In this Issue

Feature Article
Information Architecture is Just Plain Fun!
An Interview with Lou Rosenfeld

In planning the second issue of *The Rockley Report*, it was unanimous that an issue on Information Architecture would not be complete without hearing from Lou Rosenfeld, information architecture "guru" and co-author of *Information Architecture for the Word Wide Web*, now in its second edition. We posed a number of questions to learn how he got involved in information architecture, his views on information architecture for the web, and how he sees information architecture extending beyond the web, potentially to all content created, used, and stored throughout an organization. According to Rosenfeld, "structuring, labeling, and organizing information is just plain fun. Well, at least for the oddfellows among us who are into that sort of thing." We happen to agree and are happy to share with you our interview with Lou Rosenfeld.

Read more on page 3 ...

Best Practices
Strategies for Optimum Reuse

Reuse is a critical component of a unified content strategy. At each stage of developing the information architecture, information architects refine the reuse strategy to reflect multiple perspectives of reuse, culminating in an optimal reuse plan. This article provides guidance on how to achieve optimum reuse.

Read more on page 7 ...

Information Architecture
Semantic vs Generic Elements

As you begin to model your content you will be faced with the issue of whether to create semantic models or generic models. This article reviews the pros and cons of naming your elements semantically and provides some guidelines for when to name elements semantically.

Read more on page 11 ...

Tools and Technology
Implementation: Issues with Granularity

Making the transition from document management to content management means that you have to look within documents for the structure of your content management system. How big should the pieces of content in your system be? There are many factors that affect the physical granularity of the content you manage.

Read more on page 13 ...

Editorial

Information Architecture—the Key to Successful Content Management

T his issue of *The Rockley Report* focuses on a topic "near and dear"to our hearts at The Rockley Group. In this issue, we explore Information Architecture, including discussions on optimum reuse, granularity, and implementing models to support granularity. We also discuss some of the issues related to information architecture, and how they might affect your authoring team. Why is information architecture so important to us? Information architecture defines your content management strategy; it outlines how your content will be structured, where and how it will be reused, how granular it will be, what metadata will apply to it, how it will be stored, and how it will be retrieved. Information architecture is the backbone to any content management implementation, regardless of its scope. After all, you wouldn't start building a house, even a small one, without first deciding how the house will be structured. Likewise, you need to define your information architecture before implementing a content management system, even if it's just within your department.

No issue on information architecture would be complete without hearing from Lou Rosenfeld, who helped create the profession of information architecture, co-authored its leading text, and was president of its best-known consulting firm for seven years. The name Lou Rosenfeld is synonymous with Information Architecture and we're pleased to feature "the Rosenfeld interview" in which he shares—among other things—his views on the "fun" involved in information architecture. We then go on to explore strategies for optimum reuse (just how granular should your content be?), the pros and cons of semantic vs. generic elements, and how to implement the granularity reflected in the information architecture. We also feature an article on the ongoing debate between buying vs. building a DTD and in our People, Processes, and Change section, we discuss some of the issues you may face in introducing the concepts of information architecture to your team, and suggest strategies for overcoming them. Our case study rounds out the discussion of information architecture; Pat Waychoff, a single sourcing visionary and strategist for HP Network Storage Solutions, describes how a content management implementation is not really about the technology—it's about the methodology.

We hope you enjoy this issue of *The Rockley Report* and welcome your feedback. Please send comments, as well as suggestions for stories in future issues to kostur@rockley.com. Our Call for Submissions describes the kind of stories we're looking for and how you can submit articles for publication in future issues.

Feature Article

Information Architecture is Just Plain Fun!
An Interview with Lou Rosenfeld

Lou Rosenfeld
Information Architecture Consultant
lou@louisrosenfeld.com

In planning the second issue of *The Rockley Report*, it was unanimous that an issue on Information Architecture would not be complete without hearing from Lou Rosenfeld, information architecture "guru" and co-author of *Information Architecture for the Word Wide Web*, now in its second edition. We posed a number of questions to learn how he got involved in information architecture, his views on information architecture for the web, and how he sees information architecture extending beyond the web, potentially to all content created, used, and stored throughout an organization. According to Rosenfeld, "structuring, labeling, and organizing information is just plain fun. Well, at least for the oddfellows among us who are into that sort of thing." We happen to agree and are happy to share with you our interview with Lou Rosenfeld. Be sure to also check out Rosenfeld's web site at
www.louisrosenfeld.com.

Q. What is your background?

After receiving a BA in history from the University of Michigan in 1987, I spent a short stint in that hell-on-earth known as retail sales. That experience chased me back to school, where a masters in library science seemed like a safe bet. I'd heard that there were lots of new information technologies that could prove useful both within and outside the library environment.

In those days, you only needed to be a month ahead of your peers to be a tech guru among librarians. I managed that float well, taking on various positions at Michigan as an instructor, researcher, and technology manager before ending up back in grad school to begin work on a PhD. Two years at the bottom of that particular pyramid made the dog-eat-dog of the private sector look warm and fuzzy. I'd already begun

my consulting firm, Argus Associates, a few years earlier, and I left academia for good in 1994 to devote myself to Argus full-time.

Argus went on to be recognized as the leading consulting firm in information architecture, serving many Fortune 500s and other large enterprises and helping define the profession along the way. Argus reached approximately forty employees before flaming out in 2001, when corporations suddenly erased the budget line item that covered consulting services in new, abstract, difficult-to-prove-ROI, yet extremely important areas like information architecture. Since then I've been an independent IA consultant (more about me at www.louisrosenfeld.com).

Q. What drew you to Information Architecture?

Let's face it: structuring, labeling, and organizing information is just plain fun. Well, at least for the oddfellows among us who are into that sort of thing.

Looking forward from, say, 1994, IA presented unexplored terrain as the Web and related technologies were beginning to explode. Someone was going to have to organize all that content.

And finally, how many opportunities do we get to help define and develop a new profession? Everyone involved in IA is doing just that; it's hard not to be drawn to a field that is growing and morphing before your eyes.

Q. What factors made it clear to you that information architecture is required for effective web sites?

Even ten years ago, both end users and content managers were straining under the weight of huge, complex web sites and other information systems. It was clear that this scalar problem was only going to get worse for two reasons. The first we all know about: the information explosion that, according to some, is

3

doubling humanity's total output of information every five years. The second, lesser-known factor was described to me by a client at AT&T as "ROT": Redundant, Outdated, and Trivial content. What's good today will be ROT

tomorrow. IA can help both content managers and end users deal with these problems be helping make ever-larger information systems better at supporting findability, while providing guidance in minimizing content ROT.

Q. Recently you have been doing a lot of work in the area of Enterprise Information Architecture. Can you describe this area and its value to organizations?

Generic or "traditional" IA is agnostic in terms of how best to design and implement an information architecture—just use the methods that make the most sense, choose the subset of ways to connect users to content that gives you the most bang for your buck, and so forth. IA "textbooks" like *Information Architecture for the World Wide Web* (Louis Rosenfeld & Peter Morville, O'Reilly & Associates, 2002) present a broad palette of architectural options, and it's up to the information architect to choose wisely.

Enterprise IA, by contrast, is much more proscriptive.

(www.louisrosenfeld.com/presentations/seminars/eia/).

My hope is that a well-designed enterprise IA will not only aid users in their efforts to find information in the enterprise environment, but also save information architects—and their employers—from wasting huge amounts of time and money as they try to develop useful, findable content.

Q. Tell us about the methodologies you used to develop best practices for information architecture for the web.

Individual methods are interesting, but the important thing is to assemble and utilize a whole bunch of them in a logical way. So instead of talking about individual methods, I think it's better to think about how they fit together. My take on IA methodology is

4

so we'd better take a balanced approach to selecting the methods we'll use on a given project.

For example, I might utilize card-sorting and search log analysis to understand what's inside users' heads, content inventory and modeling to know what content is available and how its pieces relate to each other, and stakeholder interviews, resource analysis, and competitive benchmarking to make sure that the IA design would work well in a given business context. For various reasons, I might use a different set of methods for another project, but the key is always to maintain balance across users, content, and context.

