



FLATIRONS | SOLUTIONS

Dynamic Content Delivery Using DITA

Eric Severson, Chief Technology Officer

Table of Contents

Introduction	3
Static vs. Dynamic Publishing	4
DITA: Dynamic Assembly of Topics	6
Reusing Topics Even When There are Variations	7
Using DITA to Simulate Dynamic Content Delivery	9
A Scalable Approach to Dynamic Content Delivery	11
Our Dynamic Content Delivery Solution.....	14
A Complete DITA Publishing Solution.....	17
Conclusion: Bottom Line Benefits	18

Introduction

Have you ever waded through a massive technical manual, desperately searching for the section that actually applied to you?

Have you found yourself performing one search after another, collecting one-by-one the pieces of the answer you need from a mass of documents and web pages?

These are all examples of the limitations of static publishing; that is, the limitations of publishing to a wide audience when people's needs and wants are not all the same.

In the days when print publishing was our only option, and we thought in terms of producing books, we really had no choice but to mass-distribute information and hope it met most people's needs. But today, with Web-based technology and new XML standards like DITA, we have other choices.

DITA is the hottest thing to have hit the technical publishing world in a long time. With its topic-based approach to authoring, DITA frees us from the need to think in terms of "books", and lets us focus on the underlying information. With DITA's modular, reusable information elements, we can not only publish across different formats and media – but also flexibly recombine information in almost any way we like.

Initial DITA implementations have focused primarily on publishing to pre-defined PDF, HTML and Help formats – that is, on static publishing. But the real promise of DITA lies in supporting dynamic, personalized content delivery.

This whitepaper defines a new publishing paradigm, which we will call dynamic content delivery. Dynamic delivery changes the rules, putting the reader in charge of what content is important and how it should be packaged. It transforms publishing to an audience of many to publishing to an audience of one.

Static vs. Dynamic Publishing

Figure 1 shows the classic static publishing model, in which a set of specific publications are created and produced to meet the needs of a wide target audience. Whether we are dealing with technical documentation, policies and procedures, courseware or marketing materials – and whether the publications are delivered in print or electronic form – this model involves central decisions on the appropriate content and packaging, followed by distribution en masse to the intended audience.

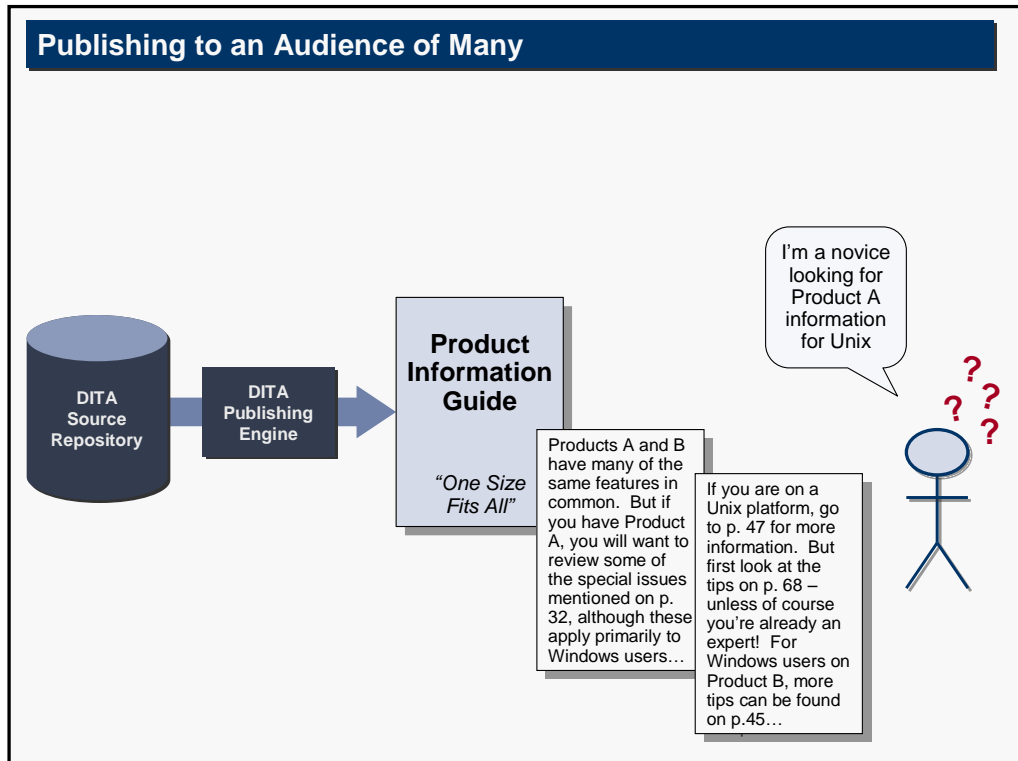


Figure 1. Classic Static Publishing Model

This model – which dates back to the invention of the printing press itself – has always had its limitations. In technical publishing, we often see examples of “one size fits all” manuals that cover so many options and variations that all readers are confused. Or, in an attempt to fix this problem, re-combining the same information across 20 or 30 manuals that are each optimized for a particular target audience – and still finding that you didn’t anticipate all the right target audiences.

The real problem is that all members of the target audience aren’t really the same. For novels and literature, that may not matter – but for technical material and professional publications, it can make a big difference. In this world, every reader has divergent needs and different ideas of what’s relevant. The classic “one size fits all” approach can end up fitting no one at all.

Figure 2 shows the alternative model of dynamic content delivery. This approach involves “pulling” rather than “pushing” content, based on the needs of each individual user.

