Project Report

News you can reuse

Content repurposing at The Wall Street Journal Interactive Edition

Alan Karben
Associate Director, Interactive Development
The Wall Street Journal Interactive Edition

TEL +1 (212) 416-2975
FAX +1 (212) 416-3291
EMAIL karben@wsj.com
WEB http://wsj.com

The content-reuse system of *The Wall Street Journal Interactive Edition* makes extensive use of SGML and XML to reorganize and reformat the content presented in the main wsj.com website. This paper discusses how the structures that define an *Interactive Journal* edition and its component articles are queried, processed, and converted by automatically triggered content-processors, allowing us to quickly fill requests by potential publishing partners to feature our branded content in their contexts.

Introduction

The Wall Street Journal Interactive Edition was launched in April 1996, with the mission to provide the richest business and financial news web site in the world. The focus of its systems developers was to supply the tools and infrastructure to create and maintain that site.

Back then, it was hardly anticipated that news from the *Interactive Journal* would appear not just at wsj.com, but also on a number of affiliated web sites, screen savers, handheld devices, and even in-flight infotainment systems. But demand from other companies for the news selected and crafted by *Interactive Journal* editors has been steadily increasing throughout our two-year existence. Consequently, the systems we use to automatically repurpose our content for various other domains and platforms have had to increase in functionality.

An early decision to go with SGML as the strategy for structuring our information has made these system enhancements relatively clean and straightforward. This paper will discuss how the structures that define an *Interactive Journal* edition and its component articles are queried, processed, and converted by automatically triggered content-processors, allowing us to quickly fill requests by potential publishing partners to feature our branded content in their contexts.

Markup Languages: Theory & Practice 1.1 (1999): 33–45

Navigation through the *Interactive Journal*

The heart of the *Interactive Journal* is organized hierarchically. Subscribers see a list of sections presented on the left side of most pages. Our first three sections correspond roughly to the three regular sections of the paper edition: Front, Marketplace, and Money & Investing. We have added sections for Technology and for Sports, and also have a customizable section called Personal Journal.

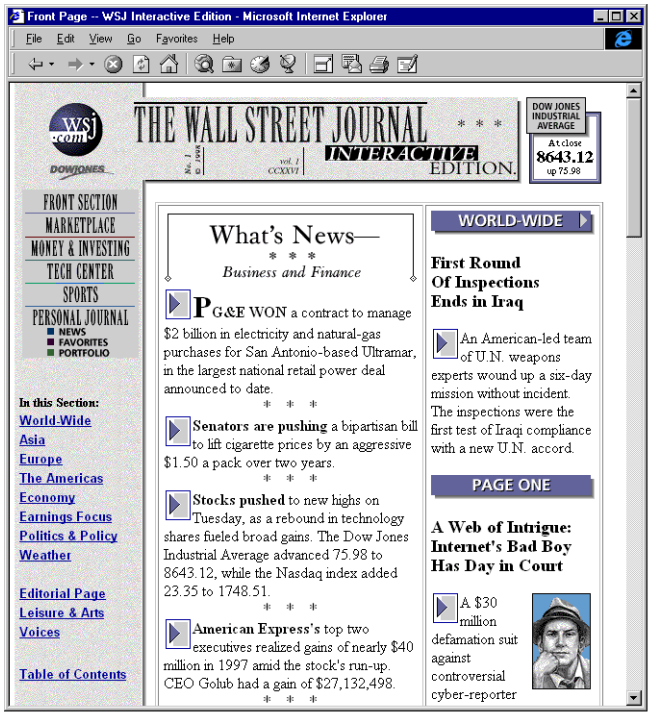

Figure 1 A sample wsj.com front page

Within each section is a set of pages, with the current section's set of pages presented beneath the list of sections. Examples of pages in the Front Section include World-Wide, Asia, Economy, and the Editorial Page.

Finally, on any given page itself, the subscriber is presented with a series of summaries for the articles grouped on that page. If a subscriber wants to read beyond the summary, he or she clicks on an icon positioned alongside it, and is taken to the body of the article itself.

Demands for reusable content

Our content reuse requests have generally involved supplying specially formatted versions of the articles and summary pages that regularly run in the edition. For example, a dynamically updating screen saver featured the first four summaries from the front pages of the Front, Marketplace, and Sports sections, as well as popular columns like Heard on the Street and Personal Technology.

