

Docbook XML

The Source for Documentation

Part 1 of 5

By: Sean Wheller
sean@inwords.co.za

Table of Contents

Introduction.....	3
The Documentation Challenge.....	4
Documentation Process.....	4
Documentation Tools.....	5
Documentation Formats.....	6
Documentation Processes	6
Documentation Storage.....	7
Standardized Structuring.....	7
Exclusive Offer for “123 Gift for Me”	10

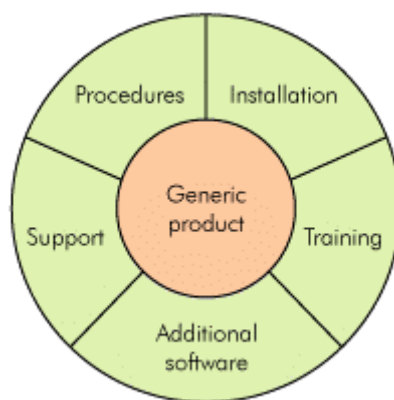
Introduction

Marketing technology products can be very lucrative when the product developer gets it right. This is especially true in the software industry, where, once a product is developed, the costs for production are low in comparison with other industries. Every year, hundreds of technology startups rise to the challenge of building the next "killer application," aiming hard to position a leading product, looking forward to that "pot-of-gold."

Of the hundreds that start, few ever succeed, for the route to product development is fraught with obstacles, challenges and lessons to learn. One of the lessons technology vendors are late to learn is that of "whole product" development. To build a "whole product," developers must not only build quality software to ship to the customer, they must also augment the software with a friendly Service Level Agreement (SLA), professional services, technical support, training, installation instructions, manuals, cables, additional software or hardware if required, and anything else the target market needs in order to feel comfortable and confident about buying and using the product. The development of a "whole product" is often the breaking point for community-based, open source technology projects and the main reason why corporate users consistently choose proprietary products over their open source counterparts.

The concept of "whole product" is not new, nor is it our invention. It was first formally introduced by Theodore Levitt in "The Marketing Imagination" first published in 1983. It became popular in the high tech arena with the publication of Geoffrey A. Moore's "Crossing the Chasm" in 1991. From these publications the concept of the "whole product doughnut diagram," shown below, has evolved as a visual representative of the concept.

Figure 1. Whole Product Doughnut Diagram



Considering this diagram, it is easy to understand why the process of developing and maintaining technical publications should be a core function of product development and a key ingredient for taking a product to market. To varying degrees, each of the augmenting areas can benefit greatly from the use of documentation to consistently and cost-effectively satisfy customer needs for information. Thinking of the "whole-product" and documentation in this way, it stands to reason that, amongst the many things product developers must get right, documentation is one of the most important ingredients in making a product successful and profitable.

Recognition that documentation plays a pivotal role in all aspects of the "whole-product," including the "generic product," is the basis upon which to start conceptualizing and engineering the processes by which documentation development and maintenance will take place within a technology vendor organization. To do this we need to take into consideration the factors inherent to developing and maintaining information without losing sight of the motivation for producing documentation that will support the "whole-product."

As technologists we often fall into the trap of first focusing on the product we are developing and the "how we will package documentation." In our search for solutions, we often look for that which

seems to deliver the most immediate way of delivering, when what we should be doing is looking for are processes and technologies that support the development of information before it is packaged.

It may sound like a search for the "Holy Grail," but if documentation is to be developed in a manner that supports the "whole-product," we need to identify a more holistic approach to documentation development. One that scales well for small and large organizations alike, is flexible to customer requirements, future technology changes and is capable of providing consistency of workflow spanning all aspects of the pre and post-sales processes.

It all sounds like a pretty tall order and yet, considering the value of documentation in context of a "whole-product" development strategy and the inherent risks of marketing technology products, it is an order we must fulfill. It is precisely for this reason that we have written this document. We know it will not give you all the answers to your particular situation. We do hope it will serve to introduce some new ideas, perhaps give you ideas of your own. If reading this document gives you a platform from which to make informed decisions about the future direction your documentation development and maintenance practices take, it will have served its purpose. Of course this document is also a marketing tool for us at InWords - the open documentation company [<http://www.inwords.co.za>]. So if you decide to adopt the idea presented, we would like to ask that if we can help you, in some nice way, with any of the ideas we've shared, we hope you will let us do so.

