Installation Documentation Documentation

Linda Branagan
David Tilbrook

Information Technology Center
Carnegie Mellon University
lb1y+@andrew.cmu.edu
dt+@andrew.cmu.edu

ABSTRACT

There is no single definition of UNIX.® In spite of the variations, vendors are expected to supply their products to a wide variety of UNIX environments. In addition to creating code portability problems, the wide range of target systems complicates the installation process. Unlike proprietary single machine type operating systems, for which installations can be fully automated, UNIX installations are characterized by a great need for human participation. Typically, installers must edit configuration files and makefiles, create target directories and diagnose the problems that inevitably arise. Worst of all, there are almost as many installation procedures as there are UNIX products.

A client's first impression of a software product is the ease with which it installs on his/her system. Because the process requires a large amount of human intervention, UNIX software should come with documentation that makes clear exactly what is expected of the installer. This documentation needs to be accurate, complete and, above all, concise. Lengthy, hard-to-follow installation documentation will not be consulted until something goes wrong – and will prove inadequate even then.

It is not hard to produce effective installation documentation, but there are many pitfalls for the unwary. (This paper contains several horror stories which demonstrate these pitfalls). Avoiding them is especially difficult because installation documentation is typically created by programmers (not writers) who are usually working under tremendous time constraints.

We have applied techniques from the field of document design to produce a set of *installation documentation guidelines*. Our guidelines, which are presented here, include a generic outline of a good installation document and a discussion of strategies for the unintrusive (i.e., painless) integration of the development of installation documentation into the development of the software itself.

End of Abstract

1. Introduction

The task of installing software has well-founded reputation for being long and arduous. An installation is rarely completed successfully on the first attempt and almost always takes longer than it should. Every minute spent fighting with an installation increases the client's cost for the software, and decreases their opinion of the supplier's products (see Horror Stories #2 and #8). The cause of this reputation rests largely on the normal quality of delivered installation documentation, which is often unhelpful and, at worst, harmful (see Horror Stories #7 and #12).

The Information Technology Center at Carnegie Mellon University will be delivering a large number of software products over the next few years. A recognition of the problems inherent in preparing good installation documentation led us to do the work that is in part described in this paper. The intention was to establish some overall guidelines for the style, scope and formatting of installation documentation, to establish a checklist of necessary ingredients and to suggest, procedures to create such documents. This was necessary to improve the chances of producing effective installation documentation. We also felt that a clear set of guidelines would enable installers to easily apply knowledge gained about one product to the installation of another.

The bulk of this paper contains the guidelines themselves and some general notes on their preparation. An appendix contains a number of the more amusing ‘‘horror stories’’ that were collected as part of the background research. The following sections are a justification of this work, an introduction to the guidelines and an explanation of the research process that brought about their formulation.

Why This Work is Important:

There are several uncontrollable factors that make producing effective installation documentation a difficult task. To begin with, the first rule of writing documentation is ‘‘always define the audience.’’ Unfortunately, it is impossible to determine the skill level of an installer. They may be anything from over-zealous undergraduates to kernel wizards.

Additionally, the supplier has very little idea about what is already in existence at an installer’s site, and configurations will undoubtedly vary among sites. Defining the target environment is very difficult because the total number of possible configurations that will still meet the supplier’s ‘‘requirements’’ is astronomical.

Therefore, to be effective, installation documentation will have to compensate for these shortcomings as fully as possible. It will have to serve the needs of less-experienced installers without boring the experts to tears, perhaps causing them to skip or miss important information due to lack of attention or over-confidence. It will have to state and explain all target site requirements. It will also have to provide information on trouble shooting, diagnosis and repair tasks that can actually be performed successfully and repetitively without additional information from the supplier. And, it will have to do all this in a clear, effective manner.

Few suppliers exhibit much desire to provide extended support. Developers have better things to do than spoon-feed instructions to confused installers and 800 numbers are expensive to maintain. Unfortunately, installation documentation is usually given a low priority. Developers are often preoccupied with completion of the software itself, so no one thinks about installation until the last minute – by which time everything is behind schedule and everyone is in a hurry.

The creation of complete, accurate installation documentation begins with a method for collecting, clarifying and formalizing input from developers. This method must be non-intrusive and non-time-consuming. Specific instructions must be given concerning the content and scope of the information the developers must provide. These instructions should evolve into a straightforward, well-understood standard with which developers feel comfortable.

Process

We began with a literature search, which yielded many basic rules about designing documents in general, but nothing that could solve the nasty problems that are specific to installation documentation.