Before building the new system described in this paper, each content reuse project would take a developer several weeks to program, test, and roll into production. Even minor requests for changes for how that content was selected or presented would take a number of days. Also, once a process for selecting and publishing content had been automated, a human editor would have no opportunity to step in and edit that news, perhaps to make it more appropriate or relevant for its new medium.

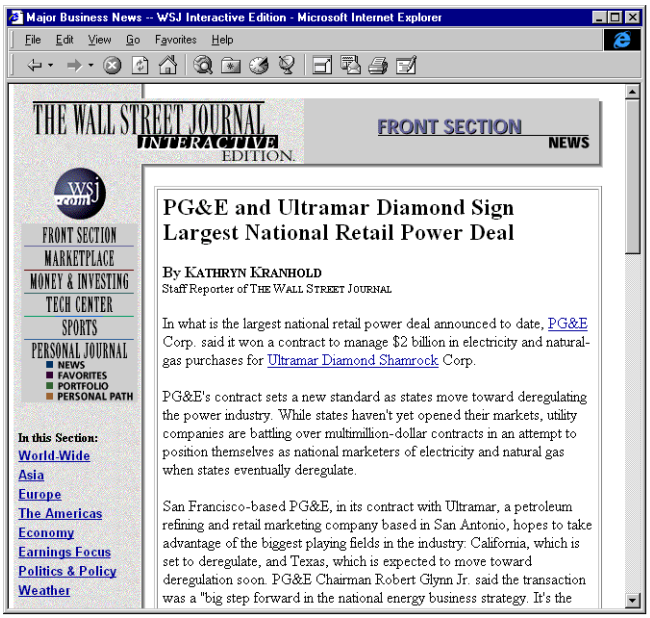

Figure 2 A sample article

The new system has been designed to make custom coding unnecessary during the content-selection process for each project. Instead, an XML-based configuration file is drawn up, which specifies which articles and pages should be reused.

Also, the new system provides an editorial interface whereby an editor can rearrange and make changes to one of these derived editions. The editor is then able to lock down the changes, so that the next automatic publish that takes place does not overwrite his or her changes.

Pre-publish processes

The *Interactive Journal* sends updated articles out to the public servers at irregular intervals, ranging from every ten minutes, to once every hour or so. Before each of these publishing processes, editors work with multiple applications to find, write, edit, and order the news.

Creating an *Interactive Journal* news article

The SGML/XML-based content generation process of the *Interactive Journal* starts with copy originating from *The Wall Street Journal's* copyflow system, a newswire application, or a reporter's notebook. An editor copies and pastes, or types, this copy into Microsoft Word, and uses a set of custom-designed Word macros to apply the paragraph and character styles that describe the structure of the article and its corresponding summary.

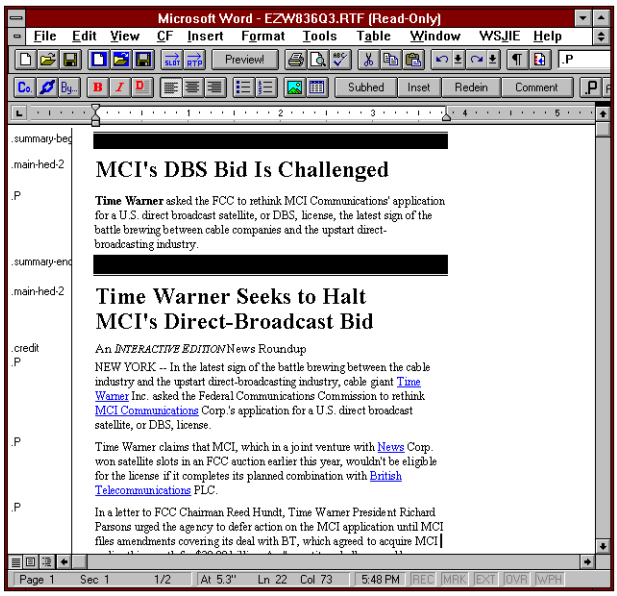

Figure 3 An article, as seen by an editor using Microsoft Word

He or she also fills out forms that embed into each what the article as a whole is about, and why its various textual components are worth marking up. The Document Attributes dialog box allows an editor to assign to an article one of more than 300 Article Types, ranging from regularly running columns such as Manager's Journal and Work & Family, to broader characterizations such as Politics & Policy and Asian Technology. The editor is also able to select what Industry Categories an article covers, and whether it is a major, normal, or minor news item.