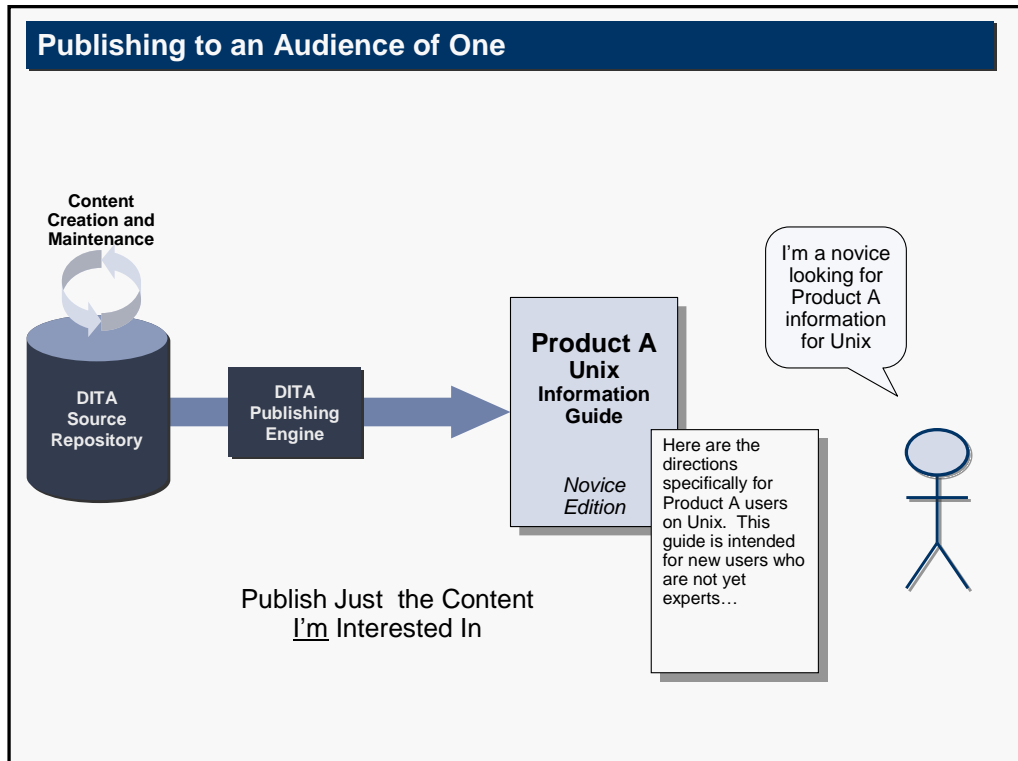


Figure 2. The Dynamic Content Delivery Model

In this paradigm, there is no central editorial decision that determines what content is appropriate for a particular user. Instead, the reader uses a dynamic search to choose the content that he or she considers relevant. The editorial process only determines the raw material – or pool of information – that’s appropriate for each subject area, from which the reader is allowed to choose.

In some applications, it may even be possible to automatically choose the appropriate content, based on a reader’s individual profile. For example, for a product installation manual we don’t need (or want) to ask the user what content they’re most interested in. If we know their user profile and the product they bought, we can automatically tailor the manual to fit their situation.

DITA: Dynamic Assembly of Topics

Dynamic content delivery is made possible by an increasingly popular technique known as topic-based authoring.

A topic is a piece of content that covers a specific subject, has an identifiable purpose, and can stand on its own (i.e., does not require a specific context in order to make sense). Topics don't start with "as stated above" or end with "as further described below," and they don't implicitly refer to other information that isn't contained within them. In a word, topics are fully reusable, in the sense that they can be used in any context where the information provided by the topic is needed.

DITA (Darwin Information Typing Architecture)¹ is a relatively new standard that was designed specifically for topic-based authoring. DITA doesn't focus on documents, but rather on independent, searchable topics that can be freely combined into documents, or assemblies, or collections, described by a DITA map. Figure 3 illustrates this approach.