The next step involved pleas for input. We posted to a fairly technically-oriented newsgroup whose readers are notorious for active discussion asking for input about what they think installation documentation should (and should not) provide and how that information should be formated. The most common answer was ‘‘find a technical writer to help you.’’ Unfortunately, the primary author IS a technical writer, and, as the literature search and the query itself pointed out, she does not have any of the answers. Incidentally, the second most common response was ‘‘Can I have a copy of this when you’re finished?’’

We conducted interviews with people who have used and written installation documentation. This proved to be a much more useful technique. Installers were more than willing to complain about particular instances of ineffective documentation, and were able to make educated, constructive suggestions for improvement. Writers of installation documentation had input that proved to be even more valuable. These people could explain the reasons behind their original decisions about the content, scope and format of their documentation (assuming, of course, that they were given enough time to make any decisions at all). More importantly, they could explain where the documentation failed, why it did so, and what they planned to do to avoid the problem in the future.

The balance of this document presents an outline of a good installation document, general notes, and an embryonic strategy.

2. Installation Document Outline

Very shortly, most readers will be wondering why we have taken such great pains to state the obvious. Admittedly, it *seems* like nothing more than common sense to include many of these items in an installation document. However, rest assured that no item is mentioned unless we found at least one (in most cases, more than one) violation of so-called common sense.

The outline explains sections that should be included in installation documentation. Obviously, the term ‘‘installation’’ covers a wide variety of processes, ranging from astoundingly simply to extremely complex. This outline is not definitive. Not all the items listed here will apply to all installations. In fact, the list is geared toward operating systems or large application packages, rather than single program installations.

2.1. An Informative Title

Most documents have names, so if you’re using one, you might as well pick one that someone else will find useful. Titles like ‘‘Installing Version 3.78’’ or ‘‘Release Notes’’ will not be particularly helpful if the document gets separated from the rest of the distribution. Squeeze in all the important information.

2.2. A Useful Table of Contents

Section titles are almost as important as the document title – therefore, they also should be somewhat informative in nature. In other words ‘‘Section 2 – Saving and Restoring Local Modifications’’ is a better section title than just ‘‘Chapter 2’’.

2.3. A Packing List

Trying to do an installation without all the materials is almost as bad as trying to do a jigsaw puzzle without all the pieces (see Horror Story #1). Provide a list of the contents of the distribution, including the titles and number of tapes and manuals that should be included.

2.4. Document Conventions

Avoid breeding confusion whenever possible. Explain what boldface and italicized words mean. Explain how commands to be typed by the installer are differentiated from text displayed by the system.

2.5. Hardware Requirements, Software Requirements, Resources

Hardware specifications should be as detailed as possible. Four important pieces of information should be present here.

1. Specific descriptions of supported hardware
2. Specific descriptions of supported hardware configurations
3. Descriptions of peripherals that will not interfere with the installation
4. Minimum configuration requirements

Software requirements should state required prerequisites such as the expected toolset. It should also list programs that may conflict with the installation, along with references to instructions for resolving such conflicts.

Necessary resources, such as the amount of memory and disk space required, should also be listed.

2.6. Overview/Familiarization Sections

Most installations can be broken down into a handful of major steps. These steps, along with the approximate time each will take to complete, should be listed and explained briefly in this section. The division of the process into separate steps should be logical, which distribution developers should keep in mind.

2.7. Term Definitions/Explanations/Assumptions

Again, avoid breeding confusion. Terms that are likely to be unfamiliar to the installer should be defined in this section. Terms or uses of terms that may be confusing should also be explained. This section is also a fine opportunity to explain the difference between "bootstrapping" and "upgrading," for example. Once you establish these conventions, adhere to them.

By "assumptions" we mean the prior knowledge you are assuming the installer will have. For example, if an installer is going to have to rebuild a kernel and you do not plan to explain the process in detail, you should tell the installer that familiarity with the process is assumed. You should also provide references to documents that can inform the unenlightened.

2.8. Before You Start

At this point, the installers should know:

1. If they got all the necessary parts of the distribution
2. How to understand the installation documentation
3. Whether or not the installation will work on their hardware and software
4. What major steps will be involved in the installation
5. What unfamiliar terms and uses of terms they will come across

This section is designed to cover any remaining bases. It should discuss actions that need to take place but are not directly associated with the installation such as backing up user directories, saving local modifications or backing up the distribution tapes (we all know that backups are a good idea). These procedures are often fairly commonplace, so it isn't necessary to detail the procedure. However, references to detailed instructions are appropriate.

2.9. Step-by-Step Instructions

This is the important part. There are some basic rules to follow:

Format: Begin by *briefly* re-stating the major step listed in the Familiarization section. Then, list all the sub-steps, in order, obviously separated. Remember to actually follow the document conventions you have set.

Warnings: Make them look important. Very important. Use capital letters, put a box around it, boldface, italics, anything. Installers will NOT see warnings if they are buried in a long section of prose.

