Cherryleaf

Tips for writing great technical documents and Help files

www.cherryleaf.com

01784 258672

Cherryleaf Ltd
Cherryleaf House
31 Arlington Road
Ashford
Middlesex
TW15 2LS
United Kingdom
+44 (0) 1784 258672
info@cherryleaf.com
www.cherryleaf.com

Table of Contents

Planning user documentation - a guide for software project managers

Introduction

A Guide to the Project Management Body of Knowledge (PMBOK® Guide)–2000 Edition is the main sourcebook in the project management field. Whilst it covers Project Communications Management, it doesn't extend to user documentation.

This article seeks to provide guidance for project managers as to how the user documentation process fits in with the overall project planning. It examines:

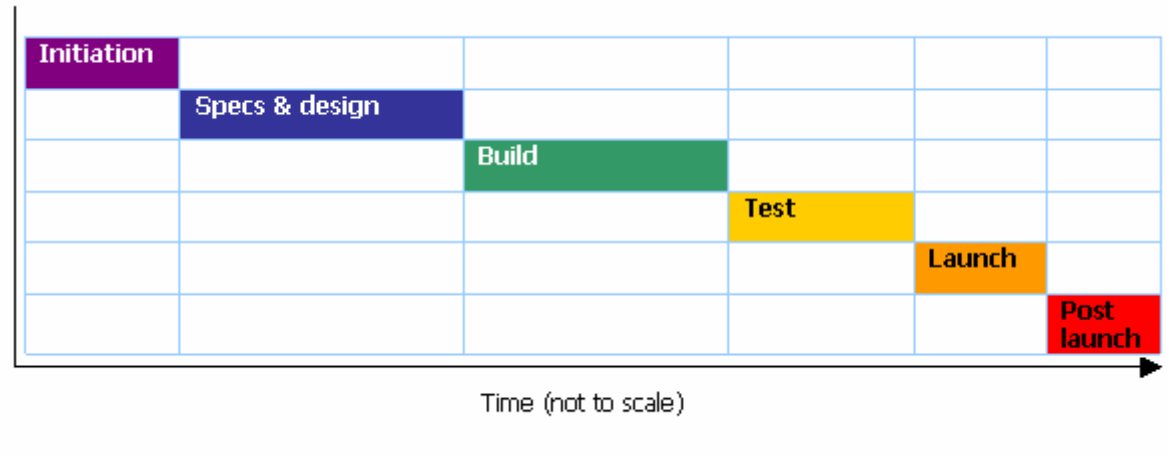
1. the traditional way documentation is approached and how it impinges on project planning
2. the effects of making changes to this traditional approach.

What do we mean by 'documentation'?

When talking about documentation for a software application, most people think of the traditional paper-based or online manual. In this article, we will be looking at ways of taking this traditional approach forward. The term 'user assistance' then becomes more useful, implying 'anything that makes the user's life a bit easier'. More on this later…

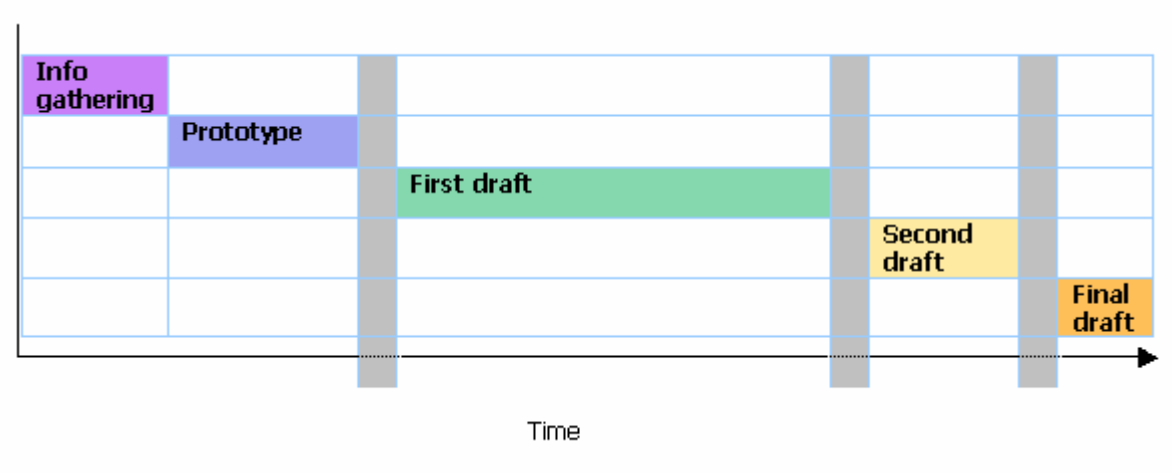
Your project plan

A typical plan for a software project might look like this:



A typical documentation project plan

A **traditional** documentation project comprises the following stages:



The vertical grey bars on the above diagram represent reviews of the work done. These bars are not to scale. Reviewing can take a lot longer than you might think and must be done by somebody with expert knowledge of the software.

