Release Management Strategy

Efficient versioning, branching and merging

in a DITA environment

Position Paper

November 2011

Let's Talk XML

Table of Content

The Context

In the good old days, it was a lot simpler…

Most of the time, the release of a product's documentation is timed with the release of the product itself. In times past, all manufactured products were physical objects like cars, medical devices and clothes dryers. These kinds of products required major infrastructures to build, and the production processes were very complex. Because of that reality, the creation of an updated version of the product was done at a slower pace (every 2 to 3 years or even longer) and this cycle was the same for the product documentation. Furthermore, the documentation was done only in paper format. The cost to reprint a manual was very high and, given that it was necessary to manage an inventory of manuals, errata pages were used to provide corrections or inform clients about new product features. Although there were aspects of the editing process that seemed easier in the print environment (there was only one format to update and one version of the documentation), it was the arrival of digital technologies that created nothing less than a revolution in the way technical documentation would be produced.

The market has new expectations

The major transformation in industry we are seeing now is that products and their many versions are being released much more often than in the past. New production technologies, the drive to innovation and a highly competitive global business environment result in companies bringing new customized products to the marketplace quickly, frequently and to a worldwide customer base. This trend is even more obvious in the software industry where products can be released very frequently because there are no physical product constraints and the incremental production costs for new versions are minimal. Some software companies even go as much as releasing a new version of their product weekly.

Paper no longer dominates as the main output because there are many media available to distribute product documentation. In fact, customers now expect to access documentation with their computer, laptop, tablet and smartphone.

Another reality for technical documentation teams is the pressure they now feel to update documentation in real time given that it is published electronically on the Web. Therefore, digital publishing pushes documentation teams to update previous versions of documents more often, which results in having them managing multiple versions of the documentation in parallel.

Strategies from the print era, like the use of errata for updates and corrections, are no longer feasible options. Customers are not willing to update paper documentation in a manual, time-consuming manner. They expect that documentation is updated electronically, on an ongoing basis, and that, given their time constraints, there will be less but more highly relevant documentation that is easy to consult. These market expectations are driving technical publication teams to implement highly customized publishing solutions that generate personalized documentation for each customer in the appropriate format, based on each individual profile and preferences.

Today people are busy and don't have time to read non-relevant information but they do have time to contribute to documentation. They want to be able to share and comment on information with the professional community they belong to. This new trend of user-generated content is accelerating the cycle for updating the documentation. In effect, product documentation is becoming more and more like a constantly updated knowledge base to which the user community contributes. In this new world, the definition of a "release" is becoming more and more fuzzy--but that could be the topic for another position paper!

These new realities put a lot of pressure on technical documentation teams, requiring them to redesign their work and production processes. This is especially the case when senior management is asking teams to produce more in less time and with less staff.

"Just in time" is the new way of doing things and is driving companies to release products and their documentation at a much faster pace than before.

The Challenge

With this business context and these market expectations, technical publications teams need to fundamentally change the way they produce and distribute documentation.
The only way to be able to keep up to and ahead of the pace of this constant change is to work simultaneously on projects in a flexible manner and to maximize the reuse of all available content. Technical publication teams need to be able to assign resources to a next version of a product's documentation while they continue to update current and previous versions of the same document. For example, if a company releases a new version of its product every month, the team needs to work on the release of September at the same time that it is finalizing the August release. The ability for various teams to work in parallel on a range of versions--and thereby leverage their collective work for maximum reuse of content--is at the heart of the successful production of documentation in this digital age. The strategy to meet this challenge is what we call release management.

This white paper cannot tackle every aspect of release management but it will focus on some of the most important and challenging issues:

- Versioning – creation of a new version without losing the previous one
- Branching – reopening an older version to do a modification
- Merging – applying a modification of an old version to a new version

The Solution

How do you manage to work in parallel?