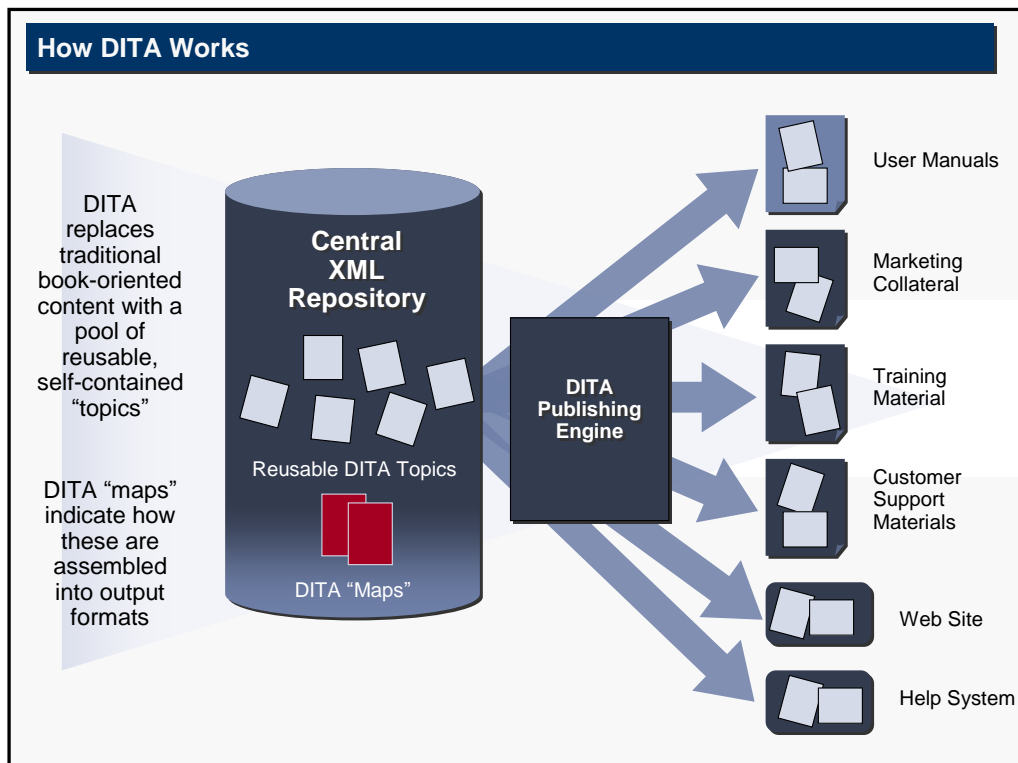


Figure 3. DITA: Assemblies of Reusable Topics

¹ More information on DITA, which was invented at IBM and is now an OASIS standard, can be found at <http://www.oasis-open.org>.

Reusing Topics Even When There are Variations

Although topic reuse is a very powerful concept, in practice it turns out that most topics contain some necessary variations that make them *almost* 100% reusable, but not quite.

Take a classic example from technical publishing, illustrated in Figure 4. Let's suppose that a certain server-based utility is used in many different situations and operating environments, and is therefore described redundantly across many different technical manuals. In a DITA scenario, this is a perfect opportunity to create a set of reusable topics that can be automatically incorporated into all the various places the utility must be described. For the introduction to the utility ("*About the Server Utility*"), this is straightforward – the same introductory text can be reused in all contexts. But for instructions to run the utility ("*Run the Server Utility*"), the situation is not so clear-cut. In this case, although most of the text is redundant, we really do need to insert different detailed instructions for each operating environment.

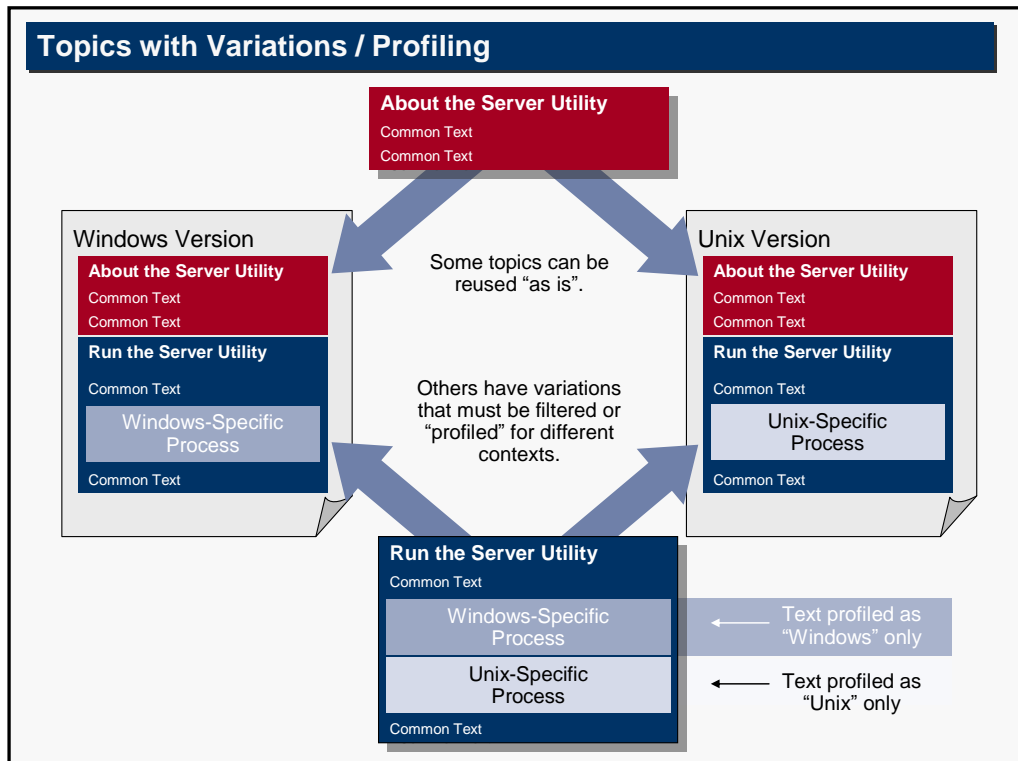


Figure 4. Profiling Conditional Text

This kind of variation could be handled by creating parallel topics – one for each operating environment – but a better way is to create one topic with conditional or *profiled* text. In this case, attributes within the XML text indicate the scenario or *profile* for which each variation is appropriate.

In the simple example shown in Figure 4, both a Windows and a Unix version of a document can be built from the same DITA map. In this case, the DITA *map* tells us to include exactly two DITA *topics* in the document:

- "About the Server Utility"
- "Run the Server Utility"

About the Server Utility is a reusable topic that is appropriate in either context. *Run the Server Utility* is also reusable in both contexts, but *profiling* must be set so that only material for the appropriate operating environment is included.

In Figure 5, we take a closer look to see how this really works. The profiling is accomplished by including parallel XML elements with different “platform” attributes. This information allows the system to filter out information applicable to other platforms. DITA allows for a variety of profiling attributes, including product, platform, audience, revision, and others of your choice.