The Documentation Challenge

While there have been many attempts at automating the development and maintenance of documentation it still remains a human process at the core. To satisfy the main responsibility for this need, technology vendors employ the services of professionals known as 'technical writers.' These people have a rare combination of technical knowledge and writing skills that, when properly applied, can produce documentation that serves the purpose of 'knowledge transfer' between subject matter experts, such as product developers, and product users. Many technology vendors see the act of employing technical writers as the answer to all 'knowledge transfer' problems. However, while employing the services of professional technical writers is definitely worthwhile, it is not the optimal situation when considering the volume of information needed in order to support a "whole-product" development strategy. To cover all documentation requirements a company would have to engage the services of many hundreds of technical writers, all focused on writing documentation at a level that is suitable for customer consumption. Obviously, this approach does not scale well as the costs would be prohibitive. For this reason, the responsibility of most technical writers is limited in scope to the development and maintenance of documentation pertaining to the "generic product" alone, as this is the most visible and sensitive component of documentation. Focus on the augmenting areas of the "whole-product" concept, are in most cases, either ignored or managed on an 'as needed' basis often known as 'putting out fires.'

In context of the "whole-product," the challenge of documentation development and maintenance is therefore not limited to satisfying the needs of the "generic product" alone. Instead the challenge is in implementing processes and systems by which the documentation needs of all areas can be satisfied while optimizing the limited resources of professional documentation developers such as technical writers. Before we can implement processes for "whole-product" documentation development, we must understand some understanding of the documentation development process. In particular we must identify problems with current processes.

Documentation Process

In reality the process of documentation development begins in the research phases of product design and continues through the development cycle, until the product is released to the market. During this time, various types of documentation are produced. Each documentation type serves a purpose to record knowledge and communicate the information to an audience group, either internal or external to the development effort. The majority of documentation is created for internal purposes, never being intended for use but external product users. As a result, this documentation is rarely shared across all departments involved in product development. Even the technical writers who are

responsible for producing documentation for the product users rarely have access to, or make use of documentation written in early development stages.

Already it is obvious that much of the "knowledge property" of an organization, captured in documents, is not leveraged in order to support the functions of a "whole-product." This is a core problem in documentation process. If we are to correctly implement the type of documentation process needed, we need to ensure that all investments in the capture "knowledge property" can be utilized, especially in context of "whole-product" information requirements.

The problem we are looking at is essentially a breakdown in communication. Analysis often reveals a number of easily rectifiable situations that, if properly addressed, can break the information impasse. Common situations include:

- A variety of tools being used for creating documents.
- A variety of file formats being used to store information.
- A variety of processes governing document creation.
- A variety of locations being used to save documents.

Variety and choice in a consumer market is healthy, but in a development environment where all processes need to be aligned for the benefit of efficiency, variety quickly becomes the cause of several obstacles. Left unchecked for a length of time, these obstacles become status quo within operations and an invisible retardant of communications. Brief reflection on each situation, quickly reveals the reasons.

The situation where a number of tools are used for creating documents is the root cause that snowballs into the situations that follow.

Documentation Tools

There are many tools available for authoring documents, most of them are easily obtained and often installed as a default part of office computing environments. Tools range from plain text editors to word processors and offer a variety of features to support the process of writing. Users quickly become proficient with the basic features of such tools and will therefore, without direction, start using the first tool with which they are familiar or comfortable. In most cases today, the word processor is the tool of choice as it gives authors a rich and powerful environment within which to format and decorate documents.

