Implementing the Big 3: A project management view on lessons learned on implementing an XML/DITA/CCMS publishing environment

Chona Shumate, Cymer, Inc.

My team successfully implemented a DITA-compliant, integrated component content management system (CCMS). As a result, we have realized significant productivity and quality gains through better efficiency and automation. This was a complex project involving several cross-functional partners. Although considered a highly successful project, it was not without problems. When we conducted an informal analysis after completing this implementation, a backward view came sharply into focus. Many lessons were learned, some subtle and some glaringly obvious in hind sight. Sharing this post-mortem view may provide you with an opportunity to avoid some of the pitfalls, or at least know to expect them and plan accordingly.

Note that this article does not focus on the technical aspects of the effort, but rather, the experiences learned from a project management point of view. For reference, I’ve provided examples of some of the deliverables I refer to in this article.

The XML/DITA/CMS Environment—It’s Not Just New Tools

Adopting a new document format, a new set of standards, and new database model for both development and delivery of content affects just about everything in a technical publications group. This kind of change permeates all and redefines it. It’s an overhaul of

♦ process
♦ tools
♦ skill sets
♦ mind sets
♦ scheduling
♦ workflows
♦ ownership

It can be difficult to know where or how to start a project like this. One approach, provided here, is to break down the project into sequenced chunks of activity: take the big, nebulous thing and break it apart into smaller pieces that are not so intimidating. I’ve sequenced some high-level activity in the following sections that reflect what we planned appropriately and what we didn’t and learned along the way. I also added observations to support the decisions we made, which may or may not be applicable in your environment. We are a small, semi-autonomous team but very dependent on cross-functional support.

The sections ahead are broken into the following sequence:

♦ Starting Out—Understand that everyone needs to know “why the change” and to accept that there will be problems. Given the usual caveats of the unexpected, it is important to set and keep to expectations where possible.

♦ Forming the Team—A strong project lead is key to a successful cross-functional team. Select team participants based on required skills and individuals’ motivation for success.

♦ Organization and Talent—Change can bring opportunities, but it also represents loss, the loss of familiar tools and known processes. Engage your entire team and make sure communication is strong and bi-directional.

♦ Selecting the Product—The team needs to establish requirements and do research. This is where you need to partner with IT to probe product offerings and evaluate the deeper technical layers. You need a strong understanding of DITA and the DITA Open Tool Kit (OT).

♦ Implementation—Define your success criteria before starting out. Know your budget, keep it within scope, and build a timeline. Plan the majority of your project schedule around developing a strategy for reorganizing your content for DITA structure and file conversion. Incorporate testing as a key process step and figure out what your success indicators are. Promote and plan training for everyone and that includes IT.

The following sections provide some detail on the lessons we learned on each of these activities.

Starting Out

Everyone Needs to Know “Why the Change”

Buy-in requires an understanding of the business case. In our situation, we were going to fail if we didn't change the way we did business. The approval for this project required everyone to know what would happen if nothing changed. My primary message to management and staff was “This is not optional.” I presented
compelling graphs of projected workloads over the next 5 years showing scarce capacity to support them. I also presented on issues of burn-out and retention and the importance of being a solid differentiator among our competitors. For my team, they needed to understand why they were being put through this drastic change and “why can’t we just do what we have been doing?” A critical first step is to tackle any resistance and get your team on board. You can anticipate that a change like this may not be accepted by everyone and some tough decisions may lie ahead.

**Know and Accept There Will Be Problems**
This is a constant for all projects, and it’s best to just accept this reality early in the going. Problems are inherent with change, and with change come the unknowns. The best defense is planning. Plan, plan, plan, and then plan some more. We adopted the “80% Rule,” which requires spending a majority of time in the planning phase. Eighty percent planning can avoid 80 percent rework. The instinct is to roll up the sleeves and jump in. Resist that urge, and dig deep for information, particularly for best practices and “what went wrong” experiences. Research the DITA standard and web sites on the OASIS web site. There are DITA user groups and web sites that have a wealth of information you can download. Take the time to benchmark and learn from the mistakes and successes of your peers in the industry.