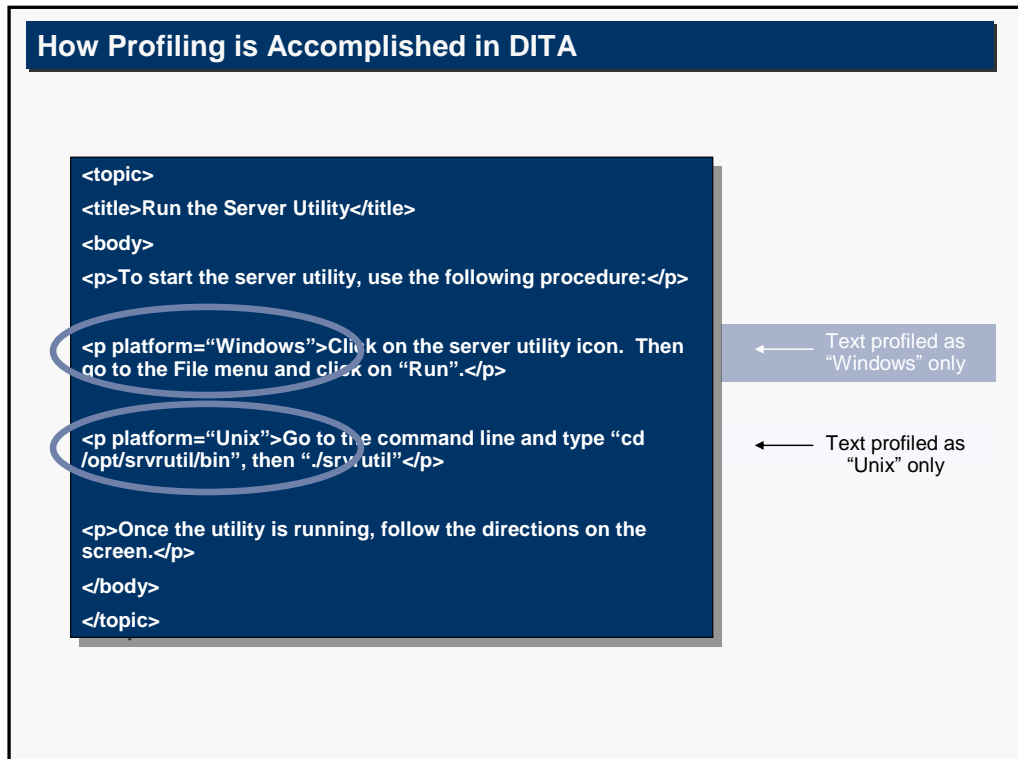


Figure 5. A Closer Look at Profiling

Using DITA to Simulate Dynamic Content Delivery

In a *static publishing* scenario, DITA maps are created in advance for each pre-defined publication type (documents, web pages, help systems, and so forth). Each of the maps contains references to the topics that should be included for that particular publication.

To avoid the “one size fits all” problem, DITA offers the flexibility to create a different map for each personalized scenario you wish to support. For example, rather than publish one book that covers multiple products, we could have a different DITA map for each. Each map would reference the topics appropriate for its product, and these topics could be profiled to select the material applicable to a specific platform (e.g., Windows vs. Unix).

In theory, extending this approach could get us closer and closer to *dynamic content delivery*. For example, if we needed expert and novice editions for each of the platforms, and one set for each of two products, we could construct eight DITA maps – one for each of these combinations. Or perhaps we could use fewer DITA maps, and use profiling to derive all the necessary combinations. Figure 6 illustrates this approach.

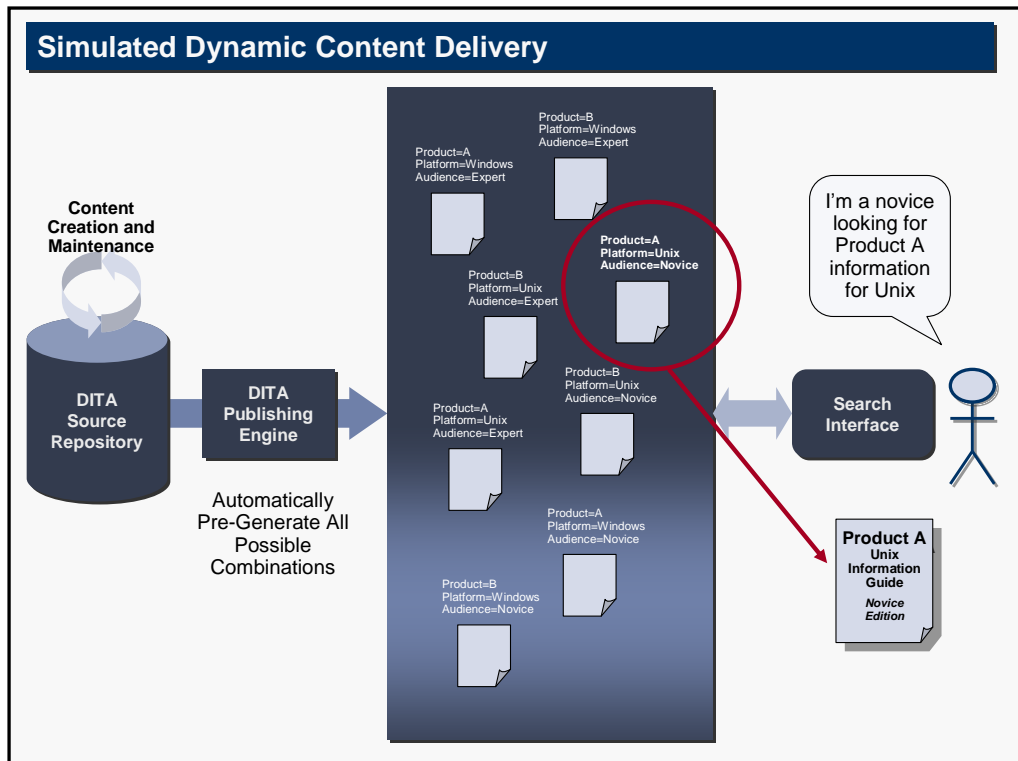


Figure 6. Using Static Techniques to Simulate Dynamic Content Delivery

To implement dynamic content delivery, all of the combinations can be pre-generated and stored with *metadata* that indicates the combination each represents. For example, the manual for *Product A, Unix, Novice Edition* would be stored with these three parameters as metadata. At content delivery time, the system could dynamically match this metadata with the user's profile (*Product=A; Platform=Unix; Audience=Novice*) and select the appropriate "personalized" edition.