Prototype

The prototype comprises a structured set of empty topics (pages of information) with a prototype look-and-feel. For each different type of topic, one is populated with content as a sample.

The review is to determine completeness and suitability of the structure together with suitability of the look-and-feel, writing style and writing conventions.

First draft

During the first draft stage, the author writes the content for all the topics. This is the longest stage and takes the longest to review.

The review is to check for technical accuracy of all the content.

Second draft

The second draft stage is for the author to make the necessary changes, reported from the first draft review.

The review is to confirm that this has been done and to request any minor tweaks still outstanding.

Final draft

This final stage is to follow up on the minor changes picked up at second draft review. The result is the final deliverable.

So where does the documentation fit in with my project plan?

Traditionally, the documentation comes in towards the end of the **testing stage** of your project. The software is more or less stable and you're close to launch. Having seen the stages involved in the documentation, you may be thinking that it's a lot to cram into this stage. You'd be right! This is why

many project managers see documentation as that necessary nuisance that takes up valuable manpower to create and review just when deadlines are looming.

So it is also vital to consider the effects of the documentation at the **initial planning stage**.

In this first stage of your project, your planning will include such things as:

- Timescales and resources
- Roles and responsibilities

Timescales and resources

When thinking about timescales, don't forget the time required for the documentation! It may seem obvious, but effective documentation takes time to develop. As seen in the typical documentation project plan above, the elapsed time can be much longer than just the writing time due to the review process.

Depending on the scale and complexity of your software, you may have to consider splitting the documentation between different authors. Authors working in parallel can speed up the process, but be warned – depending on the tool(s) used to create the documentation together with the associated workflow this may not be possible. In addition, there is the extra issue of keeping the documentation consistent across the authoring team.

As well as providing the writing resources, it is just as important to provide adequate resources for the reviewing and a documentation project manager to coordinate writers and reviewers. All this must be factored into the cost.

With so much to do in the testing stage of your project plan, the inclusion of the documentation at this stage and the drain on expert staff for reviewing presents a real project risk if not properly planned and resourced.

Who should write the documentation?

If your organisation has a specialist documentation department then your solution is simple.

So what are your options if this is not the case?

The programmers themselves

Seems logical. They certainly know the product inside out. But beware. The programmers are experts in programming – not necessarily in communicating! They may well assume a level of knowledge that the end-user does not have. Not only that but their enthusiasm for their creation often gives them an understandable tendency to explain every last technical feature in detail – focussing on product functionality rather than answering users' questions.

A contracted technical author

This is often a good solution, though it very much depends on the rest of your project plan. In the **build** stage of your project plan, you may be setting aside long stretches of time on the coding of the software between tests. Unless there are other projects to work on, a technical author can be left idle and compromise your budget.

Outsourcing

Contracting an external organisation to project manage and develop your documentation means that you will only get charged for the days spent managing and writing. This solves the problem of idle

time, but does have a couple of disadvantages. One is that the author spends most of the actual writing time off-site so direct communication may be more difficult. The other disadvantage is that it may not be possible to guarantee that one particular author will be available if timetables and deadlines change.

Who should review the documentation?

You may want or need more than one reviewer. As already mentioned above, all the documentation at every stage must be reviewed by at least one expert in the software. This ensures fitness for purpose and technical accuracy. In addition, it is often useful to get other relevant people to review parts of the documentation. One example is if you have access to some helpful end users. Their comments can be very valuable. Another example is if there is to be a training programme for the software. The training staff can provide useful feedback.

Who should manage my documentation?

As stated previously, the documentation project manager is necessary to co-ordinate the writing and the reviewing process. Unless your programmers are writing your documentation, many questions will arise on the author's part about the details of how the software is to be used. So part of the project manager's role includes co-ordinating these questions and their answers between the author and the programmers.