**Setting and Keeping Expectations**
When beginning the project, it’s great to be excited about its potential when talking to staff or management. But, be careful. Don’t promise the moon or increased revenue. In fact, don’t promise anything. Undersell and over-deliver. Be honest with your management on what these new tools will do and not do. Emphasize what’s required to implement this change, and remind them that ROI will be over time. It won’t eliminate headcount (although it may minimize it). There will be steep learning curves and lots of “gotchas” along the way. The problems you have now didn’t happen overnight and the solutions won’t either. Also, remind all stakeholders that successful implementation is entirely dependent on cross-functional collaboration and support.

**Moving From Files on a Network To Records in a Database**
Many tech pubs’ groups work on files kept in a directory structure on the network. Changing to a database-driven architecture is a fundamental shift in how your team needs to think about their content. Files become data records in a database, with properties and relationships to other data. A CCMS can bring profound capabilities with the power of a database. Conversely, with automated features and functionality also comes the cost of complexity. Two examples of a mapped CCMS architecture are provided that we used to help our team better understand the change in our publishing environment. See <http://www.infomanagementcenter.com/ShumateFiles/IP_CMSprocessMAP.jpg> and <http://www.infomanagementcenter.com/ShumateFiles/TechnicalPublicationArchitecture_B.jpg>.

**Tip:** The role of content developer becomes narrower because formatting and publishing become almost irrelevant; however, understanding how to structure information and assign and leverage metadata is critical to efficient use of a DITA-compliant CCMS.

**Forming the Implementation Team**

**Form a Cross-functional Team**
As mentioned previously, success is cross-functionally dependent and a win/loss for all. Form your team with stakeholders who have skin in the CCMS game and are motivated for change. You also may need subject matter experts (SMEs) with whom you will partner throughout your project and who will need their time approved by their management.

Each core team member has needs from the team to be successful. Our core team consisted of members from Tech Pubs, IT, and the vendor.

- Tech Pubs must establish solid, comfortable, trusting relationship with all team members and be motivated for change. Besides myself, I assigned our Information Architect (IA) and a Senior Tech Writer to this team.
- IT must have a stake in implementing a successful project for the business side and understand the existing constraints with your as-is condition. A dedicated IT Business Analyst was assigned to this team.
- The vendor needs full access to the team and to IT and should be treated as part of the team. Communication should be routine with concise reporting on project status. Our vendor assigned the Sales Rep, a Project Manager, and the Lead Product Engineer to this team.

Management needs to be kept informed and relied on for funding and critical go/no go decisions.

**Tip:** Keep your team small and flexible, and make decisions at your level whenever possible.

**Select a Project Lead**
Your project needs a point person. This is a critical role that can influence the success of a project. My recommendation is to select someone who can:

- Learn new technology and not be intimidated. In our case, the project lead needed (1) a good understanding of enforced document structure, (2) a good understanding of the DITA OT Kit, and (3) willingness to learn how CCMS systems work. A technical background in database architecture and markup language is very helpful.
♦ Develop and manage project planning, and be thinking 10 steps ahead. An important part of this is reporting and knowing who needs to know what. Being able to provide possible solutions when reporting problems is crucial for retaining confidence with management during the project.

♦ Convey the bad news. Problems with implementation don’t go away without clear understanding of their impact and the scope of possible solutions. You need a project lead who isn’t afraid to be direct and up front with concerns, even if he or she doesn’t yet have a solution.

♦ Show leadership and respect, and build rapport with the cross-functional team.

Tip: This is an important role that directly affects the outcome of the project. Often managers step in to assume this work, but consider that this presents a great opportunity for a strong team member to demonstrate key skills and value to your organization.

**Organization and Talent**

**Change = Opportunities**

This kind of project raises the bar for talent and skills. You will get resistance from some, and to others, it will be an exciting new challenge in their career. You need to determine how roles and responsibilities will change and how you go about making those changes. Consider the following:

♦ Where are your team strengths and can you exploit them effectively?
♦ Are there gaps in on-board skill sets?
♦ What training will be required and is it available locally?
♦ What job descriptions will change and provide new opportunities for your team?
♦ Do you need to hire new/other staff?

Adopt a 360 degree perspective for your project to identify contingencies and process intersections: what goes into the system, what comes out, and who does what to make that happen. At a minimum you need an “outside looking in” expert and one “inside looking out” expert. In our case, our Staff Writer and Information Architect took on these roles.