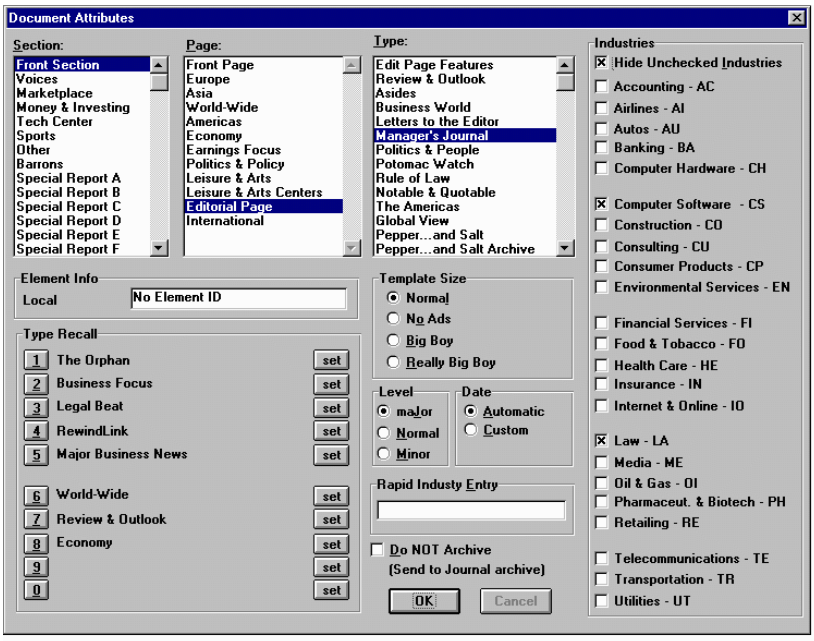

Figure 4 The Document Attributes dialog box in Microsoft Word

When applying links and enrichment to the body of the story, the emphasis is again on structure. A company name is marked-up as just that, with the ticker symbol and significance rating filled in as attributes. Links are categorized by the type of destination, ranging from an article from our own archives, to a CNBC-Dow Jones Business Video clip, to an external web site unaffiliated with the Journal.

Once created in Word, the article is routed up the editorial ladder until it is finally approved for placement into the edition. At this point, an editor uses a custom-built Edition Maintenance application to drag the article from a Ready-To-Place area on the screen into the edition itself.

This action triggers a process that converts the article from Word's ASCII-based Rich Text Format into a document conforming to the SGML Document Type Definition of the *Interactive Journal*, DJML (the Dow Jones Markup Language, named for our parent company). During this conversion, an Article Metadata Log File is updated to hold the ID, Type, Headline, Company Mentions, and other metadata relevant to the article that was just processed.

Updating the edition

The Edition Maintenance application (developed using C++ and an Object-Relational Database on the back-end, and Visual Basic for the front-end) offers a file/folder-like hierarchical view of an abstract edition structure. The root Edition node of this tree contains Section nodes, which contain Pages, which contain Columns, which hold Articles. An editor decides which page or pages an article belongs, and positions it on a column and at an appropriate rank, which usually reflects the article's newsworthiness.

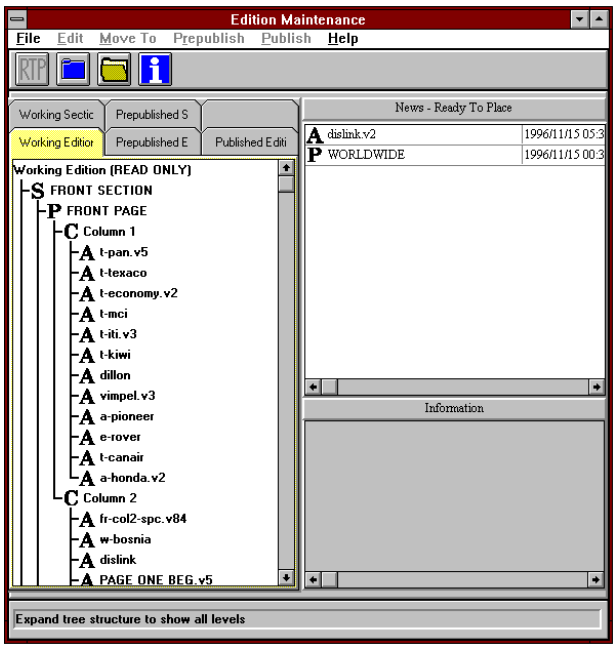

Figure 5 The Edition Maintenance application

When the editor decides that a particular arrangement of the edition is presentable to subscribers, he or she "prepublishes" the edition. This process generates HTML documents based on the DJML source, and places the HTML

summary pages and articles onto a staging area that the editor may then view and proof.