If you are outsourcing, the external company manages the delivery of the drafts, the receipt of review points and the author's questions for the programmers. However, there should still be a person named as the documentation project manager in your organisation to provide a single point of contact for the external document project manager. Without this role, review points get fed to the external company in dribs and drabs and sometimes contradict one another.

Spreading the authoring work more evenly

The traditional close-to-launch timing of the documentation presents, as mentioned already, a project risk. To reduce the amount of time and resource required at this busy time, there are a number of tasks that your technical author can do **earlier** on in the project to reduce the project risk.

Look-and-feel of the Help system

These days, authoring tools for technical writers are making it possible to create Help systems with a customised look-and-feel. Software houses are now getting more interested in their Help systems looking better than the rather grey-looking standard Help viewers (for example WinHelp and HTML Help) and more in line with the software itself. Creating a customised look-and-feel takes time. But this is one of tasks your technical author can be doing in the **design** stage, as soon as the software look-and-feel is agreed.

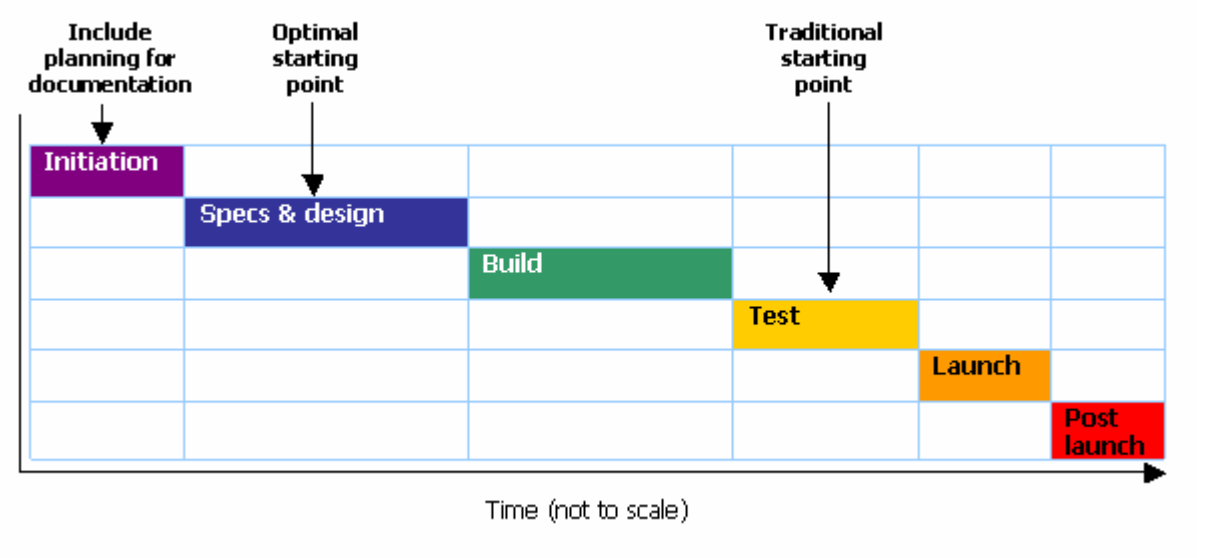
Structuring the Help system

Depending on the complexity of the software, it may be possible for the technical author to sketch out a rough structure for the topics in the Help system after the **design** stage. This would be a flexible work-in-progress structure, but would at least set out a draft of the types of information to be documented.

Conceptual topics

Generally speaking, the majority of Help topics in a Help system are procedural. They tell the user how to do something by giving detailed step-by-step instructions on using the interface so, in the main, are best left until the software is more or less stable. However, there will also be some information for the user that is of a conceptual nature. It may be possible for your author to document some such concepts during and after the **design** stage.

Summary of where your documentation fits into your project plan



Getting more value from your technical author

Specifications and design stage

It may not be something you have considered, but since good technical authors have expertise in designing information, they can also provide **useful input** at the **specifications and design** stage. Getting your technical author on board at this stage can help make the user interface more usable. This reduces your technical support costs and may reduce the time it takes to document the software. It may be something as simple as labelling or ordering the fields on the dialogs in a more helpful way. Moreover, authors with experience in writing for screens will know what sorts of layouts will and won't work. They can provide valuable input for decisions on the structuring and the look-and-feel of the interface. And remember that they are very much user-focussed – providing balance to what may be an over-enthusiastic graphics design team! You can think of the technical author as the representative of your average user as the specifications are decided and the design progresses.