The problem with this, of course, is that the number of combinations increases exponentially as the number of variations increases. In a real-life example, there may be not eight but more like 40,000 combinations. For example, suppose you have the following variables:

- *Product* (AA100, AB100, AA200, AB200, AC300, AD400, X5000, X6000)
- *Platform* (Windows, Mac, Solaris, AIX, HP-UX, Linux)
- *Audience* (Novice, Intermediate, Advanced, Expert)
- *Revision* (7.0, 7.1, 7.2, 7.3, 7.4, 7.5, 7.6, 7.7, 7.8, 7.9, 8.0, 8.1, 8.2, 8.3)
- *Geography* (North America, Latin America, EU, Asia, EMEA)
- *Interface* (Thin Client, Thick Client, Server)

Already, that's $8 \times 6 \times 4 \times 14 \times 5 \times 3 = 40,320$ combinations!

Storing that many variations of the manuals on disk may not be a problem – but maintaining all the DITA maps might. The real problem, though, is the complexity of the *pre-generation* process. To actually keep all the generated content up to date, all the affected outputs have to be re-generated every time a piece of source content changes! This approach is not scalable.

Real-World Example: Personalized Installation Manuals

A large manufacturer of computer storage hardware offers many product variations and a multitude of options. This makes the related technical documentation both complex to use and cumbersome to produce. With a Dynamic Publishing solution, users can enter their particular profiles, and receive dynamically-assembled technical manuals optimized just for them. This makes it feasible to sell products to a wider, less-sophisticated customer base, minimizes service calls, and lowers publication costs.

A Scalable Approach to Dynamic Content Delivery

Because of the scalability issues when outputs are pre-generated, we recommend dynamic generation of output *on-the-fly*. Because DITA encourages the development of reusable, standalone topics – modular information that can be flexibly recombined – it's the perfect standard for this purpose.

As shown in Figure 7, with this approach content is pushed to the delivery platform directly in XML. There, XQuery-based searches can find relevant topics, filter or “profile” topics so that only applicable content is included, and dynamically transform DITA XML into the desired output format (e.g., HTML and PDF).

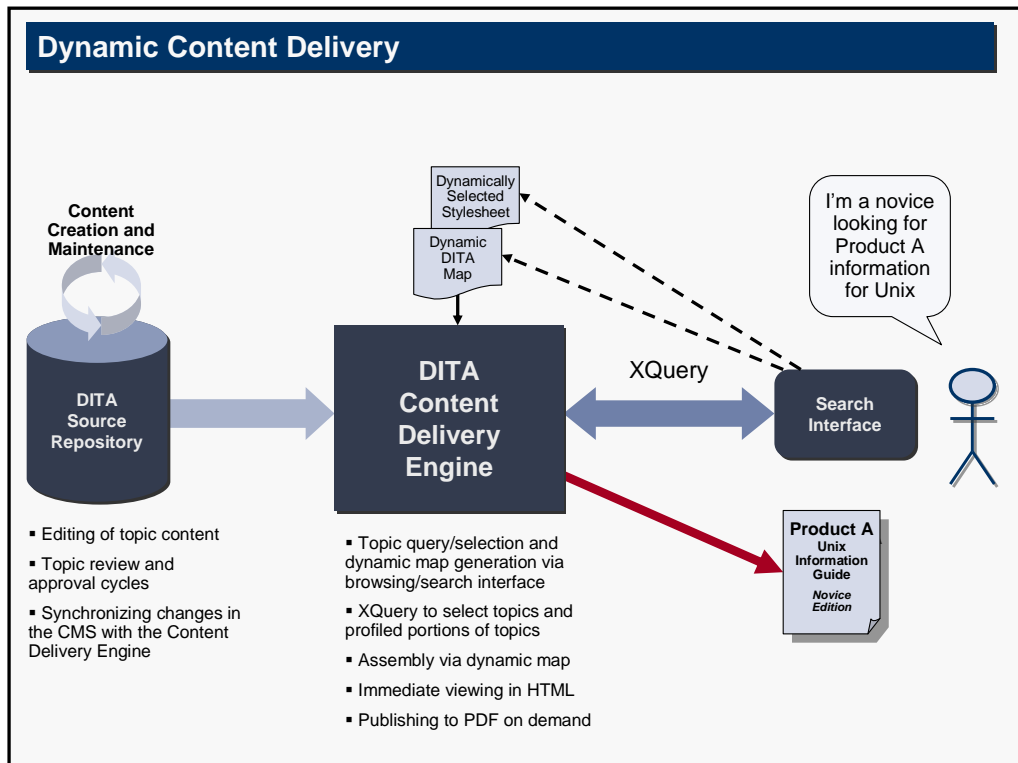


Figure 7. Scalable Dynamic Content Delivery Architecture

We chose XQuery² as the appropriate search technology, not only because it's the standard of choice for searching XML content, but also because it has the built-in ability to assemble and transform search results into alternative output formats. While doing so, it can also profile the results so that only applicable content is included.³

² More information on XQuery, which is now an official W3C Recommendation, can be found at <http://www.w3.org/TR/xquery/>.

³ DITA applicability indicators are expressed as XML attributes (for an example of this, see Figure 5). With its power to search inside XML elements and attributes, XQuery can use this information to determine which sections of content to include or exclude from the results.