Initially this situation does not seem like a problem. After all, at least "knowledge property" is being captured. However, there are three problems:

1. Most word processing environments are specialized and user proficiency in using them is quickly acquired. When a document needs to be shared between authors, it is common to discover that maintaining consistency in the usage of the word processing features, especially those for formatting, is near impossible. Even the application of templates to govern styles is problematic since adherence to the styles and their usage requires discipline on the part of the authors.
2. Documents need to be portable and they are when everyone is using the same tool on a common operating system. However, portability is lost when authors in different parts of the organization are using different tools and different operating systems. A classic example of this situation is where developers are using word processors for authoring and technical writers are using specialized tools such as RoboHelp or Framemaker. Both are correct in selecting the right tool for the job, but incompatibilities between these tools makes portability between them less than straight forward, often impossible. In order to save time and support portability, it would be necessary for all authors to be trained in the usage of all authoring tools used within the organization.
3. The third problem is related more to the misalignment of such authoring tools with the final objective of making "knowledge property" accessible to all functions supporting the "whole

product" and provides a nice precursor to our next section on the section called "Documentation Formats". The problem, regardless of tool, is that most tools do not separate the concerns of information from presentation. This creates a situation where file formats and their presentation layers are not easily converted to the format or structural display needs of the user agents that will be used for access during support of the "whole product". In order to convert information complete duplication of content is required within a separate and disparate authoring environment. While this is not a problem on the first release of a product, it becomes resource and time intensive in the maintenance cycles that follow with subsequent releases because automatic conversion is not a seamless process.

Documentation Formats

The native file formats produced by most authoring environments are normally not easily shared between tools without conversion to an interchange format supported by both tools. Formats such as Rich Text Format (RTF), are therefore used as a means to display and edit files produced by one tool in another. This is often a cause of data loss and formatting control problems that can eventually lead to file corruption.

The obvious solution to the problem of file format compatibility is to ensure that all authors are using the same tool for authoring. This will automatically reduce training requirements, but does not guarantee consistent use of features in areas such as formatting. It also does not provide an answer to the problem of separating concerns so that information can be made easily available in various presentation formats.

Standardization on formats such as the OASIS Open Document Format (ODF) is a step in the right direction, but requires that all authoring environments will support ODF on all platforms. Even then, while ODF is an XML-based format, it is still a presentation format that does not address the problems of conversion to other presentation formats and structures.

For true cross-platform portability to exist and provide a means to easily convert to presentation formats and structures we need to use a source file format that separates the concerns of data and presentation entirely and which is easily editable across operating system platforms.

Documentation Processes

The method by which a documentation process starts is also a significant challenge for most organizations. Often document creation happens without first defining the need for a specific document or without checking whether or not such a document already exists within an organization. Obviously this duplicates efforts and results in multiple sources of information with varying degrees of completeness and accuracy. This exasperates the maintenance process and increases the time required to identify the right information in any given situation.

This problem is easily overcome by implementing a framework via which authors can locate and identify existing documents. Doing so firstly helps authors define the need for creating a new document and secondly helps them identify the relationship a new document will have in context of the currently available documentation set. Having said this, changing the process by which documents are commissioned is not the only intrusion needed. We must also evaluate the execution process and align it to the process of developing the "whole-product".

Traditionally a single documentation process is viewed as having an owner. This person is the document author and generally the only person who has access the document until such time as it is ready for review or is released for distribution. Much of the reason for this situation stems from the nature of traditional authoring tools that only allow single instance access when a file is opened. Extended exposure to this method is one of the main factors that led to the perception that document development is a solitary task, in which collaboration happens only at predefined draft stages.

This process model is a serialized sequence with points that enable collaboration. In catering for "whole-product" strategy this process is found to be slow and inefficient, often leading to bottlenecks in development and unacceptable lag times for information to be accessible where it is

needed. The long period between collaboration points also induces unnecessary risk to the document development process. An author may work several days, weeks or months on a document before a mistake in direction is noted. While such mistakes are rare with experienced authors, they do happen and can be costly. More often than not, changes in direction that cause problem are small but many. Many times developments in the product are not immediately reflected in the current version against which authors are documenting. For example, a small change in a user interface will take extended time to notice and align in the document.

