the document.

1.2.3. Review
Throughout documentation development, writers periodically submit their documents for review. Although writers may solicit feedback at any point in the Development phase, four reviews are scheduled and mandatory: first editorial review, technical review, second editorial review, and final document proofing.

After the primary writer drafts the bulk of the document, a senior writer reviews the document for organization and presentation, as well as conformance to the Microsoft Manual of Style for Technical Publications and the internal Fig Leaf Software Style Guide.

When the editorial feedback has been incorporated, the document is submitted to the entire project team (identified in the DCP) for a technical review. The writer compiles all of the technical feedback and ensures that the document is updated with any new information.

Next, the document is submitted a second time to the senior writer, who performs a thorough editorial review of the document. This review focuses on document readability, style, organization, grammar, and consistency.

Finally, the Technical Documentation team conducts a final proofreading session to review the document before it is delivered to the client. Priorities during this session include ensuring the correct formatting rules are applied, checking figure numbers, and verifying cross-references.

1.2.4. Delivery
The delivery phase consists of updating the document to reflect changes suggested during the final proofreading session. When all edits have been incorporated, the writer generates the document in the requested electronic format, and prints and binds two hard copies of the document for client delivery.
Appendix D: Original Installation Steps for Client Installation Guide

Note: Sensitive or proprietary information has been removed from this sample to protect the confidentiality of the client.
The following table illustrates the computers that envelop the <client> environment at <hosting provider>. The production and development servers have ColdFusion 4.5.2 installed. The database server has SQL Server 2000 installed.

<table>
<thead>
<tr>
<th>Environment</th>
<th>IP Address</th>
<th>Computer Name</th>
<th>NT User Name</th>
<th>NT Password</th>
<th>CF/SQL Server Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dev/Staging</td>
<td>&lt;IP address&gt;</td>
<td>&lt;name 1&gt;</td>
<td>&lt;user name&gt;</td>
<td>&lt;password&gt;</td>
<td>CF Password:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>&lt;password&gt;</td>
</tr>
<tr>
<td>Production</td>
<td>&lt;IP address&gt;</td>
<td>&lt;name 2&gt;</td>
<td>&lt;user name&gt;</td>
<td>&lt;password&gt;</td>
<td>CF Password:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>&lt;password&gt;</td>
</tr>
<tr>
<td>Database</td>
<td>&lt;IP address&gt;</td>
<td>&lt;name 3&gt;</td>
<td>&lt;user name&gt;</td>
<td>&lt;password&gt;</td>
<td>SQL Server –</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>username:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>&lt;user name&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Password:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>&lt;password&gt;</td>
</tr>
</tbody>
</table>

1. Create a connection, in Terminal Services, to each of the computers in the above table.
2. Map a drive to the development computer (<path 1>) from your local computer.
3. Map a drive to the development computer (<path 2>) from your local computer.
4. Map a drive to the database server (<path 3>) from your local computer.
6. Copy the application to the share directory on the development server and to the share directory on the production server.
7. Copy the database to the share on the database server.
8. Open connection to <name 3> (database server).
9. Move the database backup file to d:\Microsoft SQL Server\mssql\backup\.
10. Open up SQL Server Enterprise Manager on the database server. Select the <name 3> group. Right click on the databases folder. Select new database. Name the new database <name 3 user name>. Leave all selections at their default values.
11. Once the database has been created, right click on the database name. Select All Tasks Æ Restore Database. Check the Restore From Device radio button. Click on the Select Devices button and select the backup file that was unzipped in step 11. Under options check force restore over existing database. Point the physical file names for the data to d:\Microsoft SQL Server\mssql\Data and transaction log to d:\Microsoft SQL Server\mssql\log\.
12. Create user <SQL Server user name> with password of <SQL Server password> with permissions on the <name 3> database.
13. Close the Terminal Services connection to the database server.
14. Open Terminal Services connection to the development web server.
15. Move the application to <directory path>.
16. Close the Terminal Services connection to the database server.
17. Open Terminal Services connection to the production web server.
18. Move the application to <directory path>.
19. Create a new web site in IIS for the production web server. Select <directory path> as the default directory. Enter <application web address> in the host header name field.
20. Bring up the ColdFusion Administrator for both the development and production web servers. The following information will be duplicated on both servers. The passwords are in the above table. Create an ODBC datasource (<datasource 1>) with the SQL server user name and password from the above table. Enable client variables for the <datasource 1> datasource. Leave the create tables option unchecked. Create an entry under the mail heading with the IP address of <IP address>.
21. Open application.cfm in the production server. Change basehref to <application web address> and the datasource variable to <datasource 1>.
22. Open application.cfm in the development server. Change basehref to <application IP address>, the datasource variable to <datasource 1> and basefilepath to <directory path>.
23. Browse to <application web address> to view the application on the production server. Browse to <application IP address> to view the application on the development server.
Appendix E: Final Client Installation Guide