DITA *maps*, which are pre-set in a static or pre-generated delivery environment, are also constructed dynamically. Such *dynamic maps* can be either created by the user or generated automatically by the system. Using XSL stylesheets – which can optionally be selected by the user – results can be previewed or published in HTML and PDF.

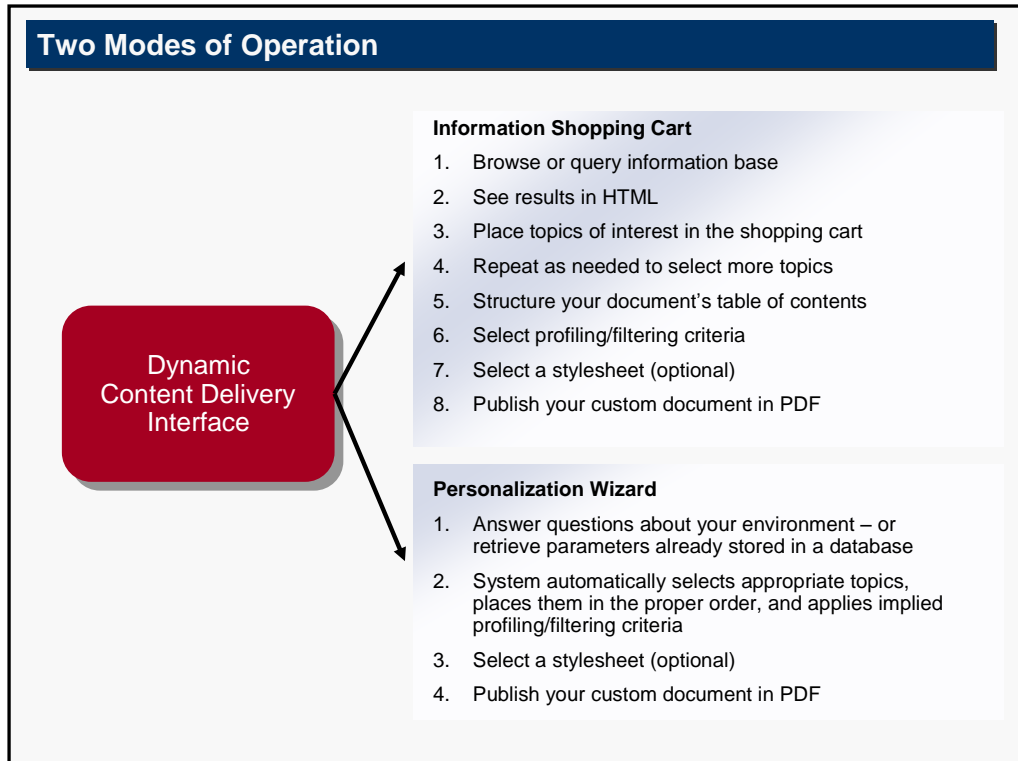


Figure 8. Example Dynamic Publishing Flow

As shown in Figure 8, two kinds of dynamic content delivery interfaces are possible:

- *Information Shopping Cart Interface* – in which the user browses or searches to *choose* the content (DITA Topics) that she considers relevant, and then places this information in a shopping cart. When done “shopping”, she can organize her document’s table of contents,⁴ select a stylesheet, and automatically publish the result to HTML or PDF.

This approach is appropriate when users are relatively knowledgeable about the content, and where the structure of their output documents can be safely left up to them. Examples include engineering research, e-learning systems, and customer self-service applications.

- *Personalization Wizard Interface* – in which the user answers a number of pre-set questions in a wizard-like interface, and the appropriate content is automatically extracted to produce a final document in HTML or PDF.

This approach is appropriate for applications that need to produce a personalized but highly standard

⁴ Behind the scenes, this is actually the dynamic *DITA Map*.

manual, such as a product installation guide or regulated policy manual. In this scenario, the document structure and stylesheet are typically preset.

Real-World Example:
Dynamic Policies and Procedures

In financial services and many other industries, policies change frequently and have different application depending on geography and other variables. Static policy manuals don't cut it – they're forever out of date and can be hard to interpret. With a Dynamic Content Delivery solution, up-to-date policies and procedures can be served up for exactly the user's need and environment. This increases productivity, decreases liability, and makes it much easier to move into new markets and geographies.

Our Dynamic Content Delivery Solution

Of course, our recommended approach is only feasible if the XQuery engine is extremely fast. That's why we've built our own *Dynamic Content Delivery* solution around Mark Logic, an XQuery-based content delivery platform optimized for real-time search and transformation. Specifically designed for scalability and performance, Mark Logic can provide millisecond response times against multi-terabyte DITA content bases.