Explanations: It isn't a particularly good idea to leave the installer in the dark about what is actually happening. Informed installers are better prepared for trouble-shooting, bug reporting and upgrading. Explain what each step is supposed to do.

Intuition: Quite often, experienced and inexperienced installers alike will do things just because they seem appropriate or intuitive at the time. For example, they'll remove disks from drives just because they have stopped spinning. Fool-proofing is desirous, but difficult; never underestimate the ingenuity of a fool. Be aware that things like this may happen and guard against it with warnings, as mentioned above.

Decisions: Frequently, installers will have to make decisions about (either the course of or the final result of) an installation. Explain the costs and benefits of all decisions. Avoid phrases like "but you probably don't want to do that anyway."

Validation Points: It is particularly helpful if installers are able to periodically validate the progress of an installation. The sooner you discover a problem, the easier it is to fix. If this capability is provided, opportunities for validation should be indicated in the documentation. Additionally, the documentation should describe what will happen if indeed everything is proceeding smoothly and should provide references to trouble shooting information in less fortunate cases.

Checkpointing: It is often convenient for an installer to be able to checkpoint an installation – to stop the installation process temporarily. The documentation should point out the places in an installation where this is possible, and should include instructions for continuing an installation after it has been stopped, as well as references to trouble shooting information should a checkpoint fail.

2.10. Trouble Shooting

Because they do not necessarily need to be used by every installer, trouble shooting instructions should be located in an appendix.

Obviously, it isn't possible (or even desirable) to try to predict everything that could possibly go wrong with an installation ... the possibilities reach outrageous proportions in a matter of minutes. It is, however, possible to predict the nature of some installation problems. For example:

People make dumb mistakes: Typos and inadvertently skipped steps are to be expected. People, especially installers, are human. These errors should be easy to diagnose and fix. It is important to remember, however, that part of ''fixing'' involved expediently returning the installation to the state it was in before the problem arose, or by explaining how to re-start the entire process.

Failed Checkpoints and Validation Points: If you have provided a method for checkpointing and validating the progress of an installation, you should provide steps to follow should either fail. This may require diagnostic actions by the installer. The tough part here is remembering to *follow all the avenues*. Diagnostic procedures may return many classes of results, and information for dealing with each must be provided.

2.11. Quick Reference

In environments where the same installation may be done repeatedly, it's nice to provide a quick-reference section for installers who will have to complete the installation process more than once. This section should be in the same order as part 9 (listing each major step and its sub-steps chronologically), and should contain:

1. commands to type
2. brief (i.e., less than two dozen words) explanation of the process
3. appropriate warnings
4. checkpoint indicators

3. Additional General Notes

Accuracy

Primarily, installation documentation should be accurate. If it does absolutely nothing else, it should at least tell the installer the truth (see Horror Stories #4 and #11). Unfortunately, the astounding number of occurrences of inaccurate installation documentation indicates that it is a problem that needs to be actively avoided.

Format

Obtaining the installation documentation for a distribution should be a very trivial task for the installer. Obviously, distributing hardcopy is the best way to accomplish this.

It's also not a bad idea to provide the installer with the documentation before s/he gets the tape. The installer may just be inclined to read it thoroughly before diving into the installation process head first.[1]

For software that can be installed on a variety of machine types, it is generally not a good idea to sandwich instructions for each type in the same document, although many organizations try. What usually results is a document that is confusing and difficult to work with. The documentation should be kept as linear as possible. If the instructions for different configurations do not differ significantly enough to warrant the production of a different manuals, then sections that apply to only one configuration should be clearly identified so

[1] This technique has been used successfully by the distributers of 4.3bsd.

they can be skipped over in instances where they do not apply.

4. The Installation Document Development Process

Unfortunately, the outline alone is not enough to insure the creation of good documentation. Now that we have described what goes into a good installation document, we will discuss a strategy for the document development process.

To begin with, it is important to notice exactly how much of an installation document can be created long before the software is ready for release. Document conventions, hardware requirements, term definitions, and the "Before You Start" sections (at the very least) can be written early, when release dates and deadlines are still far in the future.

The rest of the document, of which the largest parts are the step-by-step instructions and the trouble shooting information, may actually end up being written at the last minute, but at least they stand a better chance of being useful when combined with the sections written earlier.

Document Testing

Even the most conscientious developer may make inadvertent errors, or their editor[2] may introduce or obfuscate important material. Source code errors can often be caught by simply compiling the source and executing the program, however, no such simple approach is possible for installation documentation. We present two methods for testing installation documentation.

It is common practice to test software before it is released. It should be an equally common practice to do pre-release testing of documentation. Alpha and beta releases in particular can be good opportunities in which to test installation documentation because installers of test releases are not expecting a flawless product.