Two approaches are used currently by technical publications teams: working in the main pool or working in a sub pool. What is a pool? A pool represents a place where maps, topics, and images are stored. It is like a folder or workspace where writers and reviewers create and update their content. The main pool is like a main folder, and a sub pool is like a sub folder. Usually, users create a sub pool when they create a specific version of a document. For example: from the main pool, a user can create the sub pool “v1.0” for Map A (Figure 1).

The difference between the two approaches: you create the sub pool at the beginning of the process of creating a new document or at the end.

Creating the sub pool at the beginning

In this approach, a method we think has some limitations, the main pool contains only documents that have a completed status. All the work of creating documents is done in sub pools. The first thing that is done when a new project is launched is to create a sub pool and copy all objects (maps, topics, images, …) that are needed. These objects are then worked on in the sub pool and, when the content is stable and complete, the sub pool objects are duplicated back into the main pool. All objects of the sub pool are kept intact to be able to return to any previous version for republication or to perform some minor corrections. The objects in the main pool are duplicated so that they can be reused in other projects. Some topics are probably reused in different documents and, therefore, they need to exist in the main pool.

The major problem with this approach is that it makes some objects inaccessible to others while they are being worked on in the sub pool. This is a serious constraint in terms of workflow and productivity.

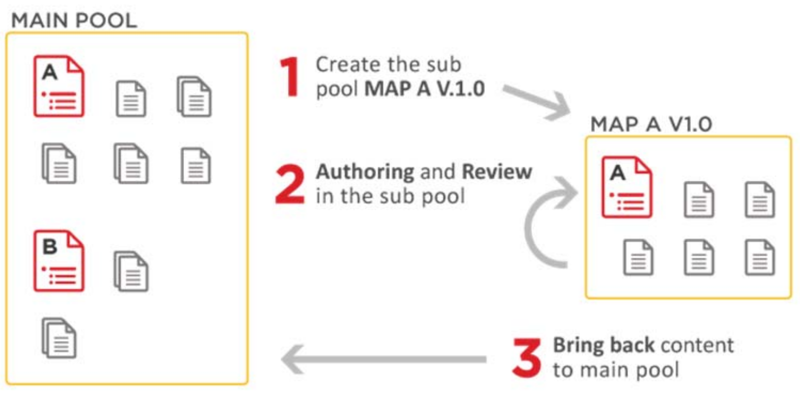
MAIN POOL
A
B
1 Create the sub pool MAP A V.1.0
MAP A V1.0
A
2 Authoring and Review in the sub pool
3 Bring back content to main pool

Figure 1. *Creating the sub pool at the beginning*

Creating the sub pool when documents are final

In this second approach illustrated in Figure 2, writers first work on the objects of the main pool and, when everything is completed and ready to be published, they create the sub pool where they duplicate the objects. In this case, a sub pool is created in order to be able to come back to this specific version in the future. This is like taking a snapshot in time of all objects of map A in order to have the opportunity to come back to this snapshot at any time in the future.

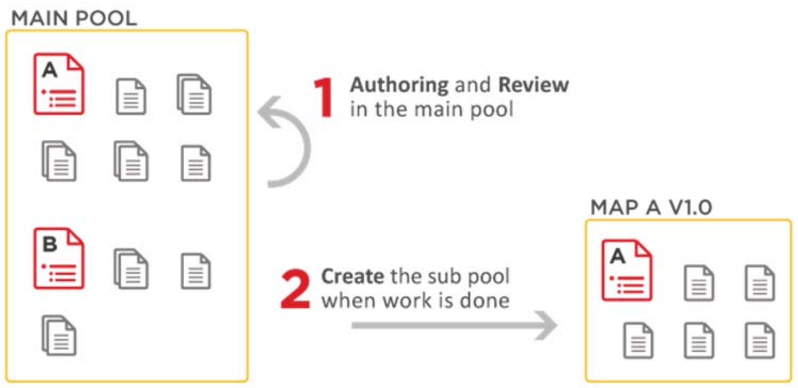
MAIN POOL
A
B
1 Authoring and Review in the main pool
MAP A V1.0
A
2 Create the sub pool when work is done

***Figure 2**. Creating the sub pool at the end*

Our preference is…

At IXIASOFT, we prefer the second approach because, with DITA, projects (i.e., maps) share many topics and other objects such as images. If there are multiple teams that are working on different projects at the same time, all of them will share the latest version of any object that is accessible in the main pool and will instantly benefit from the work of others, that is, at the very moment of the creation of the sub pool. This strategy maximizes content reuse.