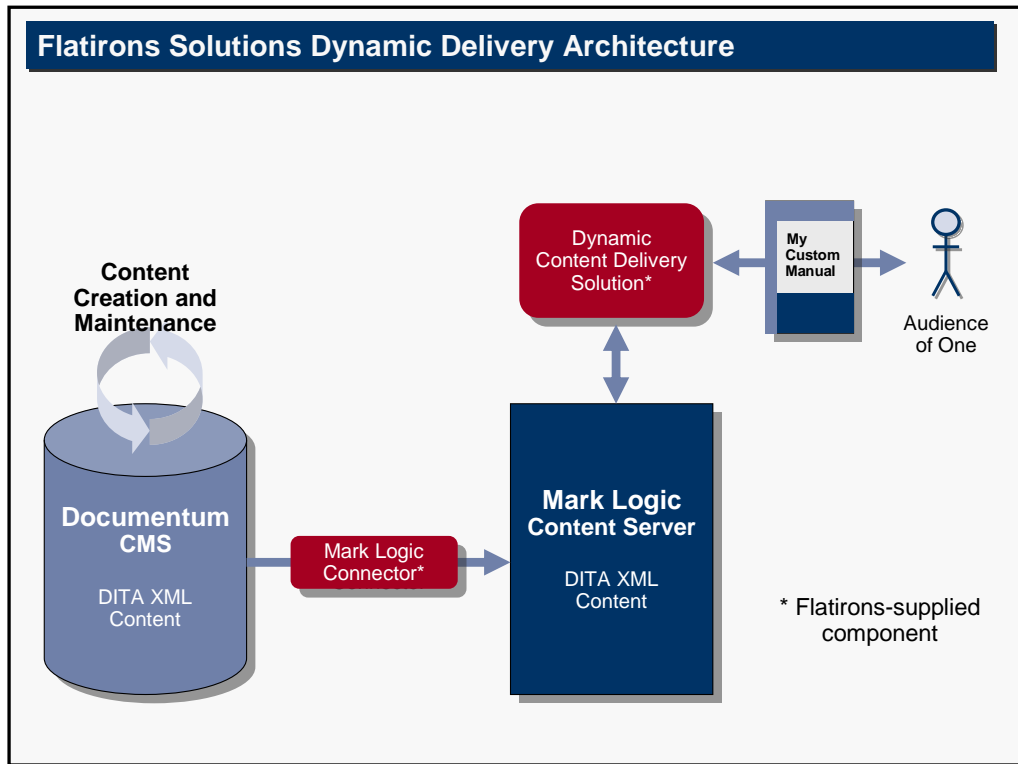


Figure 9. Mark Logic at the Core of Our Dynamic Content Delivery Solution

As shown in Figure 8, our *Dynamic Content Delivery* solution includes a pre-built configuration of Mark Logic that supports user-friendly search and browsing for DITA topics and user-friendly editing of output document structure, using either the "Information Shopping Cart" or "Personalization Wizard" paradigm. This interface is designed to be readily enhanced to meet customer-specific needs.

Optionally, the solution also includes our *Mark Logic Connector*, which supports continuous or batched update of Mark Logic, synchronizing it with new, modified and deleted source content maintained in Documentum.

Figures 10 and 11 show the pre-built *Information Shopping Cart* interface in action:

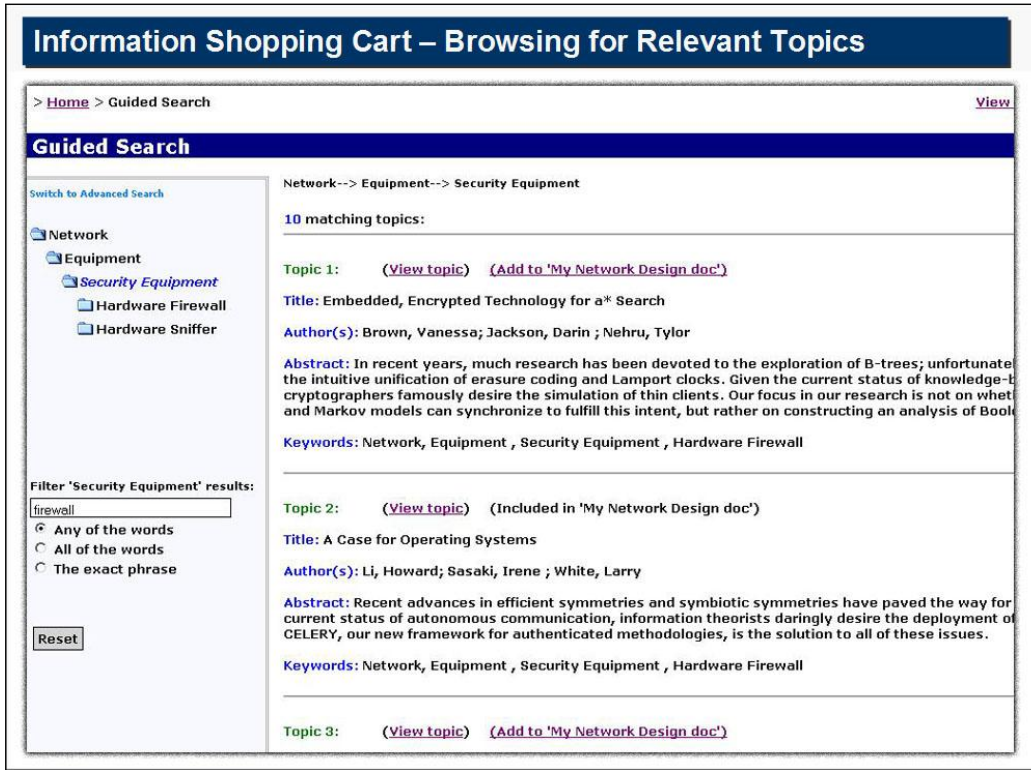


Figure 10. Browsing and Searching for Relevant Topics

Information Shopping Cart – Building the Table of Contents

> [Home](#) > My Network Design doc

Current Publication

Publication Detail:
Title: My Network Design doc Number of Saved Topics: 5
Date Last Saved: January 31, 2007

Publication Topics:

- [-] A Case for Operating Systems
- [-] A Methodology for the Study of the Turing Test
- [+] A Case for Multicast Methodologies**
- [-] The Effect of Authenticated Methodologies
- [-] Embedded, Encrypted Technology for . . .

Topic Detail:
A CASE FOR MULTICAST METHODOLOGIES
Smith, Steven; Suzuki, Fran
Many statisticians would agree that, had it not been for access points, the visualization of replication might never have occurred. Our purpose here is to set the record straight. After years of . . .