Embedded Help

If your technical author is contributing expertise during the **specification and design** stage of the project, it may well be that the traditional paper manual/online Help 'documentation', created at the very end, is reduced in scope or even unnecessary. In other words the 'necessary nuisance that takes up valuable manpower to create and review just when deadlines are looming' is all but eliminated. The user interface can contain its own 'user assistance'. This kind of approach is known as Embedded Help. Exactly how and where the assistance is provided is part of the specification and design process. For example, one successful technique is to have a dedicated (but shrinkable) part of the interface for displaying context-sensitive assistance.

Build stage

If present at all relevant meetings, the number of questions your technical author will have for the programmers when writing the Help content will be reduced, saving precious time near the end of the project.

Testing stage

If you are planning to perform usability testing, your technical author can participate in designing and giving the tests and in interpreting the results.

Post rollout

If you are outsourcing, the external organisation may provide a service for logging and collating user feedback, reporting it to you and updating the Help where appropriate.

Summary

Providing online assistance for software is important since it can reduce costs of technical support and make life easier for your users. However, documenting software is not a trivial task. It requires time and resources for writing and reviewing that must be included in the overall project plan.

The documentation work is traditionally implemented after the software is reasonably stable – during the testing stage. However, getting your author on board earlier spreads the work more evenly, reducing project risk. Moreover, including your author in design decisions and usability testing can improve the user interface.

The Cherryleaf user documentation questionnaire

How effective is your current user documentation? Complete the following questionnaire to check you are creating new sales opportunities & maximising customer loyalty for your product or service.

	Do you have some of these problems?	A problem we are experiencing today	Not a problem worth doing something about today
1	Keeping hold of customers	YES/NO	YES/NO
2	Customer support is costing too much	YES/NO	YES/NO
3	Poor documentation is holding back sales	YES/NO	YES/NO
4	Users find it hard to use the software	YES/NO	YES/NO
5	Time taken to produce documents is too long	YES/NO	YES/NO
6	User documentation isn't effective at supporting users	YES/NO	YES/NO
7	Difficulty meeting your delivery deadlines	YES/NO	YES/NO
8	Lack of staff to develop the documentation	YES/NO	YES/NO
9	The same content is being reproduced time and again	YES/NO	YES/NO
10	Staff who write the user documents are not sufficiently productive	YES/NO	YES/NO
11	Quality of output is variable / inconsistent	YES/NO	YES/NO
12	Conflicting messages are appearing in documents	YES/NO	YES/NO
13	Output is not personalised enough	YES/NO	YES/NO
14	Subject Matter Experts are tied up in production	YES/NO	YES/NO
15	Documents don't meet the needs of the customers	YES/NO	YES/NO
16	Development of the documentation cannot be shared out and developed by a team	YES/NO	YES/NO
17	There's no system for producing user documentation	YES/NO	YES/NO
18	You don't understand the users	YES/NO	YES/NO
19	There's no tracking of projects	YES/NO	YES/NO
20	Staff are not sufficiently skilled	YES/NO	YES/NO

Send us a copy of your answers to these questions for suggestions as to the best actions for your situation.

Top 15 tips to writing great Help files

We asked the Technical Authors at Cherryleaf for tips on writing Help files. These are their top 15 tips:

1. Think about what your audience needs/wants

Talk to your users. Find out what your users struggle with, what they already know, what kinds of information they crave, what tasks they perform most frequently, what vocabulary they use, what they are doing before, during and after using the system.

2. If you can't talk to your users, talk to your Help Desk and Training departments.

3. Plan your Table of Contents before you start writing

4. Structure the Table of Contents, so that you users will find it is easy to navigate

Don't bury topics too deep - the user will never find them.

5. Write to a level of detail that is appropriate for your audience.

Where possible, keep you topics short and to the point

6. Name your topics sensibly so that a user can look at them and know what information they'll find within a particular topic

"General information" or "Miscellaneous" are no use to anyone.

7. Have just one procedure per topic

8. Address your instructions directly to the user - avoid the passive voice

9. Don't mix conceptual information with instructional information

10. Use a style sheet for consistency

You can often use it again as a base for other projects, thus saving you time.

11. Use popups / expanding text / drop-down text to hide any information that will be optional for the user

12. Avoid using indentation to indicate sub-sections

Use vertical white space instead.

13. Spend time on the index, as many users will look for topics here

14. Include synonyms in your index

15. Avoid use of large screen-captures. Instead, clip the image to show only the part being described

Using a professional to develop good user documentation - who benefits?