Q. Do you feel that the term information architecture is confined to the Web? Please explain.

Absolutely not. Consider the information architecture of that most familiar information system, the book.

Well, it's kind of hard, isn't it? We take tables of contents, indices, pagination, chapters, and the details on the spine and cover for granted. After all, we grow up using this medium and its conventions.

Designers of Web sites, on the other hand, don't benefit from the trial and error of centuries of predecessors. And Web sites allow for designs that are much more flexible and therefore more complex. That's why people are so interested in IA for web sites when compared with traditional media; there are precious few conventions, and we can influence what may ultimately become conventions.

Of course, if the pundits are right, the Web and other media will all eventually converge, rendering this debate moot.

Q. In the last issue of *The Rockley Report*, you provided a quote for our article on the "Information Architecture of Content Management." Can you describe what IA for CMS entails? Where do you see IA for CMS fitting into the area of IA?

Sorry to over-anthropomorphize here, but I see information architecture gleefully holding a squirming content management's feet to the fire, ensuring that CM, in its zeal to support those users known as content owners, doesn't forget those other users, namely end users.

I remember attending my first CM conference a few years ago and being dumbstruck at how surprised many attendees were when they were reminded that their work impacts end users. What's the point of doing CM if end users are left out of the equation? IA provides tools, techniques, and expertise to help content management benefit from metadata and integrate well with search and navigation systems, thereby delivering value to end users and content owners alike.

You might also say that IA provides a model for how CM might develop as a profession. IA is technology-agnostic. Until content managers start thinking this way, and focus on users and the processes that serve them, CM will continue to be dominated by CMS vendors to the detriment of all.

Q. Where do you see information architecture going in the future?

I'm convinced we'll see two major growth areas over the next decade: 1) IA for enterprise environments; and 2) IA for global environments, where L10n and i18n become critical design considerations. I'm starting to see many clients grappling with both areas concurrently.

As far as where IA is practiced, many of us witnessed the exodus of information architects from agencies to in-house positions over the past four years. Information architects were often hired in the wake of troubled CMS and search engine implementations, where it was realized that technology alone couldn't solve all the world's ills.

However, as the population of in-house information architects reaches a critical mass, many senior IAs are leaving the warm embrace of full-time employment to provide either strategic or highly-specialized consulting and training to support those in-house folks. I'd be surprised if the same thing wasn't happening in CM.

Q. How would you recommend someone get started in IA?

You can now actually get an advanced degree in information architecture. If you'd like to know more, the Asilomar Institute for Information Architecture pro-

vides useful links to IA education resources
(www.aifia.org/pg/education.php).

But if you don't have a couple of spare years to pursue
another degree, I'd return to my three-circle Venn dia-
gram.

Consider "majoring" in one of those circles and
"minoring" in the other two. For example, maybe
you're already quite knowledgeable about *context*—
you might have a background in organizational psy-
chology or management, and having worked at your
company for a dozen years, you know more than you'd
like about its particularities. With business context as
your "major," you might begin your "minor" in *users*
by reading books on human-computer interaction or
reference librarianship, and you might learn more
about *content* by attending a conference on technical
communication or hanging around with the journalists
who congregate at your local watering hole.

However you pursue your studies, formal or informal
as they may be, seek balance among those three circles
and you'll be in great shape.

Best Practices

Strategies for Optimum Reuse

Ann Rockley
President
The Rockley Group Inc.
rockley@rockley.com

Cori Czekaj
Solutions Architect
Arbortext
cczekaj@arbortext.com

Reuse is a critical component of a unified content strategy. At each stage of developing the information architecture, information architects refine the reuse strategy to reflect multiple perspectives of reuse, culminating in an optimal reuse plan. This article provides guidance on how to achieve optimum reuse.

Every project we undertake involves reuse within information products (e.g., in the body of a document and in the quick reference section in the appendix), across information products (e.g., call center materials, brochures, documentation, training), and across media (e.g., web, paper, wireless), but designing optimum reuse requires a balance between what is *desired* with what is *possible* from both an authoring and a technical perspective. An optimum reuse strategy is based on our analysis of the documents, the potential for reuse, and the end user content requirements, as well as on our understanding of the technical capabilities/implementation of the content management system and its users. Optimum reuse is also impacted by granularity and the types of people who use the reusable objects.

Granularity

Reuse is based on granularity. Granularity identifies the smallest piece of information that is reusable. There are typically multiple levels of reuse within your information set. In one instance you may reuse large sections of information unchanged; in others, you may reuse content at the sentence or even the word level.

Defining the appropriate level of granularity can be challenging because if the granularity is too large, you may have to delete a significant portion of the content that is not applicable when you reuse an object. This may result in a lot of derivative reuse. Alternatively, if the granularity is too fine (small), content is more difficult to manage and track (e.g., the performance of some content management systems may be impacted by large numbers of small objects).

Granularity will vary depending upon the users of the reusable objects (e.g., authors, reviewers and translators) and the technology.

Authors

Authors tend to prefer a more granular approach to content, in which the elements are identified semantically. The semantic structure gives them very clear guidelines on what to write (see the article on *Semantic vs generic elements* in the *Information Architecture* section of this issue). When elements are named semantically, they can be easily identified for storage as separate content objects. Fine granularity also makes it easy for authors to identify content for reuse or filtering (e.g., steps 1–3 in a procedure are relevant to all customers, but step 4 is only relevant to expert users).

Reviewers/translators

One of the wonderful features of object-oriented content management is that we can identify the elements that have changed and route only the changed elements to reviewers or translators. However, reviewing small elements of content out of context of the surrounding content can be problematic because the element may not make sense on its own, or may appear inaccurate the way it is used. Similarly, translators cannot translate an element unless they understand the context in which it is being used. Reviewers and translators may require a higher level of granularity, such as a sub-section or even a complete section. Compliance and legal officers may also want to see the entire document to ensure that the content is correct in the overall context. To achieve this balance, consider routing large elements for review and translation, but ensure that the element for review is clearly identified so that reviewers/translators work with only the required elements, not ones they have already seen. In this way, reviewers and translators can see content in the overall context and make appropriate changes to the specified information element.

Best Practices

Technology

Granularity is also impacted by technology. Too small a level of granularity can be very difficult to manage, yet too large a level of granularity may make it very difficult for users to retrieve and reuse elements. Look at your content and determine if you need to "burst" (break your content into its component parts and store the content at that level of granularity in the CMS) at every element. Would nested reuse make sense instead of bursting? Remember that authors don't want to look for small fragments of content either. Can you take the element size to the next level (e.g., instead of a small element, such as a step, would a whole procedure be a more realistic size for authors to retrieve and for the CMS to manage)? The larger the object, the easier it is for authors to manage and retrieve. Larger content objects can also be more technology friendly. The performance of content management and composition systems is often related to the number of information elements being stored and retrieved.

Implementation challenges

You've done all your modeling and you know what type of reuse is required, but now you have to figure out how to implement it. This section describes some of the implementation challenges for authors, architects, editors/reviewers/translators, and implementers, and suggests how you can overcome them.

Authors

Implementation issues for authors include:

- Granularity
- Metadata

As previously described, authors like to be guided in their authoring by richly semantic models. In an XML-based content management system every element can be burst apart and stored separately. It is easy to burst content based on its semantic structure because semantically-named elements have more meaning for retrieval (e.g., it makes more sense to retrieve a "procedure" than an "ordered list"). However, if you have a very rich semantic structure (e.g., procedure contains steps and steps contain action and result) and you choose to burst the content at every semantic element, authors may have difficulty searching for appropriate content. Authors generally don't want to search for small reusable fragments (e.g., steps). Consider a nested reuse strategy, allowing authors to have small

fragments of content within a single element. Nested reuse also solves the problem of having to write out of context (e.g., writing step 4 on its own, out of context of the entire procedure). Nested reuse allows authors to write a complete set of information in context, yet filter out specific pieces of the content where appropriate.

Also consider providing a semantic authoring stylesheet that "maps" to a more generic structure (e.g., DTD/Schema). This ensures that authors have the semantic structure for authoring, but a more simplified DTD/Schema for creation and maintenance.