Branching in the DITA CMS

In the IXIASOFT DITA CMS, objects (maps, topics, and images) are created in the main pool. When the authoring and reviewing of these objects is completed and approved, users have the option to move this content from the Authoring cycle to the Published cycle. In the DITA CMS, the "Publish" command is the equivalent to (or a synonym of) the "tag", "label", or "version" functionality.

As illustrated in Figure 3, users can right-click on a map, such as Map A in the example, and select the "Publish" command. The "Publish" command will prompt users for a label, which will become the name of the sub pool. Typically, users will enter something like "V1.0". The "Publish" command creates a sub pool "V1.0" for Map A and copies all objects referenced by Map A.

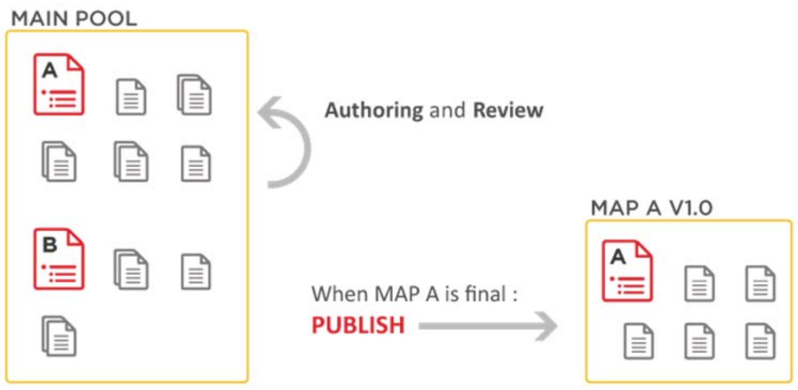

Figure 3*. Creation of the version V1.0 of the Map A*

Once a map is "Published", various options are offered such as "Localize"(if the content needs to be translated), "Generate output" (if the user wants to render the content in a specific format), "Deprecate" (if a map is no longer useful), or "Archive" (if the map needs to be removed from the live system).

When "V1.0" is published, users can start to work again with the objects from the main pool. When the next version is ready, they can publish "V2.0" and start to work from the main pool again for a future "V3.0" (Figure 4).

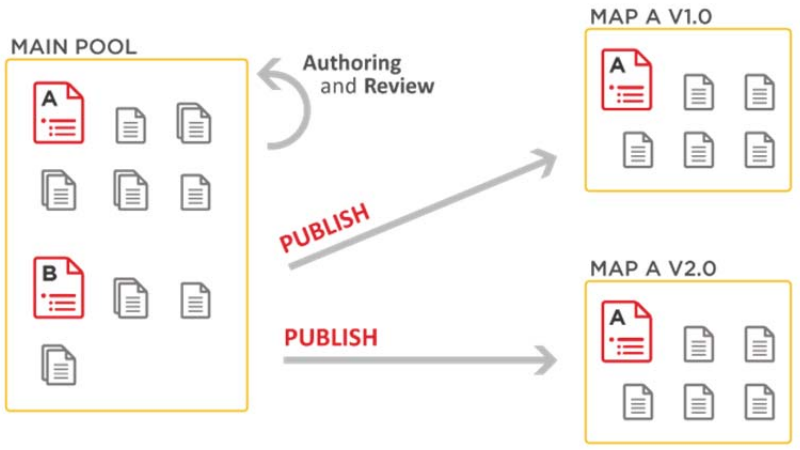

***Figure 4**. Creation of the version V2.0 of the Map A*

During this process, it is also possible that “V2.0” needs to be modified and updated to “V2.1”, as shown in Figure 5. Usually, the main pool objects cannot be used for this process because some of objects might have already been modified. For example, it is possible that a major new version (“V3.0”) has already been started and that many topics have been modified. As such, in order to create “v2.1”, authors would need to use the “v2.0” sub pool as a starting point.