Half of all investment today is in information and communication technologies and, as a result, the importance of IT projects is huge. However, IT projects have an unfortunate reputation among the public for failing to deliver. If we are to improve this performance, then we need to look at how to improve how a project is managed and resourced. Studies have revealed that **up to 40% of all information systems developed have online Help written by a non-trained writer**, and we believe this is one of the reasons why so many IT projects fail.

Is your project likely to succeed or fail?

Studies have shown that, in the UK, **only 16% of IT projects hit all their targets**. While performance is improving, the challenges are becoming greater.

Some of the main concerns we hear from project managers are:

1. Completing their project on time.
2. Completing their project to budget.
3. Finding the right team.
4. Making users productive:

- Training their end users in new procedures.
- Providing incentives for people to change.
- Overcoming resistance to change.
- Communicating the reasons for change.

You can address these issues by working with technical communicators.

Help others to understand what's going on

One of the most essential deliverables that IT professionals must provide is documentation. This is because:

- **50% of projects involve business process change**, organisational change or work design. People need information and training that enables them to be effective in their jobs. This means capturing best practice and specialist knowledge, recording it clearly and communicating it to the relevant people in the most effective way.
- **30% of projects involve package modification**, and many organisations want to expand the usefulness of built-in Help and error messages to include information unique to their situation.
- Documentation transfers information to your clients and users so they can access and understand the system now and in the future.

Technical communicators:

- Spend significant amounts of time analysing users and helping them understand the new system.
- Communicate the needs and views of the users back to the developers.
- Contribute to the design of the user interface, assisting with the usability of the system.

The result:

1. The customer gets a better product with a reduced cost of ownership - because they can use the online assistance to be more productive.
2. You are able to create a system that is more relevant and usable.

The Sales Process

Good documentation helps the selling process:

- Good documentation leads to better customer satisfaction and helps you to **retain your customers**.
- It is one of the main ways your corporate image will be presented to the client during the life of the product.
- Apart from the software itself, the users documents and Help are often the only things the client sees, once the salesman has left and the packaging has been thrown away.

The sales team will be able to show prospects that :

- Your documentation meets the customer's requirements.
- Your documentation meets the users' requirements.
- Supporting customers after the sales is important to you, and your organisation does this professionally.
- If users get stuck, the online user assistance system will enable them to get unstuck quickly and easily.
- Your organisation understands what its customers want to do i.e. the tasks they want to complete.

The Training Team

- Any training can focus on the key points, and these will be reinforced after the training by just accessing the Help system.
- CBT and animated examples can be included to provide performance support and just-in-time training.

The Development Team

- The software development team will be given a user's perspective on how easy it is to use the software, and what concepts and processes are difficult to understand.
- Help can be embedded into the application, making it easier for people to use.

The Technical Support Team

Call to support desks can be reduced by:

- Including commonly asked support questions in the Help system.
- Linking the Help system to a support Web site.
- Providing a natural language search engine in the Help system.
- Embedding Help into the software and making it more intuitive to use.

Users of systems with Help written by technical communicators use the online Help up to 54% more often, and they rate the quality of the Help more highly than do users of Help files written by someone who wasn't a professional Technical Communicator. In one study, 210 support calls were saved annually per client, which represented an **annual saving of $2 million**.

Context-sensitive Help with compiled Help systems

What is context-sensitive Help?

Context-sensitive Help is assistance that is appropriate to where the user is in the software application, and what they are trying to do.

Who is involved in creating it?

Context-sensitive Help requires that the Help author and the programmer join forces.

- The programmer creates a means of calling the Help from each dialog of the application. This could be via a Help button or a right-click menu option.
- The Help author creates appropriate topics within the Help file.

It is strongly recommended that both parties meet up early on in the project to double-check that both are clear on the process and role responsibilities.

Who does what?

To implement context-sensitive Help we need to map Help-related objects on the application interface to appropriate Help topics. This mapping is achieved by way of a **header file**.

There are two ways for the Help author and the programmer to work together to achieve context-sensitive Help:

1. The programmer creates a header file and sends it to the Help author. (This is the more common situation.)
2. The Help author creates a header file and sends it to the programmer.

So what exactly is a header file?

Each object on the application interface has a unique number associated with it (the **context number**). This number is what will be used by the application to find the correct Help topic.

However, it is much easier for humans to deal with descriptive names, rather than numbers. For example, the **context name** `CTX_SALES_DIALOG` is much easier to understand and recognise than the context number `2384`. However, the application and the compiled HTML Help ultimately require a number.