Metadata is also needed to identify and retrieve content. If you don't have enough metadata you may not be able to effectively retrieve content; however, too much metadata makes the authoring task onerous. Authors can be intimidated by having to add too much metadata. Wherever possible consider automating the addition of metadata through functions like inheritance and reduced metadata lists (e.g., show only the metadata that is relevant in the context of the content). Also, consider using a semantic authoring template that maps to a generic DTD/schema and maps the semantic element in the authoring template to appropriate metadata.

Content/information architect

Content/information architects are responsible for designing an optimum reuse strategy that is represented in the models and in the implementation strategy. Implementation issues for content/information architects include:

- Understanding the full breadth of content and organizational requirements
- Simplifying content models yet at the same time optimizing authoring and implementation
- Providing appropriate guidelines for reuse

Content/information architects need to see the whole breadth of content in an organization (or a representative sample) and look for opportunities for reuse. If they try to model content for a specific situation, they may model it only for use in that situation, which may be ineffective in the wider context of the organization.

Once the preliminary models are created and validated, architects need to simplify them. For example, they need to determine where semantic elements are most appropriate and where generic elements can be more effectively used. In doing so, they make have to compromise to accommodate authoring requirements,

Best Practices

reuse requirements, and technology requirements (see *Issues with Granularity* in the Tools and Technology Section).

Once modeling is complete and appropriate templates, DTD/schema, and stylesheets are in development, architects need to develop and enforce policies for authoring and reuse, and develop test procedures to ensure that content continues to meet quality standards.

Editors, reviewers, and translators

Editors, reviewers, and translators also need to understand reuse, specifically, they need to understand that the content they review/translate can exist in multiple locations.

In a content reuse environment, editors, reviewers, and translators no longer work with content that appears in only one location; they work with content that may be used in multiple locations. In a reuse environment, authors need to write content so it is effective everywhere it is used. Reviewers need to review it with the understanding that it can be used in multiple locations. Reviewers often want to change content to reflect their unique requirements; however, changes to a reusable element change the element everywhere it appears. If reviewers look at the content in the context of how it is reused, they may see that their changes are not required.

To understand the impact on changes to reusable content, editors, reviewers, and translators need to understand the current and future contexts for content they are reviewing/translating. Consider providing training to show them how content can be reused. Also, consider providing them with a visual context for the content they are reviewing so they understand which content has already been signed off and which content requires their attention. Editors, reviewers, and translators should also be taught how to use the content management system to see where reusable elements are reused. Seeing elements in multiple contexts helps them to understand the impact of their changes.

Technology implementers

Technology implementers need to implement the vision of the architecture. Technology implementers must balance the vision while optimizing the technology and in doing so, must work closely with everyone on the design team to ensure they can effectively implement the architecture to meet everyone's needs.

Technology implementers need to implement the models and reuse strategy to optimize both the content management user requirements and the technology. To do this they need to work with the content/information architect to simplify the architecture by pointing out how the architecture could be supported in the technology. To do this effectively, technology implementers should attend some of the architecture working sessions so they understand the information/authoring needs driving the requirements. It is also very helpful if the content/information architects receive training on the technology so they understand the information architecture in the context of the tool.

Risks

There are a number of risks involved in implementing a reuse strategy if you don't take the time to do it right. Some of the risk factors include:

- Not knowing how content will be managed

  You can't complete your models and your implementation strategy until you have selected your technology. Your technology will impact your strategy, because different tools support reuse differently. You have to understand the capabilities of the tool and design accordingly.

- Designing the content management system for one audience

  You need to design the content management system to support all potential users. You need to support authoring, editing, review, translation, and end user requirements. You need to consider all the requirements when you design your models, workflow, and publishing.

- Failing to coordinate, manage, and/or demand communication between implementation resources

  An effective implementation strategy requires a lot of communication with users (e.g., authors and reviewers), architects, and implementers. You need to ensure that communication happens between all of the stakeholders and the design/implementation team to ensure that needs and requirements are clearly communicated and everyone is in agreement. Ensure that the implementers understand the requirements and can communi-

cate how the selected technology can best meet the needs of your reuse strategy.

- Not supporting multiple levels of reuse

  You will have multiple levels of reuse throughout your information set. Make sure that you optimize reuse while making it possible for authors to create content at the level of granularity that makes sense to them. And make sure that you provide editors, reviewers, and translators with the level of granularity they require. Don't forget that reuse needs to be optimized for end users as well. Make sure that your content management system can support all your requirements and you know how to optimize all user requirements with system functionality.

Summary

A successful strategy for optimum reuse requires that you determine the needs of content management users and plan for the level of granularity that supports their tasks. Communication is critical to success! Create an environment for information sharing and knowledge transfer during all phases of the project. Be sure to invest in planning, training, and knowledge transfer. Develop a realistic implementation plan that takes user requirements and technology capabilities into account. And, don't ignore the "usability" factor for all your users.

Information Architecture

Semantic vs Generic Elements

Ann Rockley, President
The Rockley Group Inc.
rockley@rockley.com

As you begin to model your content you will be faced with the issue of whether to create semantic models or generic models. This article reviews the pros and cons of naming your elements semantically and provides some guidelines for when to name elements semantically.

The word semantic refers to "meaning." An element that is named semantically is uniquely identified by its content. In other words, the name (label) of the element uniquely identifies the element. An element that is named generically, on the other hand, is identified by its common name, such as paragraph (para) or unordered list. For example, the semantic names in a product description could include Product name, Product overview, and Key features. Those same elements named generically would be Title followed by a Paragraph, followed by an Unordered list. The illustration below shows both a semantic model and its equivalent generic model in a number of common authoring formats:

Semantic		Generic	
	Word	FrameMaker	XML
Product description			
Product name	Heading 1	Heading1	title
Product overview	Normal	Body	para
Key features	Heading 2	Heading2	title
Feature	Normal	Bulleted	ulist

Value of semantic elements

Semantic elements can be very valuable in a structured writing environment. Semantic elements clearly define:

- The type of content that should be included in your information product

  Semantically-named elements clearly identify the required content and the structure of the content to be authored. Authors prefer semantic elements because the semantic labels guide them in the type of content to include, allowing them to focus on creating effective content instead of worrying about what to include.

- The content's "identity" for "manipulation" later

  Semantic names identify content so it can be selected for reuse, filtered out if inappropriate in a particular situation, or more effectively retrieved. For example, you might want to reuse "Product name" in another location in your content set. Because it has a unique name, you can retrieve it for reuse. Or in another situation you may only want the Product name and Product overview, but not the Features. If the Features are labeled semantically, you can easily filter them out. The semantic labels act like metadata so that you can retrieve content in a particular context (e.g., Information about Product X in a positioning statement).

Drawbacks of semantic elements

However, semantic elements have their drawbacks. These include:

- Authors see a lot more tags

  In a traditional authoring tool a large number of tags can be a problem because authors have to select from a long list of style tags. However, in a structured editor, only the elements that are valid in that portion of the structure are displayed, which limits the list authors must choose from. A long list of tags may still be confusing in a structured environment if you provide too many optional tags.

- A lot more coding

  It is a lot more work for an information technologist to create all the individual tags for your templates. Maintaining templates and stylesheets can also be more work with semantic elements.

- May limit reuse

  If you are using a validating structure like a DTD or schema, semantic elements may limit your reuse because you can reuse content only if the structure allows that element to exist in that context. If it doesn't (e.g., you have named the same

type of content differently for different situations), then you cannot reuse that content in multiple places.

Making the right decision

The right decision is using a combination of semantic and generic labels for your elements. Use semantic labels if they will help authors to create content more effectively or if content needs to be manipulated later. Use generic element labels when there is no added value to a semantic label (e.g., it may not be necessary to identify the first paragraph in a section as an Introduction, it could simply be a paragraph and your writing guidelines could recommend that authors include an introductory paragraph). Consider using semantic forms or authoring templates that map to a more generic DTD/schema to aid in authoring, or consider adding metadata rather than using semantic elements to identify content for manipulation

Summary

Semantically-named elements provide authors with the guidance they need to create consistently structured content. Semantically-named elements also make it possible to identify, reuse, and filter content. However, semantically-named elements are more difficult to create in a DTD/schema and stylesheets are more difficult to maintain. A well-planned content strategy uses semantic elements where they will be most valuable, but uses generic elements where no significant value will be realized.