Satisfied with these HTML files, the editor then "publishes" the edition, triggering the migration of files from the staging area to six mirrored web servers for our subscribers.

Post-publish processes

Creating the NCD

After every publish process, several steps happen to generate and update the content we reuse in other products besides the main edition.

```
<!DOCTYPE ARTICLE-DOC PUBLIC "-//Dow Jones//DTD DJML//EN">
<ARTICLE-DOC
       ID="SB889744717195520000.djml" TYPE="Marketplace" LEVEL="NORMAL"
       PUBLICATION="The Wall Street Journal Interactive Edition"
       DATE="1998-03-13 00:11" COPYRIGHT="Dow Jones &amp; Company, Inc.">
<SUMMARY>
       <HEADLINE>
       <MAIN-HED>Kodak Wins Back Sales<BREAK>
       With New Film Design</MAIN-HED>
       </HEADLINE>
       <P><HIGHLIGHT TYPE="BOLD">The new design</HIGHLIGHT> for Eastman Kodak's
       E200 film is suddenly helping the company win back sales to professional
       photographers in its long-running battle with Fuji.</P>
</SUMMARY>
<ARTICLE>
<HEADLINE>
       <MAIN-HED>Kodak Looks for Another Moment<BREAK>
       With Film for Pro Photographers</MAIN-HED>
</HEADLINE>
<BYLINE TYPE="SIGNED">
       By <PHRASE TYPE="AUTHOR">Laura Johannes</PHRASE>
       <CREDIT>Staff Reporter of
       <PHRASE TYPE="TITLE">The Wall Street Journal</PHRASE>
       </CREDIT>
</BYLINE>
<P>It was a sports photographer's nightmare: a poorly lit rink, no flashes
allowed, and Tara Lipinski spinning like a tornado. But Dave Black got a
shot of the Olympic skater that captured individual sequins on her bodysuit
and strands of hair in her bangs. Newsweek devoted a full page to it.</P>
<P>Mr. Black, a free-lancer, credits the clarity to <PHRASE TYPE="COMPANY"
NAME="ek" SIGNIFICANCE="PROMINENT">Eastman Kodak</PHRASE> Co.'s E200 film,
whose new design is suddenly helping Kodak win back sales to professional
photographers in its long-running battle with <PHRASE TYPE="COMPANY"
NAME="fujiy" SIGNIFICANCE="PASSING-MENTION">Fuji Photo Film Co. of
Japan</PHRASE>. The product is also shaping up as a central weapon in
Kodak's effort to make amends with professionals, an influential customer
group it had alienated.</P>
<INSET style="BOX-LEFT">
       <P><LINK TYPE="ARCHIVE" target="SB888853676602545500.djm">
       Despite Nagano Exposure</LINK>, Kodak Can't Win the Gold (March 2)</P>
       <P><LINK TYPE="ARCHIVE" target="SB886545977595328500.djm">
       Kodak, U.S. Government</LINK> Team Up for Drive on Japan's Film Market
       (Feb. 4)</P>
</INSET>
<P>"Kodak had just dropped the ball in the last eight or 10 years," says
Rick Rickman, another free-lancer who shot the Nagano Games with E200 film.
"Now, a lot of us feel Kodak's back as a player again." Kodak designed the
E200 film specifically for use in low light on fast-moving subjects --
"fast" conditions that plague sports photographers.</P>
<SUBHED>Powerful Group</SUBHED>

...

</ARTICLE>
</ARTICLE-DOC>
```

Figure 6 A sample, and somewhat simplified, DJML article

First, the entire tree structure stored in the Edition Maintenance application's underlying relational database, which includes Section and Page properties as well as ordered Article references, is exported as a text file. Next, a separate application reads in this file, as well as the Article Metadata Log File, and combines them both to form a single XML file called the Navigational Container Document, or NCD.