In most cases, test releases are installed at remote sites by the one or two people in the entire world who are capable of it – namely the principal developers. Instead, test releases should be shipped with an installation document that is as complete and correct as possible. Test installers should be asked to keep a log of the installation process to record any and all problems that arise. These logs can be used to revise existing documentation.

Alternatively, documentation can be tested at home. At least two Guinea pig installers can attempt to at least execute the installation. One should be very familiar with the product and able to correct errors or rectify oversights. The other should be someone who barely meets the minimal installer qualifications and is largely unfamiliar with the product. The latter test should be passively observed and problems or blind alleys should be noted for later correction.

We have presented two methods for document testing. In the interest of time, we have reduced the testing process to the bare minimum. If time permits, it is strongly suggested that the document be tested with additional subjects whose skill levels are somewhere between complete novice and expert.

5. Conclusion

The guidelines that have been discussed are still largely incomplete, but we have begun to apply them to internal projects. For example, the guidelines helped us evaluate documentation we were preparing for exportation. To ensure their validity, we are currently using them to develop an installation document for a major ITC product to be delivered later in 1988. This activity will bring about the extension and revision of the guidelines and suggest further strategies for effective creation of such documents.

The authors wish to stress that, although we discuss the need for adequate documentation, we do not necessarily feel that this can only be accomplished by professional writers. However, good documentation does take time. And, as the adage says, time is money. Therefore, good documentation requires a commitment from the management of a software development organization.

---

[2] Both types – human and program.

6. Collected Horror Stories And Their Morals

This was the fun part. The following are extracts from some of the interviews conducted as part of this research. Each is presented as an illustration of an Installation Documentation failure, and the associated moral.

1. *"We were installing this huge statistical package on one of our mainframes. We had been plowing through it for a good three hours, then we realized that we were supposed to have gotten two magnetic tapes from the company, instead, we'd received just one. We had to stop everything, call the people, wait two days for them to ship out the second one, and then start over."*

Moral: Stop problems before they start – include a list of the contents of a distribution.

2. *"I was trying to install a couple of device drivers on my PS 2/30. It should have taken about half an hour. After three hours of wading through five different manuals, I finally got it finished, but I wasn't happy."*

Moral: Design documentation to be as linear as possible. Minimize gratuitous references, except in the interest of trouble shooting or other unpredictable problems.

3. *"I made a really dumb error. I mean, so obvious, it hurt. Unfortunately, I missed the warnings about it in the documentation ... they were kind of hidden."*

Moral: Make warnings look important!

WARNING: To guard against disaster, it is very important to make backup copies of the user file systems before proceeding.

4. *"The documentation promised commands that didn't exist."*

Moral: Put accuracy above all else, even if you have to kill some programmers.

5. *"It took twice as long as it should have. The format was messy. I kept reading from the sections for different hardware and typing the wrong things."*

Moral: Design linear documentation. Put instructions for other machine types elsewhere, even if it results in repeated text.

6. *"I skipped the whole thing. It was too busy telling me things I already knew."*

7. *"The whole thing is gathering dust in a corner, we can't make heads or tails out of the documentation."*

Moral: Produce documentation that can serve the needs of more than one audience. People who consider themselves "experts" will invariably try to do installations magically, without reading documentation. People who consider themselves novices will rely on documentation as if it were a life-support system.

8. *"Well, after three days, we kinda ran out of ideas and gave up."*

Moral: Provide trouble-shooting information, and if possible, an additional resource for really big problems.

9. *"It just lists lines to type. If you get an error, you have no idea what was supposed to happen, never mind what actually did happen."*

Moral: Don't leave the installer in the dark. Informed installers are better prepared for trouble-shooting, bug reporting and upgrading. Documentation should explain what each step is designed to do.

10. *"We had a UN*X expert there to do the installation for us. He got everything running, but he quit the window manager before he tried CMUtutor, which naturally died without Andrew underneath it."*

Moral: Guard against mistakes that may seem "intuitive" to an expert.

11. *"We spent thousands of dollars for the PC network because the documentation says it will work with our configuration. Too bad it never told us how..."*

Moral: Provide documentation for ALL hardware configurations you claim to support.

12. *"We were installing a very high speed line printer and had begun to test it. Unfortunately, no where did they mention that the cover had to be down when the paper drive was turned on. Paper flew out of the printer so fast that it literally knocked someone down and broke his arm".*

Moral: Remember Murphy. Never assume that what's obvious to you is obvious to others. And finally, check your malpractice insurance.

This work was funded in part by the Carnegie Mellon University Center for the Design of Educational Computing Scholars Program.

Thanks to Thomas Lord for invaluable assistance in the preparation of this paper.

References

This space intentionally left blank.