Tools and Technology

Implementation: Issues with Granularity

Steve Manning
Senior Consultant
The Rockley Group
manning@rockley.com

Making the transition from document management to content management means that you have to look within documents for the structure of your content management system. How big should the pieces of content in your system be? There are many factors that affect the physical granularity of the content you manage.

There are many factors that affect the granularity of information you create, manage, and maintain. Factors include:

- The type of reuse you are supporting
- The nature of your authoring life cycle
- The technology of individual content management systems
- The data format of the content management system—XML or not

Types of reuse

We should begin with a discussion of types of reuse, but a limited discussion. When we talk about reuse, we usually talk about things like opportunistic and systematic. But, these are really *modes* of reuse. The types of reuse that affect granularity are filtered reuse and modular (or building-block) reuse.

In filtered reuse, authors provide all variants for a specific chunk of information in a single block of content, e.g., they include all the content to support all levels of users, or all outputs. The variations are identified by conditional tags or attributes of some type. When the block is converted into output formats, variations that are not needed for a specific output (e.g., the user guide) can be filtered out through stylesheets. In modular reuse, content is written in separate physical blocks (building blocks) of content. The blocks are then combined (either by the author or automatically by the system) to create the output.

Of course, these two types of reuse can be—and frequently are—combined to provide the necessary functionality for authoring and presentation. However, filtered reuse has fewer implications for implementation than modular reuse; content can be managed in large chunks, because it is filtered by the publication engine.

Granularity and modular authoring

In addition to reuse types, the granularity that you implement in your content management system may also depend on the authoring life cycle of your content.

In a collaborative authoring situation, where authors provide fragments of content, you may want to maintain those fragments as contiguous pieces in the content management system. This allows contributing authors to check out and check in only the content elements they need to work on. They do not need to navigate through entire chapters to find the section or sub-section they need to author. Providing contributing authors only the pieces they need helps them to focus on the individual task at hand. They do not need to see and be distracted by the rest of the document.

Where individual elements of content from the authoring chunks will be used in multiple places (building block style), it may be necessary to break the content into small pieces for storage and reuse, but reassemble the pieces for authors so they can edit and manipulate content elements in their entirety.

The effect of technology

It would be nice to say that all content management systems are created equally, and thus be equally capable of managing individual elements of content, but it is neither true, nor is it reasonable to expect it to be true. Many content management systems began life as document management systems, and were focused on the administration and management of document files. However, content management means having the ability to manage the individual chunks of content that have traditionally been combined in individual

files to make documents. In making the transition to content management, these document management systems opened up access to the elements of the document.

In contrast, there are also content management systems on the market that were built with the express purpose of manipulating the content elements that make up information products. Note the use here of "information products" and not "documents." The term "documents" implies paper. These content management systems grew in parallel with the Web and have therefore been designed to support both paper and online content.

Older systems tend not to support fine levels of granularity. They usually require that documents be "burst" or broken into individual pieces as they are checked into the repository. Frequently, this mechanism has been "bolted" onto an existing system to add the functionality to manage elements. This bursting mechanism must usually be configured into the system as a custom extension. The drawback to this is that the system becomes "hardwired", and changes are expensive to make. Also, if you want a fine level of granularity, where many elements are broken out and managed individually, the costs of customizing the system may become prohibitive.

Newer systems, especially those created specifically for managing fragments of content, have a significant advantage. Many of them manage individual elements as part of their base functionality and don't require an add-on. These systems may or may not actually require you to burst the content. Many of them are built to automatically burst the content, giving you access to all individual elements.

The ability of a content management system to manage and manipulate elements will be a key factor in the granularity of a system. Systems can manage elements to different degrees, and not all systems will be able to support your level of granularity.

XML

There are very few content management systems that do not support XML, which is a good thing. The very nature of XML, with clear boundaries of elements and a general orientation towards databases, makes it an excellent data format for managing content. Therefore, XML-based content management systems offer the

greatest flexibility for managing content at a very granular level.

Like the basic ability to manage elements, the way in which XML support has been built into a system can affect the level of granularity. Some systems have XML capability "bolted on", where XML content is transformed into something else for storage. Native XML systems, however, were created specifically to manage XML content.

Summary

The level of granularity that is right for your content management system depends on a number of factors (in addition to information architecture), including types of reuse, collaborative or local authoring, and the technology you are using to support your content management.

14

People, Processes, and Change

Issues in Information Architecture

Pamela Kostur
Senior Consultant
The Rockley Group
kostur@rockley.com

Behind a successful content management implementation is a solid information architecture that describes such things as how information products will be structured, where and how content (and structure) will be reused, as well as what metadata is required to identify how content is used, retrieved, and tracked. Information architecture forms the specification for a unified content strategy, but information architecture brings a new set of challenges to those whose job is to create and disseminate content. This article explores some of the issues specific to information architecture, including: teaching authors about information architecture; distinguishing between reusable content and reusable structure; visually representing information structure (documenting your information architecture); and documenting the architecture.

When organizations attempt to unify their content, they start by analyzing a select set of content to see how it is currently used and where it can be used. Along with an analysis of the content, organizations must also examine the processes they follow to create, review, and manage content. Once they have a thorough understanding of the information needs within the organization (or within a department or departments), they can start unifying content, first by designing a new structure to support consistency and reuse, then by implementing the structure with new processes, and often, with new tools. Building the information architecture that describes the new structure is critical to a unified content strategy. However, many authors struggle with concepts related to information architecture. They often have difficulty modeling their content, describing their structure semantically, visualizing structure within and across information products, and they have difficulty understanding the technology well enough to know what type—and how much—information to include in their models so they will be implemented according to the authoring scenarios they envision.

Teaching authors about information models

Once organizations have identified opportunities to unify content, they need to model the content they plan to unify. Creating information models is the first stage of designing your information architecture. Models formalize the structure of content; they form the framework upon which the unified content strategy is based. As such, they are critical to the success of a unified content strategy. In general, authors create documentation—brochures and other marketing materials, press releases, user guides, technical specifications, product descriptions, training materials, policies, procedures, etc.. Their focus is typically on writing and editing (for many different media), on assessing user requirements and writing documentation to meet those requirements. However, information modeling asks them to examine documentation from an architectural perspective and across a number of different information products. For most authors, this is a different way of looking at documentation. Instead of focusing on writing and editing—the "normal" scope of work for most authors—information modeling asks them to examine their content structurally.

When authors look at their content structurally, they examine how it can be put together so it is consistent and so elements can be reused across information products. Information modeling requires that authors determine:

- Which elements make up their information products and which element (based on granularity) belongs where
- Which elements share a reusable structure and which share reusable content
- The type of reuse that applies to each element (e.g., opportunistic, systematic, derivative, locked, nested)
- The semantic structure of each element, which uniquely identifies the element's content. Semantic tags are based on content (as opposed to for-

15

mat), helping authors to identify elements for reuse and to structure them consistently
- The metadata that applies to each element
- The base structure (or presentation) of each element (e.g., para, ulist, olist, title, and so on)

Much of this is new to authors so before starting the modeling work, they may need to be educated in the process and language of information architecture (e.g., elements, granularity, the types of reuse, semantic structure, metadata). The modeling team also needs to go through some practice modeling exercises before tackling their own work. Once they have modeled something in which they have no vested interest, they are more comfortable modeling their own information products. Choosing members of the modeling team is also important. They need objectivity, plus the experience to help them to make decisions about the structure of their content. Accordingly, it is beneficial to have both experienced and newer members of authoring groups on modeling teams. An information modeling team will also need someone well versed in information architecture to lead it and to keep the team on track.

Distinguishing reusable structure from reusable content

One aspect of modeling that seems particularly problematic for authors is distinguishing between reusable structure and reusable content. Within a model, there may be elements that share a reusable structure, and others that share both structure and content. Both types of reuse need to be reflected in the model. For example, procedures (or portions of procedures) may have a reusable structure, so that wherever procedures appear, they are structured consistently. However, even though procedures may share a common structure, they do not always share content. Wherever elements share both content and structure, both types of reuse must be indicated in the model.