To address this process we must first think of documentation development as a collaborative and highly iterative process, not a solitary task as we have already described. Instead of thinking in terms of the document owner or author, we need to think in terms of document contributors. Once we accept this change to process and our roles, we can accept that development on a document can be a concurrent process involving multiple contributors working in the same space and time. With ownership of a document shared by contributors, quality control becomes the responsibility of each person within their particular perspective.

Within this process, we can adjust our attention to developing sections of a document and releasing them early so that interested parties can participate. Most people involved in the development or review of a document find it easier to check small sections as apposed to reviewing whole chapters or complete manuals. While the change we are recommending is deeply intrusive, by changing it in this way we afford contributors the opportunity take part in document development early and to distribute their participation over time in digestible information loads and manageable periods of time.

Documentation Storage

If we are to change the documentation process as suggested the previous section, we need to provide infrastructure that will facilitate open collaboration that happens in as near real-time an environment as possible. Furthermore we must provide the possibility to exchange contribute, review and accept changes in a manner that is not demanding of time.

The obvious conclusion is to store files in a single location, where everyone can access and edit them. While this is a step in the right direction, it cannot be a complete solution until we enable simultaneous contribution in space and time with controls to ensure that contributors do not overwrite each others work. Furthermore, control must include capabilities to manage revisions and accountability for changes between revisions.

Additionally, in geographically distributed development environments, we must enable the infrastructure to be secure, yet globally accessible. The documentation storage facility must therefore function as a virtual hub around which contributors can be loosely connected as and when required, across time zones.

Standardized Structuring

There are many challenges in implementing a documentation process that support "whole-product" development. These challenges cannot be met solely through the implementation of software and technologies. There are issues related to change management that must be considered. Any change to the process or technologies used, needs to be thought through, preferably documented. Development of a plan is important and needs to be clearly communicated to all stakeholders.

Since we are advocating that documentation development should be engineered in a manner that encompasses a broad spectrum of contributors, buy in from people within the various departments is a must if we are to realize the full benefit. Often, especially in more established organizations, it is not always possible to make wide sweeping changes. In such cases it is often best not to pursue the topic too passionately, not every department may want to be as progressive or cutting-edge as you. Even though you are convinced it is in the interests of everyone.

Resistance to the change is normal and can be beneficial, especially in large organizations, as it gives you more time in which to implement across the organization. Having said this, there still needs to

be a majority buy-in concerning the general direction of the strategy. Even if all department of your organization immediately accept the proposal, we warn from taking a "Big-Bang" approach, unless you have the capacity to facilitate large transformation projects. In order to reduce risks and be capable of measuring results earlier, we recommend starting small. One of the best places to start with implementation is in the technical publications department. There are a few reasons for this:

People in technical publications are the most interested in information management and process that will improve documentation development techniques.

People in technical publications encounter the pitfalls of poor systems and processes on a daily basis.

People in technical publications often have a broader perspective of who is developing documentation throughout the organization. In many cases these sources are inputs to their local development process.

Another reason for this approach is to establish a key group of people who can be skilled to a point where they can provide support to future departments. The initial group is can also play an instrumental role in planning and designing the business process behind the system and will certainly help evaluate various tools and technologies. The experiences and results of the first implementation should be harnessed into a solution that will become the "standardized structure" for all documentation development processes throughout the organization.

Regardless of the specific situation the application of a standardize approach is normally a key benefit in providing a guidance framework that everyone can reference to both during implementation and into the future. When considering the development of a framework it is often beneficial to incorporate standards-based ideas and technologies where possible. Which standard and technologies to base on can vary depending on varying circumstances.

At InWords we have structured our solution around open source methodologies and technologies that demonstrate conformance to recognized standards. In many cases these technologies were already a close enough fit to our requirements for implementing documentation development processes that facilitate the "whole-product" strategy. Our decision to adopt open source methodologies and technologies and standards was motivated by the following reasons:

Many of our customers do not wish to acquire software from us; they want consultation and services to implement a solution. Since the cost of acquiring and using open source technology is free, as in 'free beer,' using open source technology significantly reduced, even eliminated, at least the barrier to entry associated with software licensing.