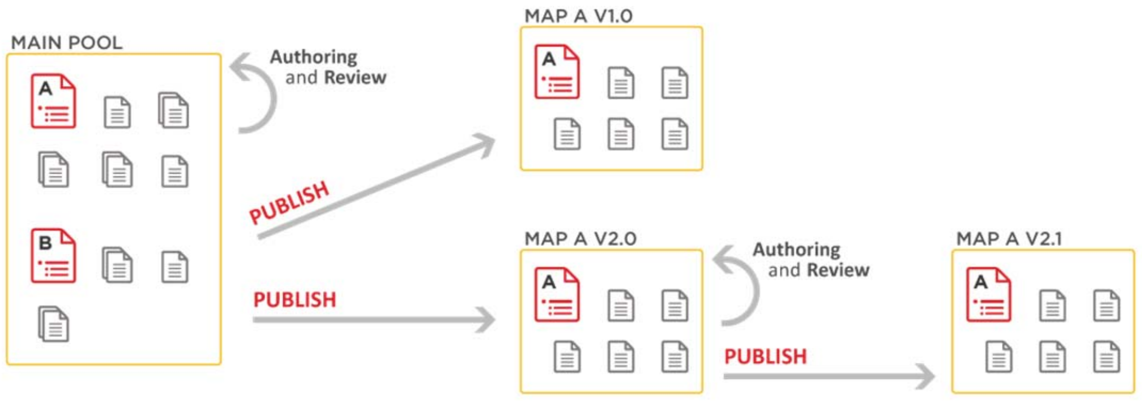

***Figure 5**. Creation of the version V2.1 of the Map A*

In this scenario, users can work on the main pool for a future “V3.0” and on “V2.1” at the same time without any problem.

In the IXIASOFT DITA CMS, there is no limit on the number of published versions, sub pools and sub-sub pools. The example provided here illustrates how various teams can work in parallel and maximize accessibility and reuse of content with only a few versions. But in the workplace, users are often required to manage a much larger number of versions associated with different releases of the product, and this is where the management of branches becomes a crucial function to optimize content reuse and consistency. The release management strategy described above provides the framework necessary to manage this complexity.

Branching is easy but merging can be tricky

What is a merge?

Assume that some modifications made in “V2.0” to produce “V2.1” could be interesting for a future version “V3.0”. In this case, these modifications may need to be applied on the main pool elements (topics, images, and so on). By using this strategy, the next version published from the main pool will benefit from the updated work done in “V2.0”. That action is referred to as a merge: that is when modifications executed in sub pool elements are applied on the main pool elements.