Illustrating both reusable structure and reusable content adds another dimension to the information model. An information model is typically hierarchical. It shows the order in which elements appear within an information product. Where elements share reusable structure within that information product, or across other ones, we track the reusable structure separately, apart from the information product hierarchy. We do this by "linking" to another model that describes the structure of each reusable element, and indicates which parts of the structure belong where. This way,

the main "hierarchical" model for an information product doesn't get bogged down in levels of reusable structure, making the hierarchy easier to see. Plus, we don't have to repeat reusable element structure over and over again; we simply link to the reusable elements.

Furthermore, if an element takes reusable content, this needs to be indicated separately, for example, in a "reusable content" column on the information product model. In this way, the reusable content is kept separate from the reusable structure. In the reusable content column, the modeling team indicates if an element takes reusable content, if that content is inserted systematically or if authors look for it and insert it opportunistically, and if the reusable content is locked or editable. As before, this requires that authors think about content from an architectural perspective—both content and structure—and that they document structure and content reuse in a way that makes sense to their team, and to those who will be implementing the models. This poses a challenge for many organizations.

Visualizing structure

Regardless of how well the modeling team understands information architecture, visualizing structure remains difficult. Besides showing hierarchy, information models also show relationships among elements; for example, the use of one element may be dependent on another element being included. One of the best ways to represent hierarchy and relationships is in a spreadsheet, with the left column representing the semantic name of the element, and the columns to the right representing the element's usage across information products. If an element belongs in an information product, we use an "M" to indicate its usage is mandatory or an "O" to indicate it's optional. The model also carries other information, such as the reusable content notes, any production or writing notes, and the metadata that applies to the element.

Because models contain so much information, they are difficult for authors to learn to read, and hence, verify. Models are usually "textual", with hierarchy and relationships being shown through a tabular structure. But, the models that represent information products don't exactly look like information products; in fact, a house blueprint looks more like a house than an information model looks like an information product.

To assist authors in visualizing information products in their spreadsheet representation, we recommend

16

"connecting" the model to the information products being modeled, possibly marking up actual information products to correspond to the parts of the model. Content is multi-dimensional, and it's difficult to represent its structure visually without a sample that accompanies the model. This marked-up information product is also useful for the DTD/authoring template developers; it gives them a clearer idea of what the authoring template should allow authors to do and what the output should look like. However, if you are making significant changes to the information products being modeled (and we recommend modeling what you want your information products to be, not necessarily what they are), or if you are modeling new information products, you may need to create a sample information product to go along with the model, marking it up to correspond with the model. Without information product samples that correspond to the information model, it's also very difficult for authors to verify the models.

Documenting the architecture

Once models are created and verified, they need to be implemented in DTDs or authoring templates, and the content needs to be managed in a CMS (content management system) or document server. The biggest issue in implementing models is ensuring that their implementation corresponds with what the modeling team envisions. Just like an architect gives a house builder a blueprint for a house and the builder follows it, an information architect gives the DTD/authoring template developer (and the person responsible for setting up the CMS) the blueprint for the information products. But, the model does not stand alone; it needs an implementation strategy to accompany it.

The model illustrates the hierarchical structure of the information products; it shows the order of elements within an information product and where elements (both structure and content) are reused across information products. It also provides implementation details such the metadata, the presentation of each element (e.g., para, ulist, title), their frequency, etc. However, the model does not illustrate the authoring process, which is a critical part of the authoring template. Accordingly, along with the model, you need an "implementation strategy" that fully describes how you envision the authoring process, from the time authors sit down to write or edit the document, through to its being stored in the CMS or on a server. This strategy describes the implementation of your

information architecture and should include such things as:

- How authors select the template for the various information products
- How the template is presented to them (e.g., in chapters/sections or in its entirety) and what rules apply to the template (e.g., Will optional elements be displayed? Will authors be able to proceed without writing the mandatory elements?)
- How systematically reusable content is retrieved and pulled into the template
- How authors search for opportunistically reusable content
- How content is tracked as it is updated/used in other information products, including how derivatives are tracked
- How locked content is updated
- How information products that have reusable content are updated when the reusable content elements are updated in other information products (e.g., are authors notified or is updating done automatically, without notifying authors)
- How metadata is added to the elements
- How content is stored or checked back into the CMS

Many implementation issues are tool dependent and authors should work with the DTD/authoring template developers as well as the CMS experts to ensure their authoring vision will work in the selected tools. It's also useful to have the DTD/authoring template developers as members of the team defining your information architecture, so they share the same interpretations as the authors who create it and so they can guide authors in what is technically possible, given the tools they are using to create and manage their content

Summary

As information architects, or as those heading content management projects, we need to describe unified content and information structure in more familiar ways, at the beginning of projects and throughout their implementation. Looking at content management from an information perspective (i.e., beyond the purely systems perspective) is new for many of us. Everyone working on a unified content project—from management supplying the funding and the support through to the authors developing the models and the IT departments implementing and supporting the technology—needs a common understanding of what the

models are and what they're intended to do. With that in mind, we need to:

- Educate everyone who will be involved in the unified content project about what unified content is and how information architecture supports it
- Teach everyone who will be participating in defining the information architecture (and its implementation) the language and the concepts of information architecture
- Find ways to create more visual information models, including mapping information models against existing information products (even if you have to create new ones), thus making them into visual aids that support the models
- Document the model in an "implementation strategy" that describes, in terms that authors are familiar with, how the model will be implemented, from the time they start writing, to the time the content is published and stored in the CMS
- Share the strategy with those who are creating the DTDs/templates and with those who are designing how the content will be stored/retrieved/tracked

A unified content strategy is only as successful as the blueprint on which it is based. A solid information architecture forms the basis for the unified content strategy and it's critical that everyone involved in the project understands the strategy behind the architecture is, how to read the models, how to implement them, and how to move from their current authoring environment to the one described in the strategy.

Gaining Management Support

XML and DTDs: The Buy vs Build Argument

Steve Manning
Senior Consultant
The Rockley Group
manning@rockley.com

When deciding to adopt XML as an authoring standard/backbone, one has to consider the question, "Do we need to create our own DTD?" Some will tell you that there are ready-made DTDs out there for you to grab and use. Others will tell you that you must either start from scratch or forget about it. So what's the answer? Here are the pluses and minuses in the Buy vs. Build debate to help you decide what makes sense for you and get the DTD you need.

It's a common question for anyone contemplating a move to XML: "Do we need to create our own DTD?" The answer, guaranteed to frustrate you is, "Maybe ... maybe not."

You have 4 choices:

- Build your own DTD from scratch
- Adopt an existing (possibly industry-standard) DTD as is
- Modify an existing DTD
- Create your own DTD as a layer on top of an existing/industry standard DTD

What's available

Let's start by discussing what's available. DTD is used here to represent any predefined, "validatable" structure. That is, the structure could be a DTD (XML or SGML), or it could be a schema (the XML-based equivalent to the DTD). Both types of structures are readily available for use.

Note that you don't necessarily have to buy a DTD to use it. There are many DTDs that are available as open source DTDs and schemas, available for you to use for free. To get an idea of just how many, surf the Applications section of the Cover Pages (xml.cover-pages.org) to see a rough list. Or, go to xml.schemas.org and see what's available there.

For technical writers, a couple of the noteworthy structures are the Darwin Information Typing Architecture (DITA) and DocBook.

DITA

DITA gives you the building blocks for creating your own topic-based markup and is described as follows:

The Darwin Information Typing Architecture (DITA) is an XML-based, end-to-end architecture for authoring, producing, and delivering technical information. This architecture consists of a set of design principles for creating "information-typed" modules at a topic level and for using that content in delivery modes such as online help and product support portals on the Web. [1]

DocBook

DocBook works out of the box and it comes in a couple of different forms (full and simplified) and with a suite of stylesheets. DocBook is defined as follows:

DocBook is a DTD maintained by the DocBook Technical Committee of OASIS. It is particularly well suited to books and papers about computer hardware and software (though it is by no means limited to these applications). [2]

It has to be "production worthy"

Beyond knowing what's available, you also need to consider what makes an XML implementation production worthy, regardless of the tool. Too often, companies focus on the content model defined in a DTD, or the availability of ready-made stylesheets, and forget about the environment in which the DTD will be used. Some of the factors that can affect XML implementation are described below.