**Learning Motivates Adoption**

Promote self-learning, research, and testing before and during the CCMS product search phase. For example, download and practice with free XML-related tools:

♦ Conversion applications for converting from the existing file format to XML (MiF2Go®)
♦ Free DITA compliant XML editors for learning XML and DITA (i.e. XMLmind®)

Install and become familiar with the DITA Spec and the DITA OT. These tools are not for the novice and will take considerable study. This task is often assigned to the Information Architect, as it was in our case. Our IA was the DITA OT expert on our team. If no XSLT coding experience exists within your team, send someone to basic training. (A week long training at AcademyX® was invaluable.)

**Don’t Forget the Rest of the Team**

Make sure your entire team knows exactly why you’re moving to a CCMS/DITA publishing environment. Be all-inclusive and constant with open communication and status. You should anticipate some “kicking and screaming” to get there, so solicit ideas and concerns, brainstorm solutions, and capture that input. Put action plans around the gaps and assign owners. Reinforce the mindset that you have a new paradigm. Convey reassurance to the team. No doubt, moving to DITA may be the biggest change they’ve ever experienced. Invest in training. Don’t nickel and dime on bringing the skill sets you need back into your team. Groups commonly underestimate the importance of training and the requirements for project funding.

**Over Communicate**

Set up weekly meetings and review the schedule constantly. All team members are accountable for contributing and problem solving, so plan for contingencies. Report upward on a weekly basis and keep management constantly informed—good or bad. Delays cause concern but are inevitable. Communication is key in setting management expectations and confidence. Document everything throughout the project. Six months later you won’t remember “why did we decide to do that?” or “didn’t we talk about that at one time?” Track with metrics and graph your progress. Communicate your progress with data-driven, visual information.

**Selecting the Product**

**Establish requirements and then do your research**

Establish core requirements collaboratively with the entire team. Requirements take a few iterative tries because they will eventually be your product spec. Your product requirements will drive product selection. Remember the 80 percent rule. We spent nine months of research and learned that products change and evolve. When talking to vendors, probe and ask lots of questions. Early on we created a matrix of must-have and nice-to-have features, and we used the matrix to conduct product evaluation by comparison, filling in checkmarks detailing which product provided what features.
Develop a Product Research Summary
Capture the features and evaluations for all products you looked at. Write brief reviews on the results of your comparative analysis, and include the criteria you used for ranking them. Summarize why you selected the product you did, identifying what was a priority for product selection. In our matrix of features, we weighted each feature from 1 to 3 by its importance. Weighting gave us a quantified analysis and made selection data-driven. An example file is provided that we used for defining CCMS requirements and vendor selection <http://www.infomanagementcenter.com/ShumateFiles/ShortListComparison8-9-2007.xls>. Also provided is a sample file containing the results of that vendor comparison <http://www.infomanagementcenter.com/ShumateFiles/VendorEvaluation.pdf>.

Get to Know the Social Side of Vendors
It is critical that you get to know the vendor and their culture; rate your relationship among your highest criteria of importance. The product you purchase is a package deal. Nothing, (I’ll say it again for emphasis) nothing makes up for poor customer service. Take note of the following: Are the vendors eager to come out repeatedly for meetings and demos? Do they answer your phone calls with a sense of urgency? Are their emails professional but informative? Are they direct when they have to answer “No”?

Schedule and Plan for Several Product Demos
We learned new questions to ask after every demo. The vendor we selected ended up coming on-site three times to give demos. Encourage your upper management and internal customers to attend so they can indirectly learn about your current constraints and the possible technical solutions. Make sure the product and vendor you choose is aligned with your vision. Don’t forsake your roadmap of excellence for a product that has nifty features.

Understand Options for an Integrated System
We implemented a fully-integrated system, which is common for a DITA-compliant CCMS. By integrated, I mean that we installed a system that comprised the database, development, and production servers, CCMS application(s) and utilities, an authoring tool, the DITA OT, and an output engine. All of these components operate via a single interface from the developer’s desktop. In our case, each developer’s computer has client–side applications installed. This creates some overhead with IT when patches, upgrades, and configuration settings need to be maintained on each computer. Alternatively, some vendors offer hosted or cloud-based systems, which are web-enabled and all system applications are set up and maintained remotely. Look into your options to determine what makes sense for your environment.