Note: Sensitive or proprietary information has been removed from this sample to protect the confidentiality of the client.
Contents

CHAPTER 1
Introduction .......................................................................................................................... 1
OVERVIEW ............................................................................................................................ 1
ABOUT THIS GUIDE ........................................................................................................... 1
OPERATING SYSTEMS AND BROWSERS ......................................................................... 2
CONVENTIONS .................................................................................................................... 3
NOTES AND REFERENCE ICONS .................................................................................... 3

CHAPTER 2
Hardware and Software Requirements ............................................................................... 4
HARDWARE REQUIREMENTS .............................................................................................. 4
SOFTWARE REQUIREMENTS ............................................................................................ 4
ADDITIONAL SOFTWARE RECOMMENDATIONS .............................................................. 4

CHAPTER 3
Installation ......................................................................................................................... 5
STEP 1: INSTALLING THE DATABASE ................................................................................ 5
STEP 2: TRANSFERRING THE APPLICATION TO THE WEB SERVER .................................. 6
STEP 3: CONFIGURING MICROSOFT IIS ON THE WEB SERVER ........................................ 6
STEP 4: CONFIGURING THE COLDFUSION SERVER ON THE WEB SERVER ..................... 7
STEP 5: VIEWING THE INSTALLED APPLICATION ............................................................. 8
Overview

The <client> <application> application provides a portal to current, accurate, worldwide environmental data. It is intended for use by <client> constituents, researchers, policy makers, and students. <Application> enables users to search and retrieve environmental information from a variety of trusted sources using its five information tools and ten subject categories. Users of the site can find specific data in the searchable database; browse detailed data tables, country profiles, maps, and source notes; and read in-depth feature articles.

Figure 1 shows the <application> home page.

About This Guide

This guide is intended for <client> system administrators who may need to install the <application> application.

Note The guide assumes that system administrators installing this application are familiar with Microsoft Windows NT/2000, Microsoft Internet Information Server (IIS), Microsoft SQL Server, and ColdFusion Administrator.
The guide contains both overview information and instructions about installing the application. Please read Chapter 2, “Software and Hardware Requirements” before installing the application.

The information in this guide is organized into the following chapters.

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction describes the application and lists compatible operating systems and browsers.</td>
</tr>
<tr>
<td>2</td>
<td>Hardware and Software Requirements provides information about the hardware and software needed to support the &lt;application&gt; application.</td>
</tr>
<tr>
<td>3</td>
<td>Installation provides steps for installing the application.</td>
</tr>
</tbody>
</table>

**Note** To install the <application> application properly, you must follow the steps for all sections in the order that they appear in Chapter 3, “Installation.”

**Operating Systems and Browsers**

The following table lists the supported operating systems and browsers.

<table>
<thead>
<tr>
<th>PLATFORM</th>
<th>BROWSER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Macintosh OS</td>
<td>Internet Explorer 4.0</td>
</tr>
<tr>
<td></td>
<td>Internet Explorer 4.1</td>
</tr>
<tr>
<td></td>
<td>Internet Explorer 5.0</td>
</tr>
<tr>
<td></td>
<td>Internet Explorer 5.5</td>
</tr>
<tr>
<td></td>
<td>Netscape 4.x</td>
</tr>
<tr>
<td>Windows 95</td>
<td>Internet Explorer 4.0</td>
</tr>
<tr>
<td></td>
<td>Internet Explorer 4.1</td>
</tr>
<tr>
<td></td>
<td>Internet Explorer 5.0</td>
</tr>
<tr>
<td></td>
<td>Internet Explorer 5.5</td>
</tr>
<tr>
<td></td>
<td>Netscape 4.x</td>
</tr>
<tr>
<td>Windows 98</td>
<td>Internet Explorer 4.0</td>
</tr>
<tr>
<td></td>
<td>Internet Explorer 4.1</td>
</tr>
<tr>
<td></td>
<td>Internet Explorer 5.0</td>
</tr>
<tr>
<td></td>
<td>Internet Explorer 5.5</td>
</tr>
<tr>
<td></td>
<td>Netscape 4.x</td>
</tr>
</tbody>
</table>
## PLATFORM | BROWSER
---|---
Windows NT | Internet Explorer 4.0
| Internet Explorer 4.1
| Internet Explorer 5.0
| Internet Explorer 5.5
| Netscape 4.x
Windows 2000 | Internet Explorer 4.0
| Internet Explorer 4.1
| Internet Explorer 5.0
| Internet Explorer 5.5
| Netscape 4.x