Gaining Management Support

Exposing the XML

For publications, XML authoring begins in the authoring tool. A typical authoring scenario goes something like this: users sit at their computers, select File > New, and choose a type of document to create. The list of document types corresponds to the list of DTDs (Document Type Definitions) that the authoring tool knows about. (The details might vary, but this is the basic procedure for creating a new document in pretty much all XML editors.)

The operative word in this scenario is "users" because defining production-worthy always begins with users. Typical users are a little nervous about technology. They've heard about XML, but really don't understand it. And, in a good XML implementation, they shouldn't need to know much about it. That's the first rule of production worthy: hide as much of the XML from users (in this case, authors) as possible—it scares them! Users need to understand structure and structured authoring. They also need to understand the concepts of attributes and metadata. But they don't need to understand things like the coding rules of XML. That should be hidden. Expose only as much XML as possible.

Reducing the learning curve

Hiding the complexities of XML from users removes "XML Training" from the list of courses that people usually think are required for moving to XML. XML training is required, but not for all users. Hiding the XML is one way of reducing the learning curve. There's also a second way: provide users with a DTD where the element names are meaningful to them.

One of the great benefits of building a DTD is that you get to create the tag names. XML itself is not a markup language, but a standard for creating markup languages. If you are creating your DTD from scratch, you get to make up all of the tag names. That's an advantage, because you get to give your structural elements names that have meaning to you and your authors. That's the second way of reducing the learning curve. Your tag names will fit into the natural language of your authors, making your markup easy to use and author in.

On the other hand, buying a DTD brings the risk of imposing a new language on your authors. You call it a "caution", the other markup calls it an "alert." This can lead to confusion and inefficiency and will definitely add to the learning curve.

Supporting information exchange

One of the arguments people use when promoting industry-standard DTDs is that they come with the ability to improve information exchange. This can be a very persuasive argument to adopt an industry-standard or existing DTD. But, so what if you have to exchange information? You already do. You exchange it with users. Okay, I'm being obtuse; the point of information exchange is to share the source so it can be reused by others. But, you needn't approach it any differently than delivering information to users.

There are two phases of information development when using a markup language—the authoring phase and the delivery phase. Information exchange is just another delivery output. With XML, you have the opportunity to improve the efficiency of authoring and the efficiency of delivery. So why not do both? Make the authoring version as effortless to use as possible, and transform it (using XSL stylesheets) into as many output (delivery) languages as you need. You can take a custom (modified) DTD and transform it to a standard DTD in order to exchange information. Now that is not to say that an industry-standard DTD will never match the language/terminology of your authors. It might. If it does, you've got the best of both worlds: efficient authoring and easy source sharing.

The bottom line is, to be considered production-worthy, an XML implementation must be optimized for both authoring and for delivery.

Supporting multiple outputs

The buy vs. build decision can also extend to stylesheets. XML requires stylesheets for output. You associate XML with a file, pass it through an output generator, and get the appropriate output format (e.g., HTML, PDF, or other XML markup). Generating multiple formats requires multiple stylesheets and possibly multiple output generators. For authors or publishers, this is not necessarily a big deal. It's technically not really any more complicated than associating a Word document with a specific template.

However, the complexity is for the individuals creating the stylesheets. The more complex the output, the more complex the stylesheet. The more complex the list of outputs, the more complex the maintenance of those stylesheets will be. How many stylesheets might be effected by a style change? Can they be modularized to share common style properties?

20

Gaining Management Support

Here is where there are some advantages to the bought or borrowed DTDs. They usually come with stylesheets that are, at the very least, excellent starting points that you can modify for your own use. Some, like DocBook, come with a suite of stylesheets for things like HTML pages, Web Sites, HTML Help, and PDF. Building upon an existing stylesheet can really shorten the implementation process.

Ranking the choices

So how do you choose the best approach? The choices are:

- Build your own DTD from scratch
- Adopt an existing (possibly industry-standard) DTD as is
- Modify an existing DTD
- Create your own DTD as a layer on top of an existing/industry standard DTD

Build your own DTD from scratch

This is the most time-consuming approach, but it also has the most potential for getting exactly what you want from a DTD.

Pros

- You get exactly what you want
- Greatest opportunity to improve both the authoring and delivery sides of content
- Shortest learning curve
- Easiest to get buy-in for from authors

Cons

- Time consuming
- Technically demanding

Adopt an existing (possibly industry-standard) DTD as is

The fastest approach to implementation, but you risk limiting the effectiveness.

Pros

- Shortest to implement
- Facilitates source sharing

Cons

- Long learning curve where the language of the markup tags is not natural to the users
- Could be more tags or fewer tags than you really need
- Stylesheets are designed for someone else's styles, if they exist

Modify an existing DTD

Faster than starting from scratch, but may have issues for long-term maintenance.

Pros

- Takes less time than starting from scratch
- Can build on existing stylesheets
- Can add tags when tags are missing or delete tags when not needed

Cons

- Long learning curve if the language of the markup tags is not natural to the users
- Modifying can be technically demanding
- Can be difficult to maintain when the source DTD changes

Create your own DTD as a layer on top of an existing/industry standard DTD

This requires that you author in a tag set of your own making, then transform the markup into that of an existing DTD/standard.

Pros

- You can create a markup set that meets the users' needs exactly
- You can still take advantages of stylesheets supplied with the DTD
- You can optimize for both authoring and delivery

Cons

- Still requires technical expertise to modify the stylesheets to match your style

Summary

There are advantages and disadvantages to both start-ing a new DTD from scratch and using or modifying an existing DTD. When starting with an existing DTD, you can build upon the information models repre-sented by the DTD, as well as take advantage of stylesheets that frequently support the DTDs. The dis-advantage is that you have to adapt your authoring to someone else's vision of the content. Starting from scratch means that the content models match your work, but the effort to create everything—the DTD and all the stylesheets—may be prohibitive.

References

[1]   Description of DITA is available at http://www-106.ibm.com/developerworks/xml/library/x-dita1/

[2]   Description of DocBook is available at www.doc-book.org

Case Study

It's Not about Technology ... It's about Methodology: HP's Lessons Learned

Scott Abel and Lisa Woods

In this case study, Pat Waychoff, a single sourcing visionary and strategist for HP Network Storage Solutions, TCE Metrics and Initiatives department, describes the evolution from traditional documentation authoring and publishing to single source XML content management. Waychoff offers advice for others who hope to tackle such an initiative on their own. HP's lesson learned: "To be successful," Waychoff says, "you need to recognize that there is no 'software' solution. It's not really about technology. It's about methodology."

Background

First, here's some background on HP Network Storage Solutions and the challenges they were facing:

Target audience: End-users, customers, operators, system administrators, technicians, field service personnel, customer call center staff

Deliverables: User guides, instructions, information, notes, and help systems.

Delivery formats: Print, PDF, HTML, Java Help, HTML Help, WinHelp.

Languages: English, Japanese for all deliverables. English, Japanese, major European and Asian languages for some deliverables.

Challenge: More than 100 content creators in 12 geographic locations (North America, Europe, Far East) required the ability to collaborate on documentation projects including: content authoring, localization, translation, and delivery into multiple output formats in order to reduce costs, increase productivity, and improve quality by eliminating unnecessary manual tasks.

Revelations: The HP Network Storage Solutions Team identified two requirements that would hold the key to tackling this challenge: the need to adopt structured writing (to address issues of consistency, quality, content reuse and re-purposing) and the need for a move to an XML authoring environment.

HP's approach: What they did and why

Waychoff describes his foray into the world of content management as a "four-year journey" that would not have been possible without adopting a structured authoring methodology. "Structured writing is foundational," says Waychoff. "Without it, single sourcing would not work."

To get the project off the ground, Waychoff and his team relied on research from the Society for Technical Communication, attending conference presentations, reading white papers from industry luminaries, and mastering the concepts outlined in the book *Managing Enterprise Content: A Unified Content Strategy* (New Riders), which Waychoff and team view as their "bible for single sourcing". The team also turned to the book *Single Sourcing: Building Modular Documentation* (William Andrew Press) to learn how to author content in a modular fashion.