The NCD provides a convenient base format from which to derive interactive Tables of Contents and Indices to Businesses, and also serves as the reference document for our Content Reuse applications.

```
<?XML version="1.0"?>
<!DOCTYPE EDITION PUBLIC "-//Dow Jones//DTD DJML-NCD//EN">

<EDITION DATE="1998-07-05" TIME="17:34:54" ID="e-000">

 <SECTION TYPE="Front Section" SLUG="FRONT SECTION">

  <PAGE TYPE="Front Page" URL="front.htm" SLUG="FRONT PAGE">

   <COLUMN>

       <ARTICLE ID="SB899382193513151000" TYPE="Economy" SLUG="t-economy.v3"
                REV-ID="SB899404798911322500" ARCHIVE-STORING="ARCHIVE"
                LEVEL="NORMAL" DATE="1998-07-02 14:43">
             <MAIN-HED>Unemployment Rate Rises to 4.5%,<break/>
             But Economy Keeps Adding Jobs</MAIN-HED>
             <COMPANY SYMBOL="gm" SIGNIFICANCE="PASSING-MENTION"
             NAME="General Motors"/>
       </ARTICLE>

       <ARTICLE ID="SB899391647443511500" TYPE="Tech Center Main"
                SLUG="tc-cellular.v2" REV-ID="SB899416071348726500"
                ARCHIVE-STORING="ARCHIVE" LEVEL="NORMAL"
                DATE="1998-07-02 18:08">
             <MAIN-HED>New York Investment Firm to Buy<break/>
             Centennial Cellular for $1.5 Billion</MAIN-HED>
             <COMPANY SYMBOL="cycl" SIGNIFICANCE="PROMINENT"
             NAME="Centennial Cellular"/>
             <COMPANY SYMBOL="ctya" SIGNIFICANCE="PROMINENT"
             NAME="Century Communications"/>
             <INDUSTRY SYMBOL="DTE"/>
       </ARTICLE>

       …

   </COLUMN>

   …

  </PAGE>

  …

 </SECTION>

 …

</EDITION>
```

Figure 7 A sample NCD

Analyzing the Content Reuse Configuration Files

For every Content Reuse project we set up, whether for a specialized "push technology" application such as Pointcast or a mini-edition offered to users of handheld devices such as the PalmPilot, an individual Content Reuse Configu-

ration File must be set up. This Configuration File, maintained in XML, essentially specifies a customized mechanism for querying the NCD.

Several criteria are offered to the News Desk and Business Development teams for setting up the process that automatically selects articles for regrouping onto new pages, and repurposing into new formats. These criteria hinge upon which Article Sets are defined, and which destination Pages and Columns they are place in.

An Article Set is the container used to describe particular qualifications for articles to be taken from the main edition, and come in two flavors, Position-oriented and Type-oriented. A Position-oriented Article Set selects its articles by specifying the page and column from which articles should be grabbed. The Page ID and Column Number are specified as attributes, as is the rank of the first article in that column that should be examined, which is considered the Article Set's Start Position.

```
<?XML version="1.0"?>
<!DOCTYPE tedbot PUBLIC "-//Dow Jones//DTD DJML-TEDBOT//EN">

<tedbot edition="e-001" action="set-up-and-publish">
      <page ID="page-one-summaries">
            <column>
                  <article-set
                        source-edition="e-000"
                        source-page="front.htm"
                        source-column="1"

                        start-position="1"
                        finish-condition="total"
                        finish-value="6"
                  >
                        <article action="exclude" type="Technology">
                        <article action="exclude" type="Asia Technology">
                        <article action="exclude" type="Europe Technology">
                  </article-set>
                  <article-set
                        source-edition="e-000"
                        finish-condition="none"
                  >
                        <article action="include" type="Asia Stocks">
                        <article action="include" type="Europe Stocks">
                        <article action="include" type="Americas Stocks">
                  </article-set>
            </column>
      </page>
</tedbot>
```

Figure 8 A sample Content Reuse Configuration File

A finish-condition is also specified, which indicates how this Article Set's requirements should be fulfilled. A finish-condition of "total" indicates that some total number (listed in the finish-value attribute) of articles should be grabbed. A finish-condition of "position" indicates that articles should be grabbed up to and including a ranking position (also listed in the finish-value) in that column. Finally, a finish-condition of "none" indicates that all articles below the start position should be considered candidates for reusing.