And, oh yeah, IT . . .

IT is Your New Best Friend
Build a partnership with IT early on. Include them in your core requirements definition. They should have a lot of input on your enterprise compatibility, scalability, database architecture, technical services, and so on. IT groups may not always understand the business case or sense of urgency, but they know how to look under the covers at system applications. We learned a lot about how our CCMS was actually a series of bundled apps and utilities. Although the product was robust, this bundle was of great concern to our IT group, which later proved accurate. A recent upgrade we went through was complex and difficult due to the way the system was architected.

We used a second-tier project plan solely for IT. Experience told us that we needed to be crystal clear on our expectations for IT resources and their time for planning, product reviews, contract review, installation, configuring, testing, problem solving, and the guarantee that we had the adequate hardware necessary and ongoing IT support for our CCMS. This plan was reviewed and signed by IT, our executive sponsor, and our management.

Work with IT to allow for all contingencies, and then plan for them. We hit a snag with IT vacation time we were unaware of, and the project languished for a couple of weeks right before the Go Live phase. Develop your schedule collaboratively with IT and insist on an IT backup person. Ask that IT be included in training on the product. Our IT team member did not attend the user training and had no familiarity with the interface during the implementation. This lack proved to be a handicap in the testing phase and later in troubleshooting configuration settings.

Tip: I learned to be generous with praise and lots of kudos. IT are like the police. They are often involved only when things go wrong. You have the ability to build a positive relationship on both sides.

IMPLEMENTATION
Define Success Criteria
Understand clearly what the objectives are that you want from this implementation. What are the success factors and how do you quantify them. Understand how you will know whether your project was successful—for you and your team, for IT, for the company, and for your customers. Success should be data-driven.

Know Your Budget
First and foremost when beginning the project, know exactly what was budgeted. Carefully examine the vendor quote with your business analyst. Be familiar and know the source of every line item in the contract/project plan they provide. Sort out
the true product costs early in the discussions. Ensure vendor technical support is itemized with roles and responsibilities and clearly identifies the process for problem tickets and escalation. Don't forget to budget for training, and make sure training is included in the quote. Keep to your budget and watch for scope creep. Spend everything per plan. If it's budgeted, spend it if it adds value. Report budget tracking in weekly updates.

Convert the Content
This activity will most likely encompass the lion share of work and complexity. The process of file conversion to XML should consume a large part of your planning. The strategy for converting data depends entirely on how your content is currently organized and structured. The operative word here is strategy because there are different methods and approaches to converting your files to XML-source information. This topic deserves lengthy discussion, and many articles have been written on best practices for converting content to DITA-compliant XML. For this article, I summarize a few key steps that may help you with this planning. The following is highly simplified and requires a good understanding of DITA.

♦ Decide if your output will be book-based or topic-based. This choice influences how you organize your content. Book-based content historically is organized around the type of activity (installation, operations, service and maintenance, site prep, and so on). Topic-based content focuses information around a single subject (ventilation module, chamber system, software variables). Our content was already topic-based and highly structured, so we were ahead of the game. I use the term “topic” to refer to a subject, but in the DITA environment, these are known as “maps.” Maps are containers for collections of related information.

♦ Decide how your content is to be categorized. For example, will it be categorized by customer, product, subsystem, region, software version, user levels, and so on? Your categorization will determine how you set up your metadata so you can single-source and repurpose content based on which properties you select for output.

♦ Tease apart all content and redistribute it into types of information. A DITA-compliant structure provides three information types: concept (overviews, descriptions), task (procedures), and reference (graphics, tables). Information should result in stand-alone pieces of content.

♦ Search for and eliminate all redundant content. You only need one instance of it to re-use for multiple outputs. If files contain very similar information, keep only one version, then apply your metadata properties to assign differences and allow multiple outputs from the same content.

♦ Determine your file naming convention. It needs to be scalable to accommodate future products, expansion of features, new customers, and so on. If you have 642 files to convert, plan that over time you will eventually have n times that many. Don't forget about your graphics files. They must be included in this naming convention, as well.

♦ Determine a method for migrating/Converting your content to XML. There are outside services which will do this, but not cheaply. You'll need to develop a specification that clearly explains what the resulting deliverable must be. I would suggest selecting a small subset of content to use as your pilot. We picked one platform and discovered a utility called MiF2Go which converted our FrameMaker files directly into XML. We mapped our FM tags to the associated XML tags, and MiF2Go was able to convert files by batch mode—and in minutes!