To succeed, they learned they needed an approach that would serve the needs of all their content creators, translation and localization staff ... and the clients they serve. They also needed tools that would help them to create more content with less money. "The bottom line," Waychoff says, "we had to reduce budget costs."

Convincing management Of course, taking on a content management project is more than just doing research and learning what you need. You also have to sell the idea to get the project funded. And this—convincing management of the need to fund a paradigm-shifting initiative—can be difficult, Waychoff says, especially if they aren't able to clearly perceive the necessity.

Waychoff's team was painfully aware of their need for change. They were struggling to create increasing amounts of documentation with decreasing budgets. "Most days," Waychoff says, "I felt like I was the Blue Light Special at K-Mart as far as budgeting went. Something had to give."

Case Study

To illustrate the need for structured XML authoring and content management to those who control the budget, Waychoff's team set out to gather meaningful cost savings metrics. "We had no real budget to buy any new tools (such as an XML authoring and content management system). We had to focus on using tools that didn't cost much," Waychoff says.

Proof-of-concept The initial effort involved restructuring and reworking a sub-set of content using Adobe FrameMaker (an inexpensive commercial software tool that facilitates structured authoring and single source content creation). This low-cost, proof-of-concept exercise yielded impressive results. The move to structured authoring alone (before adding a content management system to automate workflow) significantly improved content creator efficiency and produced a noticeable increase in quality. Both benefits were realized, Waychoff says, by "freeing writers and editors from the mechanical processes of information development." Authors and editors were suddenly able to focus on key processes, effective information design, and the development of new content.

The proof-of-concept also yielded impressive payoffs for word and page count metrics. Waychoff and his team were able to reduce the total source word count of a single deliverable from 33,200 to 27,500 - a decrease of 6,000 words (approximately 20%). Total production page reductions were even more impressive. One restructured document shrank from 2714 pages to 1908 pages - a decrease of 806 pages. These reductions were made possible by structuring the content for reuse and using the "conditional text" feature native to Adobe FrameMaker. Real-world savings would be significantly higher, Waychoff notes, as these metrics don't take into account additional downstream savings including anticipated shortened review times, reduced translation and localization fees, and lower production costs (printing, packaging, shipping, and storage).

Page count savings (reductions in the number of pages published) led to impressive financial paybacks. Before single sourcing, revising 1306 pages of content would cost HP about $59,000US. After single sourcing was introduced, this cost dropped to $22,000US—a savings of roughly $37,000US on content production and development costs alone.

Additional benefits

Determining additional savings While metrics collected during the proof-of-concept were valuable, Waychoff's team wasn't ready to stop there. They knew that to gain full advantage of the many benefits single sourcing and structured authoring can provide, they'd need funds to implement a content management system and a robust XML authoring tool. To get this funding, they'd have to provide potential cost savings estimates to management. And, they'd need to provide more than page count savings estimates: "We determined that measuring page count was not specific enough. We were not really measuring actual content, just the number of pages," Waychoff says.

The team used a cost savings calculator designed to factor in actual costs to generate potential savings estimates. The calculator worked with actual project data: number of writers, annual number of new content pages created, annual proportion of content in need of revision, number of languages required, translation cost, writing cost, conversion costs, labor costs, etc. The results? Substantial potential cost savings estimates. The estimated savings were so impressive that the team decided to err on the cautious side, and lowered the figures provided by the calculator before presenting a more conservative cost reduction estimate to management.

The proof-of-concept data combined with the results of the cost savings calculation "was enough to convince management to fund the purchase of a content management system and XML authoring software," Waychoff says.

Cost savings from structuring content Structuring content has resulted in a 5% to 44% reduction in total word count, as well as reductions in both content creation time and localization costs. "Today, we are using our own data (reduction in labor) to demonstrate that structured writing and XML are working for us," Waychoff says.

Cost savings from automation "We did a pretty thorough analysis of the tasks being performed, especially examining how much time we were doing manual tasks that could be automated [workflow, formatting, publishing]," says Waychoff. The content management system was recently installed and training is underway. Adding XML authoring and a content management system to the mix will result, Waychoff and his team foresee, in an additional 20% reduction in overall content

24

Case Study

development costs as a result of automating the labor-intensive manual formatting and publishing processes.

Outcome and lessons learned

Authors love it In the pilot project underway, Waychoff is hearing feedback from the participating authors that they prefer the new structured XML approach and that they "would not want to go back to the old way." The authors who were not involved with the pilot are pressing to get started as soon as possible. "There is excitement! [In the new paradigm] they really can spend most of their time on content creation," Waychoff says. "The tedious and time-eating tasks of formatting and publishing are automated. And reuse of existing content is also being made easier."

New role is required: Information Architect Waychoff and team convinced upper management to create and fund a new role dubbed Information Architect (IA). "You need someone to become almost an expert. They need to know your business and to make logical decisions that support your business model," says Waychoff. The information architect, in addition to performing tasks related to the structure of the content models, is also charged with helping to keep the initiative alive and with preparing the entire team for formal implementation.

Selecting tools requires analysis It was important that the new way of working didn't automate bad business processes and wasn't hostage to the limitations of the new tools. With this in mind, the IA and a writing management team of managers from four geographic business areas developed requirements for the authoring tool before the tool was selected. "We spent over a year analyzing tools available." In the end, the team selected Arbortext Epic as their XML Authoring tool. "We looked at everything. And while it certainly wasn't the cheapest tool, we determined it was the best investment for us."

After months of reviewing content management tools, the team selected Vasont as their content management system. Not only did Vasont meet their business requirements, but also another area of HP had been using Vasont for over three years. This fortuitously provided the department with an internal pool of advanced and expert users to help ensure the current project succeeded.

Training is key to success "Training in structured writing was most important," Waychoff says. "It set the foun-

dation for our success." Training is provided at each geographic site and all content creators and editors are trained in the paradigm shift and the accompanying tools used to author and edit modular content. Structured writing rules are captured guidelines which are captured in a living document that evolves as the business needs dictate.

Grass-roots buy-in is critical While management support is crucial to getting your project funded, Waychoff says he cannot overemphasize the importance of also obtaining grass-roots buy-in. "You have to show the individual writers and editors what's in it for them. They have to be convinced that this is in their best interests. Persistence and dedication are key [to conveying that message]." Waychoff says.

Advice for others: Underpromise and overdeliver "Keep your cost savings estimates very conservative," Waychoff says. "Whatever you say, someone (especially in upper management) will remember and hold you to it."

25

What's in the News

In the news: Information Architecture

You may not realize it, but information architecture is about far more than the web. In this issue of *The Rockley Report*, we provide you with several resources you may find valuable in your quest to learn more about the discipline of information architecture and how it relates to the management of all types of content, regardless of delivery medium. Check out the online resources, read a book or two, or attend a conference or online event. Whatever you do, learn as much as you can about information architecture—it's an important aspect of content management and should not be overlooked.

What is Information Architecture? What do Information Architects do?

"Today, most information isn't presented in the detailed form of a map to direct and guide us to new lands where we can find a wealth (or a wealth of information). Rather, it's fired at us like buckshot, with the hope that some might hit a target. In response, a group of people is emerging that feels like a life force-an undeniable drive-to make life understandable. They feel compelled to create a new world map with this barrage of data. They are information architects." (Richard Saul Wurman, *Information Anxiety 2*, 2001, Que Publishing, Indianapolis)

Richard Saul Wurman coined the term "information architecture" back in 1975. In *Information Anxiety 2*, Wurman writes: "When I came up with the concept and the name information architecture in 1975, I thought everybody would join in and call themselves information architects. But nobody did - until now. Suddenly it's become a ubiquitous term. Of course, as is the case with any ubiquitous label, there are some information architects who legitimately meet the definition of the term, but there are lots who don't."

"Today's information architects," Wurman writes, "must get through to a population that makes choices every day about what to view and what not to view. People are bombarded from all sides by television, print, and online ads. We have newspapers, magazines, journals, newsletters, e-zines, and online news sites where we can tailor our daily dose of news to fit personal interests. Individuals who cross these boundaries (information technology professionals, graphic designers, writers, journalists) have the potential to make good information architects."