Within a Position-oriented Article Set container, several exception conditions may be present. These exceptions can indicate that stories of a certain Article Type be excluded from the selected articles, and can also indicate whether the selected articles must also have been flagged for indexing by the *Interactive Journal* text-search engine (another attribute found on articles in the NCD).

In the sample Configuration File provided, the first Article Set is seeking a total of six articles, starting from the first article in Column 1 of the Front Page. A total of 6 articles should be selected from that column, none of which may have the Article Types of "Technology", "Asia Technology", or "Europe Technology".

Within a Type-oriented Article Set, the Page ID and Column Number attributes have no meaning, and neither does the Start Position. Instead, all stories throughout the entire NCD of the Article Types listed within the Article Set container are considered for selection. The total number of articles that wind up being selected is dependent upon whether the Article Set has a finish-condition of "none" (select all articles of these Types) or of "total" (select up to the number of articles specified in the finish-value attribute).

In the sample Configuration File provided, the second Article Set is seeking any and all stories present in the NCD of the following Article Types: "Asia Stocks", "Europe Stocks", and "Americas Stocks".

Building the derived editions

The Content Reuse Configuration Files as well as the NCD are then jointly analyzed to resolve the Configuration Files into actual mini-NCDs, complete with references to the precise articles that should be selected for each derived edition. The SGML/XML-savvy programming language developed by OmniMark Technologies (www.omnimark.com) was used to develop this functionality.

Special attention was paid in the implementation of this procedure to make performance as efficient as possible. For example, an easy way to program this functionality would have been to run through the entire NCD for every Article Set specified in the Configuration Files. However, our average NCD holds more than 800 articles at any given time, and this method would have been too slow.

Instead, after the system analyzes all Content Reuse Configuration Files, the NCD is processed and examined just once for articles that may fit the designated criteria. The (far smaller) Configuration Files are then processed a second time, and articles previously recognized as candidates for inclusion in a mini-edition are either selected or thrown away.

Now that the IDs and positions of the desired articles have been established, our system creates what we call a Distributed NCD. Like the base NCD, the Distributed NCD represents an edition hierarchy. But unlike its sister document,

which is one large document, the Distributed NCD is comprised of dozens of individual files. All Edition, Section, Page, and Article nodes are described by tiny, individual XML "property files," which are linked together by entity references.

The derived editions are specified via Distributed NCDs rather than self-contained NCDs in order to increase system performance when human editors decide to add value to any of our repurposed mini-editions. For example, if an editor decides to flip the order of the first two stories on a particular page, no lengthy tree-climbing is necessary. The system only needs to rearrange the entity references in that page's Property File.

Hand-crafting a derived edition

What tool does an editor use to do such rearranging? Another application, which on the surface appears much like Edition Maintenance, presents editors with file/folder-like listings of our various mini-editions. However, instead of being coded in C++ and Visual Basic, this application uses OmniMark-driven CGIs for the back-end, and Microsoft Internet Explorer 4.0 for the front-end.

```
<edition id="e-001" slug="pointcast" rtp="r-001" editor="alan"
              pub-time="19980624144113" pub-all-time="19980424100153"
              tedbot-routine="every-publish">
       &s-front;
       &s-money;
       &s-technology;
</edition>
```

```
<section id="s-front" slug="front section" type="front section">
       &p-page-one-summaries;
       &p-front-truncated;
</section>
```

```
<page id="p-page-one-summaries" slug="Page-page-one-summary"
              type="Law" chg-time="980630135058">
       <column id="c-1">
              &a-sb889745661659619500;
              &a-sb889741316190819000;
       </column>
       <column id="c-2">
              &a-sb889741453659327000;
       </column>
       <column id="c-3">
              &a-sb889895310453482500;
              &a-sb889858144327279000;
              &a-sb889754784146007500;
       </column>
</page>
```

```
<article id="a-sb889745661659619500" rev-id="sb889745661659619500"
              slug="guerrilla-ldrl" type="Leader">
       <parents>
              <object type="page" id="p-page-one-summary">
       </parents>
</article>
```