♦ Plan time for post-conversion cleanup. Even though our conversion was fairly painless, we noticed some inconsistencies in formatting, tables, margins, and symbols.

You and your team will need a full understanding of DITA and how the DITA OT is implemented. I recommend some upfront research, and then taking a DITA writing course. An excellent one is the DITA Boot Camp, offered by Comtech Services <http://www.comtech-serv.com/workshops/ditabootcamp.shtml>.

Love the Phased Approach and “KISS the Pilot”
The KISS principle has been around for a long time and for good reason. It stands for “keep it simple, stupid” and can be a good rule of thumb for planning your pilot implementation before the big “Go Live.” Again, be vigilant for scope creep and stick to plan. Work with the stakeholders to develop criteria for what is in, what is out, and what gets bucketed. What’s not KISSable, bucket into future development phases if it adds value. New ideas will emerge, and when they do, have an agreement that all new ideas will be bounced off the original criteria. Stay on track with the original plan as much as possible, and be willing to make tough, unpopular calls, especially if not all aspects are known.

Develop a Timeline
Build a padded timeline, then double the time for every task. Yes, double it. Update it weekly and publish to IT and Management, and in your weekly status report. Plan for contingencies, and factor in holidays, sick time, and vacation schedules. Pace your work and expectations. Remember, implementing a CCMS environment changes everything and challenges every assumption on how you did your job previously. Identify the constraints and anticipate they will cause delays. Use your milestones’ status as one of your metrics. Map out and put a time table to the bucketed items for the next phase(s). A sample project plan is provided at <http://www.infomanagementcenter.com/ShumateFiles/CCMSImplementationPlanning.pdf>.

Tip: Plan for scale; implement by phase.

Once the project is approved, you and your team are in for the long haul. Most projects like this can take 18 to 24 months to complete. The finish line is the point where all users are trained, the change-over to the new system is complete, success criteria are met, and the old system is obsolete. Working in parallel with old and new is not a finished project.
Don’t Forget the Training!
Training was a big gap in our implementation. What the vendor thought was training did not meet our expectations and was insufficient. It consisted of two days of overviews taught by a project programmer. Several of us came away from training not fully understanding product functionality, which became a problem during the testing phase. Have the vendor provide an agenda for the training course. Insist on training by a trainer (not an engineer), and make sure you’re provided training materials. Determine where training will occur—onsite or at the vendor’s site. If onsite, schedule the training space, networked computers, and other equipment you’ll need ahead of time.

Build Use Cases
Use cases are helpful during testing to test features and functionality of the CCMS. Our IT department provided us with a template to capture all the tasks we needed to be demonstrated by the CCMS and their expected behavior. Because our system is integrated with an authoring tool and output engine, we needed to test all components and their automated handoffs to other parts of the system. This step is critical. Spend the time developing all the use cases your team can think of for every step in the development and publishing process. An example of uses cases we developed is provided for reference at <http://www.infomanagementcenter.com/ShumateFiles/UseCasesProcedures.pdf>.

Map Out Your Workflows
Our CCMS has a graphical workflow for projects. It is a great project tool for quickly identifying where content is in its development cycle. Because workflows are somewhat different for a tech pubs organization, they must be designed for your process. We did several white board sessions mapping out the high level steps that the project proceeds through until publication. In our case, developing this collaboratively with IT helped them learn more about our content development process. I’ll warn you that workflow development looks deceivingly simple and straightforward. Be sure to test the assumptions of your workflow; for instance, what would happen if you had a one-off project if you had to reverse projects back to a previous step in the workflow, or if you put a project on hold. Steps in your workflow may require “triggers” to push it to the next step in the cycle. Workshop steps get overly complicated if too detailed.

Tip: The simpler the workflow, the more easily you can adapt it to the unique events that happen during development. We ended up with two simple workflows: one for Topic Edits (9 steps) and one for Map Edits (7 steps).

Testing is Your Validation
Apply your use cases to test everything—every step in every task, no matter how simple it seems. For your timeline, double the time you would estimate for each test. Test remotely and from different computers.