So what is an information architect? According to a 1996 article by Wurman published in *Information Architecture*, an information architect is "the individual who organizes the patterns in data, making the complex clear."

While that sounds simple enough, there is still a lack of clarity surrounding the discipline of information architecture and role of the information architects. We've prepared a list of online resources and books that might help you better understand the world of information architecture.

Online Information Architecture Resources:

The InfoDesign Interview: Richard Saul Wurman, January 2004 - an interview with the father of information architecture
http://informationdesign.org/special/wurman_interview.htm

What's in a name? Information Architecture versus Information Design - various views on information architecture from information architecture luminaries
http://www.stcsig.org/id/dmatters/apr01.pdf

Defining Information Architecture Deliverables by Christina Wodke
http://www.sitepoint.com/article.php/326?

Asilomar Institute for Information Architecture (AIFIA)
http://www.aifia.org/

Introduction to Information Architecture (AIFIA Library)
http://aifia.org/library/

Books About Information Architecture:

Information Anxiety 2 -- by Richard Saul Wurman (Que Publishing)
http://www.quepublishing.com/title/0789724103

What's in the News

Introduction to Information Architecture for the World Wide Web -- a free chapter from the book by Louis Rosenfeld and Peter Morville (O'Reilly Publishers)
http://www.amazon.com/exec/obidos/tg/detail/-/0596000359/102-3298120-0863349?v=glance

*Dynamics in Document Design: Creating Text for Reader*s by Karen A. Schriver (Wiley Publishing)
http://www.wiley.com/WileyCDA/WileyTitle/productCd-0471306363.html

Information Architecture With XML by Peter Brown (Wiley Publishing)
http://www.wiley.com/WileyCDA/WileyTitle/productCd-0471486795.html

Information Architecture Events

Sixth Annual International Workshop on Internationalisation of Products and Systems 2004 - July 8-10, 2004, Vancouver, British Columbia, Canada
http://www.iwips2004.org

WebVisions 2004 - July 16, 2004, Portland, Oregon
http://www.webvisionsevent.com/schedule/?PHPSESSID=d2cc12fd33c29067705d631054e2f0ac

Information Architecture for Content Management QuickStart - August 7, 2004, Tampa, Florida

2005 Information Architecture Summit - March 4-7, 2005, Montreal, Quebec
http://www.asis.org/

Contributors

Scott Abel

Scott Abel is a freelance technical writing specialist and content management strategist whose strengths lie in helping organizations improve the way they author, maintain, publish and archive their information assets.

Cori Czekaj

Cori Czekaj is a Solution Architect, Data Modeler, and Project Manager for Arbortext, Inc. Cori has over six years of experience architecting, implementing, and managing XML-based, multichannel e-business and technical publishing solutions. With doctoral degrees in Chemistry and Materials Engineering, Cori is the primary ATI consulting contact for the MathFlow implementation with Design Science. She is a member of HL7 and the American Chemical Society.

Pamela Kostur

Pamela Kostur is a Principal with The Rockley Group, specializing in information analysis, information modeling, and structured writing to support a unified content strategy. Pamela has over 18 years experience developing information solutions. During that time Pamela has completed many projects and presented papers at numerous conferences on topics including iterative usability, miscommunication, structured writing, editorial "magic", building and managing intranets, creating usable online documentation, unified strategies for web-based learning, information modeling and analysis. Pamela is a co-author of *Managing Enterprise Content: A Unified Content Strategy* with Ann Rockley and Steve Manning.

Steve Manning

Steve Manning is a Principal with The Rockley Group and has over 16 years experience in the documentation field. He is a skilled developer of online documentation (WinHelp, HTML Help, Web sites, XML, and Lotus Notes) and has created single source production methodologies using key online tools. Steve has extensive experience in project management and has managed a number of multiple media, single source projects. Steve teaches "Enterprise Content Management" at the University of Toronto, and is a frequent speaker at conferences (ASIS, AUGI, STC, ACM SIGDOC, DIA) on the subject of XML and Content Management. Steve is a co-author of *Managing Enterprise Content: A Unified Content Strategy* with Ann Rockley and Pamela Kostur.

Ann Rockley

Ann Rockley is President of The Rockley Group, established to assist organizations in adopting content management, unified content strategies, and information architecture for content management. Ann has been instrumental in establishing the field in online documentation, single sourcing (content reuse), enterprise content management, and information architecture of content management. She is a frequent contributor to trade and industry publications and a featured speaker at numerous conferences in North America and Europe. Ann is the author of *Managing Enterprise Content: A Unified Content Strategy* with TRG Senior Consultants Pamela Kostur and Steve Manning.

Lou Rosenfeld

Lou Rosenfeld is an independent information architecture consultant. He has been instrumental in helping establish the field of information architecture, and in articulating the role and value of librarianship within the field. Lou co-authored the widely respected "polar bear" book and, as founder and president of Argus Associates, he helped build one of the world's most admired information architecture firms. Lou is also co-founder and board member of the Asilomar Institute for Information Architecture.

Lisa Woods

Lisa Woods is a senior technical writing consultant with Keane, Inc. She specializes in helping clients find the intersection between documentation assets and the audiences they serve.

Call for Submissions

The Rockley Report publishes original material related to content management, including its goals, its implementation, the technology required to support it, and its affect on organizations. If you're interested in submitting to *The Rockley Report*, we'd like to hear from you. Please send us your ideas for articles in the following categories:

- Best Practices — Articles in this category describe content management in the "ideal" world and suggest how to put those ideals into practice in the "real" world. Best practices focuses strategies, activities, or approaches that have been shown through research and evaluation to be effective.
- Information Architecture — Articles in this category explore the relationship between information architecture and content management, including topics such as building a blueprint for a content management strategy and content modeling.
- Tools and Technology — Articles in this category investigate the technology required to support content management.
- People, Processes, and Change — Articles in this category discuss management issues related to content management, such as changing roles and writing in a content management environment.
- Gaining Management Support — Articles in this category provide strategies for helping management understand the benefits of content management, focusing on topics such as building a business case for content management and calculating ROI.
- Case Studies — Case studies explore how companies are implementing content management and focus on what they did and why, their benefits, and their lessons learned.

If you have an story you'd like to submit, please write a 250–word description of your topic, the category you think it best fits, then send it, along with a 100–word bio, to Pamela Kostur at kostur@rockley.com.

Next Issue

The next issue of *The Rockley Report* focuses on Tools and Technology and our featured interview is with Tony Byrne, the founder and editor of CMSWatch.com. Byrne is the President of CMSWorks, Inc., a USA-based content management consulting and training firm. A former reporter, publisher, international educator, and 14-year Internet veteran, Byrne formerly led the Engineering group at IDEV, a Web integration and solutions firm. He also serves as principal author of The CMS Report. As a consultant, he helps organizations identify and select appropriate CMS technologies and as such, is an ideal candidate to be interviewed for our Tools and Technology issue. Other highlights include:

- XBRL: What's it all about? (An interview with Blast Radius)
- How to build a business case for the purchase of new tools
- Information architecture to ease the authoring burden
- A case study in developing a content management system to meet the needs of community and non-profit organizations

The next issue will be available to subscribers in late September.

Subscription Information

For US and international subscriptions

Subscriptions are $99 a year (four issues) or $30 for a single issue, payable in US funds. To subscribe, go to http://www.rockleyreport.com/index.php/subscriptions/US_International/

For Canadian subscriptions

Subscriptions are $125 a year (four issues) or $40 for a single issue, payable in Canadian funds. Please add 7% GST. To subscribe, go to http://www.rockleyreport.com/index.php/subscriptions/canadian/

Payment can be made via Pay Pal, check or money order.

Call Us!

We'd love to hear from you. What do you think of the *Rockley Report*? What would you like to see in the future?

If you have any questions, comments or suggestions, please feel free to let us know. The easiest way to reach us is via email. Our Editor, Pamela Kostur, can be reached at kostur@rockley.com.

Visit our corporate website at www.rockley.com, or the website for our book, *Managing Enterprise Content: A Unified Content Strategy* at www.managingenterprisecontent.com.

We hope you enjoyed this issue, and hope to hear from you soon.