**Note** Internet Explorer 5.5 and Netscape 4.7 are the most recent browser versions compatible with <application>.

### Conventions

Fig Leaf Software™ uses common conventions to describe tasks.

1. **Bold** text identifies an action to perform. For example:
   
   Click **OK** to continue.

2. **Italicized** text describes a variable value to substitute. For example:
   
   On the Admin page, click **Users**, and then click **user name** to edit the user’s profile.

3. Monospaced text identifies system commands to enter or a system response that appears. For example:
   
   ```
   PutLn( NameTag(#, “dname “, name) );
   ```

### Notes and Reference Icons

Whenever notes or icons appear in the left margin of a page, it alerts you to special information.

**Note** A note provides information that emphasizes or supplements important points of the main text.

- The **Warning** icon cautions you that a fatal error, unsatisfactory output, or loss of data may occur if the directions are not followed carefully.

- The **Tip** icon offers suggestions to simplify a task or describes a useful shortcut. Tips may also describe an alternate way to use the techniques described in the text.

Other icons appear to identify a process or button described in the text.
CHAPTER 2
Hardware and Software Requirements

This chapter provides detailed information about the software and hardware needed to effectively run the <application> application.

Hardware Requirements

The <application> application required the following hardware:

- One server to host Web and application services
- One server to host database services

Software Requirements

The following software applications are needed to run the <application> application.

<table>
<thead>
<tr>
<th>SERVICES</th>
<th>TYPE</th>
<th>SOFTWARE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web services</td>
<td>Operating system</td>
<td>Windows NT 4 SP 6a or Windows 2000</td>
</tr>
<tr>
<td></td>
<td>Web server</td>
<td>Microsoft IIS 4 or IIS 5</td>
</tr>
<tr>
<td></td>
<td>Application server</td>
<td>ColdFusion Enterprise Edition 4.5.1 SP2</td>
</tr>
<tr>
<td>Database services</td>
<td>Operating system</td>
<td>Windows NT 4 SP 6a or Windows 2000</td>
</tr>
<tr>
<td></td>
<td>Database server</td>
<td>Microsoft SQL Server 2000 Standard Edition</td>
</tr>
</tbody>
</table>

*Note* Although <application> may run on Microsoft SQL Server 7 SP 3, the application was developed to achieve optimal performance running Microsoft SQL Server 2000.

Additional Software Recommendations

For remote access and management, a remote control application such as Symantec pcAnywhere or Microsoft Terminal Services should be installed and configured on each machine before beginning installation.
To install the <application> application properly, you must follow the steps for all sections in the order that they appear in this chapter. The chapter assumes that you have access to the server console or remote access using a remote control application installed on each server.

Before installing the application, configure both the Web and database servers according to best practices for securing publicly accessible Internet servers. All available security patches should be installed.

In addition, the application requires an SMTP server for sending mail. You can install the SMTP server on the Web server, or you can use a remote SMTP server, if one exists.

### Step 1: Installing the Database

1. Copy the database file to the database server.

   **Note** If you are using the original installation media, you are installing from a database backup.

2. From the server console, open the SQL Server Enterprise Manager.

3. Select the registered entry corresponding to the local database service.

4. Right-click the database folder, select **New Database**, and create a new database. Record the name given to the new database (it will be required later).