The **header file** is simply a list of all the context names and their corresponding context numbers. It is a simple text file whose layout depends on the programming language being used to create the application.

Language	File extension	Syntax	Example
C	.H	#define *string number*	`#define CTX_SALES_DIALOG 1000`
Pascal	.PAS	*string* = *number*;	`CTX_SALES_DIALOG = 1000;`

Basic	.BAS	*string* = *number*
		or
		GLOBAL CONST
		string = *number*
			`CTX_SALES_DIALOG = 1000`
			
			`GLOBAL CONST CTX_SALES_DIALOG = 1000`

The header file is not used at runtime. The information it contains is included in the Help system on compilation.

When the programmer creates the header file...

… the **programmer** must:

1. Produce a list of the context names and numbers for all dialogs and fields that require context-sensitive Help– the header file.
2. Send the header file to the author.

… the **author** must then:

1. Import it into the Help project.
2. Set up the context names in the header file to be **aliases** for the corresponding topic names. This literally means giving the topics alternative names.

When the Help author creates the header file...

… the **author** must:

1. Create context name aliases and context numbers for all dialog level context-sensitive topics.
2. Create context numbers for all field level context-sensitive topics.
3. Produce a list of the context names and numbers – the header file.
4. Send the header file to the programmer.

…the **programmer** must then:

1. Import it into the application.
2. Associate all the context names in the header file with the corresponding Help-related objects in the application.

Writing the content

It is important to decide what sort of information to include in the context-sensitive topics.

Dialog level

Dialog level Help usually provides one or more of the following:

- A general description of the dialog.
- A description of the various fields within it.
- Links to topics that provide Help on the tasks relevant to the dialog.
- If the dialog has only one task associated with it, step-by-step instructions on how to complete the task.

Field level

Field level Help usually provides a short topic with information about the selected field. This is usually in a popup window and is known as 'What's This?' Help.

When providing popup field level Help using Microsoft HTML Help

In this situation, the topics must be written in a **text file**. HTML Help does not support the use of graphics and formatting in popups. Even if you are using an authoring tool that has a method for producing formatted popups, this functionality is not applicable in a context-sensitive environment.

Write your text-only topics in a text file, using the syntax:

.topic *context name*

topic text

You can use more than one text file, but all the topics for any particular dialog must reside in the same text file. Since the context names are hard-coded into the text files, there is no need for aliasing.

The text file(s) must be included in the compilation of the Help project.

Example

A text-only topics file for the sales dialog in an application might start of like this:

```
.topic CTX_SALES_DIALOG_ADD_BUTTON 1010
Click this button to add a new sales item to the list.

.topic CTX_SALES_DIALOG_ADD_SALENAME 1020
Type in the product name using the correct capitalisation.

.topic CTX_SALES_DIALOG_ADD_SALEDATE 1020
Type in the date that the sale was confirmed in the format DD/MM/YYYY.
```

Help authoring tools and context-sensitive Help

Different authoring tools have different interfaces for dealing with header files and text-only topics.

Importing a programmer's header file

- Some HATs allow an automated import of the programmer's header file.
- Others require you to (laboriously) type in the context names and context numbers by hand for every context-sensitive topic.

Associating the context names in the programmer's header file with the corresponding Help-related objects in the application

- Some HATs have a simple point-and-click interface for setting-up context name aliases for topics – thus keeping the aliasing process hidden behind the scenes.
- Others require you to know about aliases.

Creating a header file for export to the programmer

- Some HATs hide the aliasing process from the author.
- Some HAT's can automate the context numbering.
- Some cannot directly create the header file. Authors must open the fundamental project text file (.HPJ for WinHelp, .HHP for HTML Help) and copy the relevant information into a new file!

Creating text-only topics for HTML Help

Although these can be created in any text editor, some HATs provide a simple interface for writing them so that the text files are automatically added to the Help project and compiled.

Some produce the required text files behind the scenes as they compile HTML Help.

Help authoring tools and context-sensitive Help

Problems often arise in communication between the Help author and the programmer due to there being inconsistent terminology for the elements involved. Confusion also arises within the Help community due to different authoring tools using different terminology.

The table below shows some of the names that programmers, Help authors and authoring tools use to refer to the elements of context-sensitive Help.