Figure 9 Sample property files

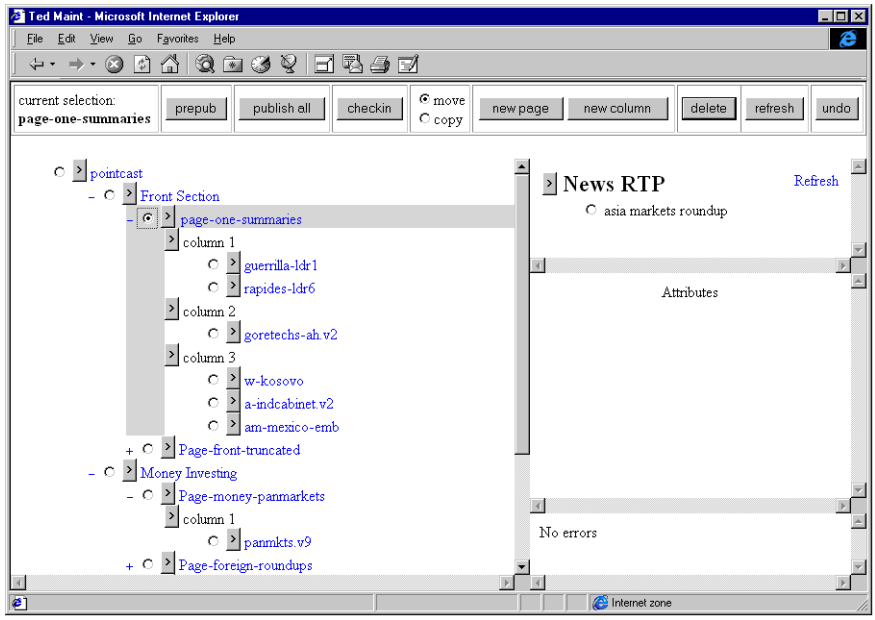

Figure 10 A snapshot of the Content Reuse edition-management interface

Two capabilities of IE4 made it an appealing platform for development. It's Dynamic HTML actually allows you to invent new attributes (just like XML), and use a scripting interface to query and react to their values. Furthermore, Dynamic HTML as implemented in IE4 allows portions of the content of a page to be rewritten on-the-fly, based upon user-triggered actions. Therefore, when an editor indicates that the order of two articles should be flipped, a CGI can be called to update the back-end Property Files, and that change can be reflected within his or her browser window without any annoying page redraws.

Every Article Property File contains two attributes that an editor can alter, to insure that his or her work is not overwritten by any upcoming automatically spawned publishing processes. The content-lock attribute indicates that automated systems should not overwrite the given article with any future revisions of that article. The position-lock attribute indicates that automated systems should not alter the position of that article. These independent flags allow editors the flexibility to lock the content and position of an article in place, or to keep one article at a certain spot on a page, but allow the system to keep its contents in step with what is being reported on that story in our main web site.

Conclusion

All of our content-reuse processes owe their flexibility and ease of implementation to our use of SGML and XML. Articles created in SGML have been translated and served out in all sorts of flavors of HTML and other plain text formats. Edition structures and configuration files specified in XML are processed and tailored by custom software that allows our editors to specify what constitutes a mini-edition. And when our automatically generated content falls short of serving their audiences completely, an editor can step in and finish the job.

It is interesting to note that software packages are now becoming available that implement standards-based XML querying mechanisms such as Extensible Stylesheet Language and the Document Object Model (www.w3.org/Style/XSL/ and www.w3.org/DOM). Where will this leave our OmniMark-crafted solution for selecting articles from the main edition?

We believe that our editors and news production staff will always be more comfortable with using a task-specific XML configuration file or GUI to specify which articles and pages our content-reuse processes should monitor, as opposed to being forced to master a more generic (and therefore more complex) query syntax. However, this does not mean that a standards-based back-end couldn't be used for the actual query-resolution process. We will have to wait and see whether these software packages evolve such that they can be configured to perform at the speed of our optimized processes.

But system performance, though important, has never been the overriding factor for driving our implementation decisions. After all, back-ends that use binary formats and compiled code can be optimized far more thoroughly than our ASCII property files and 4GL programming.

Instead, it has been the flexibility of our authoring and content-reuse systems that really make our content shine. Our editors and designers are charged with constantly improving how our news can be accessed, navigated through, presented, and used. And our business-development staff is constantly seeking new ways to raise the visibility of our brand, which often means spreading excerpts from our trove of content out to places and platforms that our primary web site would not otherwise reach. Having our news, and the processes that direct where that news belongs, in an extensible format has proved to be the key to fulfilling their requirements.

Received 7 July 1998
Revised 12 August 1998