5. On the Database Options window, click **Truncate Log on Checkpoint**.

   **Note** Use the default values for all other options.

6. After creating the database, right-click the database entry, select **All Tasks**, and click **Restore Database**.

7. On the Restore Database window, complete the following tasks:
   - Click **Restore From Device**.
   - Click **Select Devices**.
   - Click **Restore Devices**, and then click **Add**.
   - Click **Restore Destination**, browse for the database backup file, select it, and click **OK**.
   - Click **Restore Devices**, and then click **OK**.
Click the **Options** tab, and then click **Force Restore Over Existing Database**.

Point the physical file names for the database and transaction log to the following location:

```
datapath\filename
```

**Note**

The datapath corresponds to the data storage location for SQL databases on your database server.

On the Restore Database window, click **OK**.

8. Create a user with the following property settings:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>PROPERTY SETTING</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Name</td>
<td>Type <code>&lt;user name&gt;</code> for the user name.</td>
</tr>
<tr>
<td>Password</td>
<td>Type and record a password of your choice.</td>
</tr>
<tr>
<td>Access Rights</td>
<td>Assign <code>&lt;user name dbo&gt;</code> rights to your database.</td>
</tr>
</tbody>
</table>

---

**Step 2: Transferring the Application to the Web Server**

To transfer the application to the Web server, follow these steps:

1. Create a Web root directory for the application.

   **Note**

   This directory should not be contained within any other Web-accessible directories.

2. Copy the application files to the directory.

---

**Step 3: Configuring Microsoft IIS on the Web Server**

To set up the Web site on the Web server, follow these steps:

1. Open the **Internet Services Manager**.

2. Right-click the **Computer** icon.

3. Click **Action**, select **New**, and then click **Web Site** to begin the Web Site Creation Wizard.

4. Click **Next** to continue the Web site creation process.

5. On the Web Site Description window, type a description for the Web site, and then click **Next**.

6. On the IP Address and Port Settings window, type `<application Web address>` in the Host Header field, and click **Next**.

7. On the Web Site Home Directory window, browse to the directory where the application files are located, set it as the root directory for the Web site, and click **Next**.
8. On the Web Site Access Permissions window, click Next.

9. In the Internet Services Manager, select the Web site you have created, click Action, and then click Start.

Step 4: Configuring the ColdFusion Server on the Web Server

To configure the ColdFusion server on the Web server, follow these steps:

1. Open the ColdFusion Administrator Web application on the Web server.

2. On the Settings page, click Timeout requests after \( n \) seconds, and set the number of seconds to 30.

3. On the Caching page, set the Template Cache to 8192 KB.

4. In the ODBC page, create an ODBC data source with the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>PROPERTY SETTINGS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Name</td>
<td>Type <code>&lt;data source&gt;</code> for the data source name.</td>
</tr>
<tr>
<td>Server</td>
<td>Enter the IP address of the database server.</td>
</tr>
<tr>
<td>Database</td>
<td>Enter the name of the database created earlier.</td>
</tr>
<tr>
<td>User name</td>
<td>Enter <code>&lt;user name&gt;</code> for the user name.</td>
</tr>
<tr>
<td>Password</td>
<td>Enter the password assigned to that user.</td>
</tr>
<tr>
<td>Maintain Database Connection</td>
<td>Select Main Database Connection.</td>
</tr>
<tr>
<td>Restrict SQL Operations</td>
<td>Select all available options.</td>
</tr>
</tbody>
</table>

5. On the Variables page, select the `<data source>` data source, and click Add.

6. Select all Client Variable Storage options, and then click Create.

7. On the Mail page, type the hostname or IP address of the SMTP server for the application.

8. On the Debugging page, enter `<application IP address>` as the IP address to prevent accidental display of debugging information to users.
9. Clear the **Enable Performance Monitoring** and **Display the template path in error messages** checkboxes.

10. Open the application.cfm file in the root application directory, and then update the following variables:

<table>
<thead>
<tr>
<th>VARIABLE</th>
<th>VARIABLE VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basehref</td>
<td>Change the basehref variable to <code>&lt;application Web address&gt;</code>.</td>
</tr>
<tr>
<td>Data source</td>
<td>Change the data source variable to <code>&lt;data source&gt;</code>.</td>
</tr>
<tr>
<td>Basefilepath</td>
<td>Change the basefilepath variable to the path for the root application directory.</td>
</tr>
</tbody>
</table>

**Step 5: Viewing the Installed Application**

To view the installed application on the Web server, type `<application Web address>` into the address bar of your browser.