Term	Synonyms
Header file	• map file • mapping file
Context number	• context ID • map ID • map number • mapping number • UID (unique ID) • GUID (globally unique ID) • Help ID
Context name	• context string • symbolic constant • constant • define • macro • topic ID

Summary

Well-written context-sensitive Help gives users the information they need at the point that they need it.

It involves a simple mapping process:

context number > **context name** > **alias** > **Help topic**

However, problems may arise in its implementation due to:

- inadequate communication between programmer and Help author.
- confusion over terminology.

It is well worth making the effort to overcome these problems, since it gets us nearer to the just-in-time ideal of user assistance.

10 tips on single sourcing

We asked the consultants at Cherryleaf for tips on implementing a single sourcing solution. These are their top 10 tips:

1. Define your reasons for reusing information

Configuring information for single sourcing can be time-consuming. You need to know what you stand to gain by putting in the effort.

2. Define your level of granularity

Granularity of information is key to reuse. To customise content to be used in varied publications you need to:

- Represent information differently
- Exclude content
- Alter the purpose of information.
- Be consistent in the way documents are broken down into granular chunks

To maximise flexibility for these changes you sometimes need to identify quite small units of information. The smaller these units, the more granular your information is. But if they're too small they become hard to manage.

3. Separate format from content

If you're not doing it already, start separating format from content. Use Word styles, FrameMaker paragraph formats or cascading stylesheets. It will then be much easier to format content appropriately for each publication.

4. Name your styles (or paragraph formats) semantically

It will be more useful to your single sourcing process to know that a word or phrase is, for example, a User Interface control than to know that it is big, blue and bold.

5. Try to implement top-down writing when describing concepts

Top-down writing is similar to journalistic writing. Try to encapsulate the whole topic in the first paragraph then progressively increase the detail as you move down the page. This makes it easier to produce a shorter version if certain publications need it.

6. Learn about XML

Not all single sourcing approaches are XML-based, but it is the single most important technology in the field

7. Learn about DITA

In particular learn about the DITA XML schema- reuse is at the heart of DITA.

8. Think about which media output is most important to you

Although single sourcing is all about producing more than one output from one set of information, solutions often favour either online or printed outputs. Think about which type of output is most important to you, and choose a solution that favours it.

9. Carry out an information audit

Carry out an information audit (as defined by Ann Rockley). This will show you what information you have, how many people are writing content, how much content is common to the documents you produce, and how it might be reused. It will help you assess your return on investment. Cherryleaf can help with this.

10. Talk to other departments in your organisation

It may be easier to build a case for single sourcing if information produced by other departments is included

12 tips on hiring a technical author

We asked the consultants at Cherryleaf for tips on hiring a technical author. These are their top 12 tips:

1. Write a clear job description

Be clear about what qualities you want in a technical author. Write a clear description of the position, the project and what you want to achieve.

2. Think about the type of personality you're looking for

Communication and interpersonal skills are an important part of being a writer, as they may need to glean source information from very busy people. If you have a team of writers, then check whether they would work well with the existing team members. Does the candidate understand your goals? You should pick a writer that you feel comfortable to build a working relationship with, someone you like and trust.

3. Set a realistic budget

Be realistic about what you pay a technical author. They have highly specialised skills and can command decent salaries, if they have the experience. If you're just starting out in business, costs are big issue, but remember cheaper isn't always better.

4. Check they have a track record

Does the person have a good history of solving the issues you have? Technical authors pick up software quickly; it's part of their job. Just because they haven't worked on exactly the same type of environment as yours, doesn't mean they won't become "expert level users" very quickly.

5. Listen

Listen to advice for your technical writer. Remember that they are there to help you gain the users' perspective, and they can help you avoid problems in the future.

6. Think about your opinion towards age and experience

If you have the resources to train/mentor it can be worth taking on a graduate. If you don't have these resources, there's a lot to be said for experience.

7. Permanent or contract?

Do you need a contractor or permanent member of staff? It sounds obvious, but you might not have thought through the benefits of each for your particular situation.

8. Tools of the trade

Candidates should have experience of the software commonly used by technical authors, such as FrameMaker, RoboHelp, Word or an XML authoring tool. However, it's easier to teach a good writer how to become an expert in an authoring application than the other way round. Don't rule out a promising candidate applicant just because they've not used your favoured tools.

9. Look at any samples of work

Look at the overall structure of the documents. Is there a logical progression of information from the beginning to the end? Structuring a document well is one of the most important skills needed in a writer. You should also look for an awareness of a documentation process and a wish to follow it.