Our core competency is not in developing software, nor do we plan to build capacity to support any software development processes. By adopting open source technologies we inherit an expert developer and support community capable of supporting us at various levels. We also benefit from ongoing development and enhancement of the various technology components through their releases, without the overhead costs associated with development.

Since the technologies we have chosen are standards-based we can be relatively sure that they will work more or less the same for every instance we deploy. When in doubt, we can always just refer to the standard for clarity. Choosing open, standards-based technologies also gives us the benefit of not being locked in to the tool or technology of a single vendor. This point greatly increases the possibilities we have to leverage alternate tools without major upheaval at any point in time.

Open documentation is available for all the technologies we use. This not only helps us provide training to customers, it also help customers support themselves.

At InWords we were able to implement a solution using the following basic building blocks:

As the source for our documentation we selected Docbook XML, an OASIS standard and open source project with a very large developer and user support base.

For collaboration and revision control we selected Subversion. This is a powerful and yet easy to use revision control management system. Like Docbook, Subversion is well supported implementing a wide range of standard interfaces for access.

For overall project management we selected Trac. Trac is a minimalist approach to web-based management of software projects. Its goal is to simplify effective tracking and handling of software issues, enhancements and overall progress.

For email communications we selected to use the GNU Mailman list manager.

Using the combination of these building blocks InWords was able to implement open and transparent documentation development processes.

End of part 1

Collaborative authoring and content creation is not limited to product documentation. All content creation can be executed in a collaborative methodology to save time and money. Anyone in need of creating content of some type, especially for the web, can make extensive use these methodologies and technologies to reduce overall costs and reduce the time required to deliver content to their customers.

To learn more about this, stay tuned for the next 4 parts.

Next parts to follow:

Part 2 – Using Docbook for Content Generation

Part 3 – Implementation Challenges

Part 4 – Collaborative Work Process

Part 5 – Production Environment

Exclusive Offer for “123 Gift for Me”

FREE 1 Hour Consultation on either one of the following subjects of your choice:

1. **Content Generation** – Learn how to create a collaborative environment where many writers can work on the same content and have it published real-time, with no software cost, all free for you to use.
2. **Authoring Process** – Learn how you can create a collaborative authoring process with open source tools that doesn't cost you a cent, all for free.
3. **Websites** – Get a DEMO on how to create superior websites for FREE, with easy-to-use content management tools and easily maintainable by anyone (and the best part you don't need to be a programmer or even know HTML.) **** (This consultation is courtesy of our sister company TheWebDesignCompany.co.za) ****

This FREE 1 hour consultation is valid only for 30 people and only for registrations entered in before the 22 of December.

After the 22nd of December, or on confirmation of 30 consultations, which ever comes first, we will charge regular price on all consultations.

We will only consult people that we TRULY believe we can HELP solve their publishing problems and add VALUE to THEIR BUSINESS.

ONLY fully completed registrations will be considered for consultation. This is highly important, as we need to allocate the appropriate consultants with the correct solutions.

Therefore, please fill in the applications **FULLY** and as **ACCURATELY** as you possibly can.

After the first allocated registrations we will continue to give valuable information via our newsletter, so don't forget to subscribe to our website. Our newsletter will keep you informed of all the advanced publishing and writing solutions.

To receive your GIFT of one hour FREE CONSULTATION please send email to: 123GFM@inwords.co.za containing the following details:

1. Subject line: Gift of choice: 1 (Content Generation) 2 (Authoring Process) 3 (Website/CMS)
2. Full Name of contact person
3. Skype username [all consultations are done via skype (www.skype.com)], unless you prefer to contact us via landline. Please state your preference.
4. Industry business operates in (e.g. automotive, health, internet, software, publishing, etc)
5. What problem would you like to resolve in this one-hour consultation? Please provide description of problem and background, also some background about the person receiving the consultation.

✧ **We will confirm your FREE Consultation via email to arrange a date and time.** ✧