Our project lead developed what he called a “smoke test” to check for expected screen displays, navigation, and menu options. When unexpected functional behavior occurred, he took screen shots and added notes to describe what was wrong. This process helped IT isolate a problem quickly. You also need to verify that your workflows accurate represent where projects are positioned in the development cycle. If you have triggers, such as an email notification or an automated push to the next step, confirm they work as architectured.

When the project is going well, it is very tempting to think you can whiz through the testing or shorten it. Testing is one of the most critical steps in your project, showing where the unexpected kinks are. We had plenty of gotchas after we went live. We could have done a better job with more thorough testing.

Brainstorm and then document for all contingencies. What do you do if this test fails? What are the risks if this test fails? Know that at any time you could be confronted with a showstopper. Be prepared with fall-back ideas but be careful of “work-arounds” that seem viable at first, but may not be a robust solution over time. Avoid being forced to ask yourself “Now what?”

Understand the Process Side
Maintaining two publishing processes until all data is migrated, tested, and proven in the field may be required. This maintenance can be hugely underestimated and risk confusion with priorities and frustration if not planned properly. Select a small amount of data for the pilot. We chose a smaller, older product line that had minimal ongoing changes. Make your internal customers aware of your constraints during this project period. Your team is essentially doing their day jobs simultaneously and in parallel with system implementation.

Metrics Define Value
Decide what measurements are meaningful to you and your team. Where are the pain points, time sinks, the chronic problems that, with your current environment, prevent you from optimizing your processes and team talents. Include metrics that are important to your management and the company. Know that these can be different, especially if one group looks for product-centric data and you’re looking at process-centric metrics. Some of these metrics can translate into success criteria for your project.
Tip: It is very important before starting the project to take Before metrics to compare with the After metrics. Comparison metrics after implementation are guesswork and can be inaccurate.

Plan for collecting data. Some metrics to think about are:

♦ Maintaining a timeline and hitting milestones on target
♦ Staying (or not) within budget
♦ Meeting a first output production date from project start (workflows)
♦ Measuring Cycle times—before and after implementation (ROI)
♦ Time through publication cycle
♦ Time to market (your deliverables in support of company product launch)
♦ Meeting output volume over a specified time—before and after
♦ Reworking content due to manual re-touches before and after automation
♦ Measuring workload—changes in size and volume of assignments per developer before and after

Implement the Product as Recommended
Stick with vendor hardware recommendations. Don’t be a guinea pig for new IT systems or be “the perfect opportunity” to try out a new process. Our IT department decided to use our project as the test bed for their new virtual server environment. This decision caused some delay because it was new, in-house IT engineers were not experts, and they had never tested VM against our new CCMS database.

Ensure the vendor relationship and communication paths are solid and purchase the “platinum” version of CCMS software maintenance. You’ll need it for at least the first two years. Request that an IT representative be assigned as an ongoing resource after the CCMS goes into production.

Most IT organizations have a tracking system for application issues. Follow their process and put in a ticket for every bug, no matter how small. What may seem a simple work around at first can be a costly headache later.

Finally, the Post Mortem
Each new project results in a lot of new awareness. Learning what worked and what didn’t is part of a project’s definition. Personalities emerge, technical difficulties occur, and typically, tension builds during a project. There’s no room for regrets, but lots of opportunity for new knowledge. The best way to not repeat the negative results is to capture all aspects of how the project unfolded. A “post-mortem” is a really bad term, because it should be a keen rear view that has high value for all stakeholders. You can’t foresee all problems, but you can dissect and analyze them afterwards. Hold a post-mortem jointly with the entire cross-functional team after your Go Live phase. Get challenges documented for future avoidance/lessons learned and for planning a Phase 2 or upgrade. Ask everyone on the team to contribute comments. Remind them that problems are not about blame, but about solutions and preventing a repeat of poor decision-making.

Summary
Implementing “The Big 3” can be intimidating, and result in disaster if you don’t plan. A good approach is to break up the project into manageable, well-planned chunks of activity. Start your project 18 to 24 months before you want to go live. Be all-inclusive with your project and keep the entire team informed of everything but agree on a standardized and efficient communication method. Anticipate problems, plan for contingencies, place priority on training, provide constant reassurance and praise, and have a big party when you go live. Reward your team with recognition, and let the important people in your company know what you accomplished.