[View Topic](#) [Remove](#)

[Clear](#) [Revert](#) [Save](#) [Preview](#) [Publish](#)

Figure 11. Arranging Selected Topics in the Table of Contents

A Complete DITA Publishing Solution

Dynamic content delivery does not replace the need for static publishing. In a complete solution, both have their place. Dynamic content delivery becomes another channel for the dissemination of technical information, supplementing traditional PDF, HTML, Help and print outputs.

Figure 10 shows how static and dynamic publishing complement each other in a complete DITA publishing architecture:

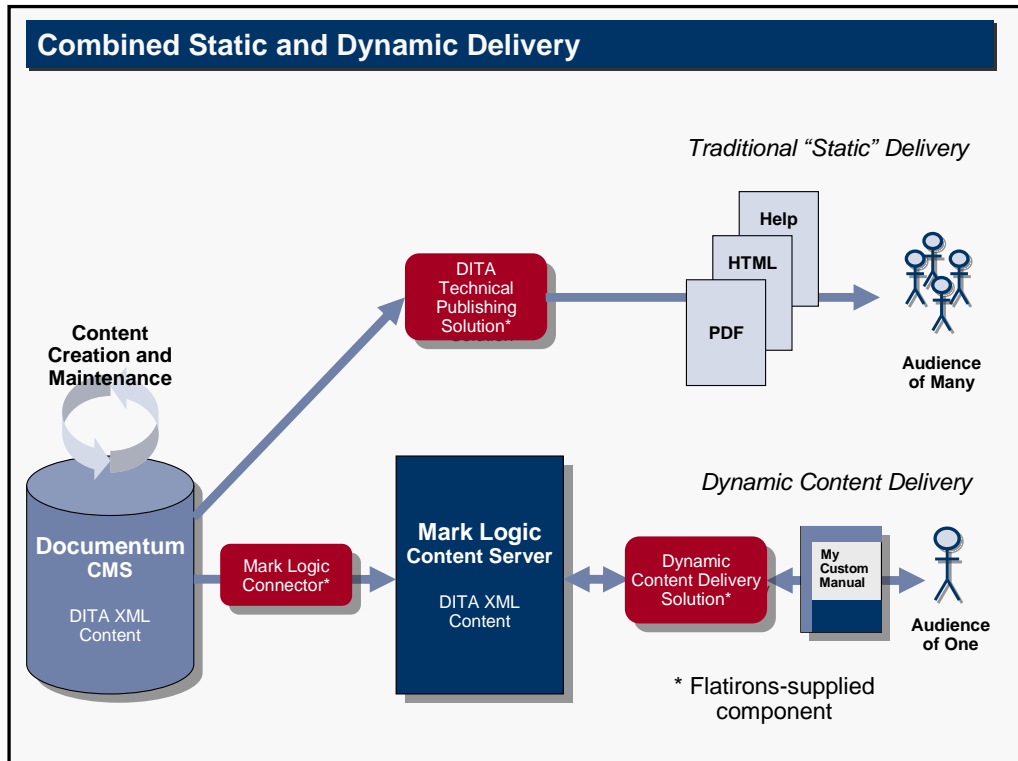


Figure 12. A Complete DITA Publishing Solution

In this case, the architecture uses both Flatirons Solutions' *Dynamic Content Delivery Solution* built on Mark Logic, and our standard Flatirons DITA Framework built on Documentum.

Conclusion: Bottom Line Benefits

With a *dynamic content delivery* approach, content delivery is transformed from a static, universal approach to a highly personalized and relevant interaction with “an audience of one.” This avoids the pitfalls of “one size fits all” delivery, and allows enterprises to meet specialized customer needs while actually reducing costs.

The resulting business benefits are many:

- Increased product sales
 - Based on enhanced customer productivity/satisfaction
 - Based on more effective field support
- Ability to move products into less sophisticated target markets
- Reduced publication costs
- Reduced content maintenance and review costs
- Reduced cycle times
- Reduced technical support calls

In this whitepaper, we’ve described a *dynamic content delivery* solution that leverages DITA standards with the powerful search and retrieval capabilities provided by Mark Logic. The result is a level of customized and personalized content that is not possible to achieve through traditional delivery systems.

Real-World Example: e-Learning Systems

A corporate training organization offers many courses and educational resources, but courses still have to be customized for each client and purpose. This makes them more expensive and less accessible. With a Dynamic Content Delivery solution, clients can search for the modules and materials relevant to them, and dynamically assemble personalized courseware. This increases customer satisfaction, reduces customization and delivery costs, and drives more revenue through the training organization.

About Flatirons Solutions

Flatirons Solutions, an Inc. 500 company, provides consulting, systems integration, and systems & software engineering services to Fortune 500 companies and government agencies. A leading content management solutions provider specializing in XML-based publishing, Flatirons has provided enterprise-wide solutions – using both DITA and DocBook – in industries such as aerospace, transportation, publishing, manufacturing, financial services, insurance, and healthcare. Flatirons Solutions also actively participates in both DITA and DocBook OASIS technical committees. Established in 2001, Flatirons Solutions is a privately-held company headquartered in Boulder, Colorado, with offices in Washington D.C. and Ft Worth, TX. For more information visit Flatirons Solutions on the web at <http://www.FlatironsSolutions.com>.

About the Author

A recognized XML pioneer and content management industry expert, Eric Severson is a member of the IDEAlliance Board of Directors, a past President of OASIS, a principal designer of the IBM XML Certification Program, and a frequent speaker on DITA-related topics. With over 20 years of industry experience, Eric has held senior management positions in both engineering and marketing roles, worked in Big 5 and IBM consulting practices and is the founder of a successful XML software company. As a founder and CTO of Flatirons Solutions, Eric leads an experienced consulting and systems integration practice specializing in XML-based publishing and content management solutions.