Can you spot any typos or inconsistencies within the document? Is the writer aware of these? Is there a reason why they crept through the quality control process (such as having been produced to really tough time constraints)? Was the document written exclusively by the candidate?

10. Technical experience and skills

The writer needs enough knowledge to explain the subject; they don't need to be able to actually do it. Do they have the ability to learn? Could they gain an in-depth understanding of a subject?

11. Ask around

Find out from friends and colleagues who they use and if they would recommend them.

12. Use a specialist agency

When it comes to technical authoring positions, most agencies don't know their AuthorIT from their elbow, so it's best to use a specialist technical authoring recruitment agency (such as Cherryleaf) that can send you candidates who fit the bill.

Reducing translation and localisation costs

Introduction

Question: How do successful organisations get their message across to different cultures?

Answer: very expensively!

Translating documents takes a great deal of time, effort and money, issues that multiply with every new language and version update you produce. Translation comes at a high price, exceeding the cost of writing the original content after only a few languages. In recent years, we have seen only limited improvements, with the main change being the drive to reduce the cost per word to its lowest possible rate. It is an industry that is under increasing pressure to find new ways to improve cost-efficiency, quality and the time-to-market.

These days, staff in localisation departments spend their time essentially on project management, translation and quality assurance. However, by using one of the emerging systems that integrate content creation, localisation and content management into an efficient system, many of these activities can be automated or avoided all together. We are now seeing the emergence of technical content control systems that can be used to improve the turn-around time, translation costs and the quality of the translations themselves. In recent projects, where these systems have been implemented, organisations have seen substantial savings in localisation costs, with word count reductions and translation costs of around 30%.

Reducing repeated translations

How have these improvements been achieved? The answer is by using single-source technical content systems that make intelligent use of XML. These systems know what content is translatable, what had been previously translated, and was has been reused, added or changed. It means that only content that requires translation is sent to translators.

In addition to costs for translation, localisation companies often charge for scanning text that has already been translated. This cost is now eliminated, because these new systems submit only content that is completely new content or has been changed.

Breaking document into components

The biggest change is that, in these systems, the Technical Authors write document components rather than entire documents. A component could be a procedure, a process, technical data etc. and can be reused in different manuals. A master document then holds links towards the different components that make up the manual. When similar content is found in different documents, the small differences can be published conditionally. This limits considerably the number of modules that need to be managed.

This re-use of content reduces the size of the documents that have to be translated.

This approach enables you to translate each component as soon as it is ready. The system knows what content is translatable, what has previously been translated, and what has been changed. It only submits components as and when they are ready.

The systems we advocate use XML in a way that allows translators to use their favourite translation memory tools. The software tracks and manages this source content as it is translated into each language. They export the text to XML in a format-neutral mark-up, which increases accuracy by eliminating the effects of formatting on memory matches. They can translate the content, and then return it to the system intact.

Translators are given a document that highlights the content that appears in the XML files. These documents ensure the translators have the required context for translation, avoiding the need for unnecessary reviews.

Summary

It is encouraging to see that new products are coming on to the market that actually work and enable the localisation process to happen more swiftly and accurately. By using single-source technical content systems that make intelligent use of XML you can reduce the size of the documents that have to be translated. This leads to cost benefits, saving of time and better project planning. More time will be required at the beginning of the project to set up the document correctly, which is well worth the effort for the rewards.

About Cherryleaf

Cherryleaf Limited has locations in Birmingham, Brighton, Heathrow (London) and Thame, although we carry out work throughout the UK and the rest of Europe.

We explain things. We work with:

- Developers of software who are afraid of losing their customers and frustrated with the cost of supporting them.
- Anyone who needs to create straightforward, easy to use information and get it to the people who need it.
- Developers of software who lack the time and resource to document their systems adequately.
- Companies who need help organising, writing or delivering their policies and procedures.
- Technical communicators who are looking to improve their skills, find a new job or need help with their processes.

We are technical writing specialists, also known as technical communicators, documentation specialists, technical authors and technical writers.

Cherryleaf Ltd
Cherryleaf House
31 Arlington Road
Ashford
Middlesex
TW15 2LS
United Kingdom

+44 (0) 1784 258672
info@cherryleaf.com
www.cherryleaf.com

Index

Cherryleaf Ltd
Cherryleaf House
31 Arlington Road
Ashford
Middlesex
TW15 2LS
United Kingdom
+44 (0) 1784 258672
info@cherryleaf.com
www.cherryleaf.com