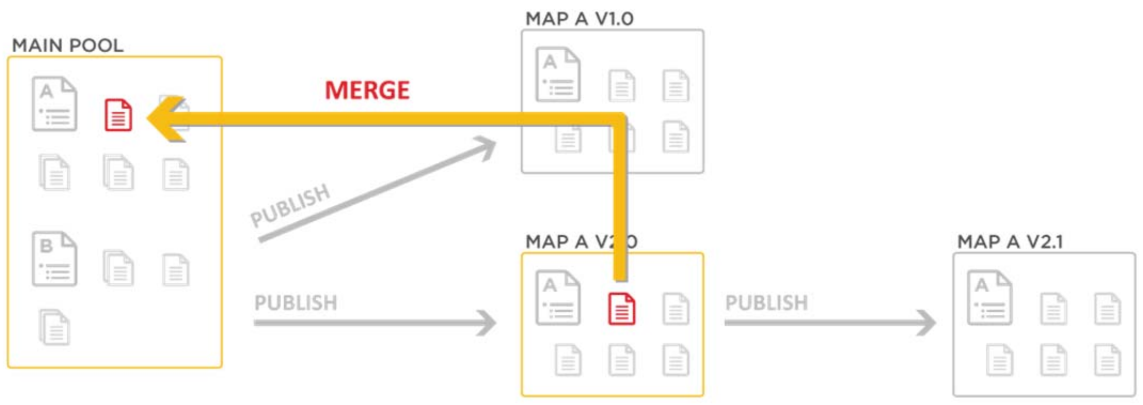

***Figure 6**. Merging a topic of the Version V2.0 with the main pool*

Merge scenarios

There are four possible scenarios for a merge:

- Scenario 1: The topic changed in “V2.0” was not modified in the main pool. In this case, the main pool topic is replaced by the “V2.0” topic.
- Scenario 2: The topic is no longer available in the main pool. Therefore, there is no reason to merge the topic with the main pool.
- Scenario 3: The topic is new and was created in the sub pool “V2.0”. In that case, we copy the topic back into the main pool. Actually, in the IXIASOFT DITA CMS, all new topics are automatically created in the main pool; users don’t have to merge them. The new topics are available for everyone as soon as they are created.
- Scenario 4: The topic in the main pool was modified. In that case, we need to merge the two topics manually because both topics contain new content. There is no tool currently on the market to do this kind of merging automatically. The only tool that may be useful is a comparison tool that will show you the differences between the two topics.

Merging in the DITA CMS

The IXIASOFT DITA CMS will assist users in the merge process: when a user displays all topics of "V2.0", the DITA CMS will automatically flag with an asterisk all topics that are candidates to be merged (e.g., all topics that were modified in the sub pool). Then, when a user selects a topic with an asterisk and decides to merge it, the system will evaluate if it can do so automatically (one of the first three scenarios). If the topic requires a manual merging, the DITA CMS will automatically launch the comparison tool between the two topics. The merged version will then replace the topic in the main pool.

The DITA CMS optimizes production documentation processes

One of the biggest advantages of DITA is that technical publications teams can make major gains in productivity and cost reductions by reusing content. This is especially true for translation because the cost of translation can be significantly reduced by reusing topics that have already been translated. The "branching" process duplicates elements by cloning. Cloning a topic is generally not a recommended best practice for the optimal reuse of content. To take advantage of branching without compromising the reusability of topics, the IXIASOFT DITA CMS minimizes duplication by keeping the links between topics "under the hood". In the DITA CMS, all objects are stamped with a revision number which is used by the system to make automated decisions in branching, merging, and localization processes, thus reducing the time users spend on the manipulation of content. By doing so, the system maximizes the reusability of content while saving significant time for the publications team.

Another example of process optimization is the creation of a new topic in a sub pool. In this case, in theory, the DITA CMS should create the topic in the sub pool and duplicate it when it merges it with the main pool. Because the DITA CMS creates the topic in the main pool, there is no need to merge it, thus allowing other team members to reuse the content as soon as it is created.

There are many optimizations like this one in the DITA CMS that allow users to reuse content, save time, and become much more efficient in managing releases of documentation.

The Bottom Line: Saving Time and Money through Smart Release Management

We live in a new digitally driven marketplace where time is unceasingly shrinking but where expectations are continuously increasing. Companies are releasing products much more quickly and frequently than in times past, and technical publication teams need to be on top of the action. The only way to produce technical documents at the lowest cost possible is to produce the content in a highly structured but flexible information sharing environment where projects can be worked on in parallel by a number of teams and all of the elements of content are accessible at any moment. To manage all these parallel tracks, technical publication teams need a strategy, an architecture, and a CMS that will help them realize major gains in efficiency, productivity, and quality of service.