End-User Customized Chemistry Journal Articles

Steven M. Bachrach,* Anatoli Krassavine, and Darin C. Burleigh

Department of Chemistry and Biochemistry, Northern Illinois University, DeKalb, Illinois 60115

Received May 1, 1998

A system for delivering journal articles customized by the reader is described. The articles are marked-up with custom tags to identify portions of the document that can be modified to meet an individual's specification. These tags are interpreted by the Web server and converted into HTML that can be read by a Web browser. This system has been implemented within the *Internet Journal of Chemistry*. The relationship of this process with chemical markup language is detailed.

INTRODUCTION

With the advent of the worldwide Web, chemists are offered an opportunity to radically alter the manner in which they communicate their results. This change goes far beyond delivery of text by electronic means rather than on paper. Electronic delivery in and of itself can certainly be advantageous, leading potentially to more rapid delivery and cheaper publication costs. We and others have previously argued that the Web opens the door to communicate information that either cannot be transmitted at all or not in a cost-effective manner on paper.[1−8] We have termed this "chemistry-enhanced publication".

There is a wide variety of content that constitutes a chemistry-enhanced publication. Some examples are rather obvious. Color images are rare in current journals due to the cost of production, but there is essentially no difference in the costs of preparation or delivery of a color image versus a black-and-white image using electronic tools. Publication of animation in a conventional journal is restricted to perhaps a few snapshots at specific time points, while an entire animation can be published online, available for the reader to view at full speed or stop-action. Interactive tools which allow the reader to manipulate data, such as three-dimensional models of molecules and spectra, can be incorporated into an electronic publication but cannot be printed onto paper.

A few chemistry journals are beginning to adopt some of these enhanced publication features.[9] We felt, however, that this emphasis on the advantages of electronic publication for authors has neglected the other component of a successful publication—the reader. In this article, we describe the concept of end-user customization of the chemical literature and how this approach is implemented within the framework of a new electronic journal, the *Internet Journal of Chemistry* (IJC).[10]

CUSTOMIZED JOURNAL ARTICLES

The worldwide Web oftentimes presents a facade. Beginning Web surfers and Web authors often view the Web as a collection of servers delivering documents resident upon the servers' disk drives. These documents may appear to have been authored and stored and remain fixed and unchanging, waiting for someone to request them. In fact, on many Web sites, this is far from the truth. Web pages do not reside as fixed documents but rather are created upon request, dynamic objects that come together only when some individual comes calling.

Dynamic Web pages are created typically from a database with selection criteria provided by user input. An example might be a page that offers the closing market stock prices. The individual can select the particular stocks of interest. The server then delivers the information via a page that is created by retrieving the final quotes, the number of shares traded, and the price range for the day and plugging them into a template. Most online catalogs also work in this fashion.

Another example of the process of creating a dynamic Web page requires passive activity from the user. For example, a service might have features that take advantage of a specific browser. The identity of the browser software (along with additional identification materials) is always passed by the browser to the server with the URL request. The server can use this information to create a page that best displays upon that particular browser configuration. Perhaps the most efficient way of keeping information on customized options is through the use of cookies, a small string of data that is stored on the client machine at the request of the server when a page is transmitted. On a subsequent request to that site, the server can obtain the previously established cookie, read its content, and select particular content to deliver. For example, a Web site might have a registration page, and upon completion of this form the server places a cookie containing the user's name. When that person selects a new page on the site, the server grabs the cookie, reads the name, and then personalizes the response by including the name on the delivered Web page.

Print-based chemistry journal articles are static objects whose layout is determined by an editorial staff and this same layout is delivered to all readers. Realize that the term "layout" here refers to more than just the look of the page, i.e., the font face, the point size, the number of columns, size of the page and margins, etc. The standard units, the absence (or presence) of color images, the format of the citations, and the chemical nomenclature system employed within the articles are just some examples of editorial content/

10.1021/ci9800864 CCC: $18.00
Published on Web 10/02/1998

layout issues that have been dictated by the publisher. All authors must conform to this style, and if they do not, their articles will be typeset within these rules for them. The reader has no choice but to accept these dictates.

Dynamically generated HTML allows for the creation of content specific to the particular needs and requests of the reader. Instead of delivering the exact same page to all readers, it is possible to allow each reader to make their layout decisions independently. Customized pages designed to meet these criteria are then sent to the reader. Apply this concept to a chemistry journal and one has end-user customized chemistry articles delivered on demand.

Before detailing how this concept is implemented and what technologies are employed, we describe an example of how this concept works for a specific chemical application. At the NIST Webbook site,[11] a user can elect to receive the subsequent data in either SI or calorie-based units. Let us expand on this idea. Suppose an author who "thinks" best in units of kcal mol^{-1} would like to report an energy measurement in this unit. Upon reading the instructions for a traditional print journal, he discovers that all energies must be reported in units of kJ mol^{-1}. The author then makes all the needed unit conversions, submits the article, which is subsequently accepted and published. All readers of this article will find the energies reported in kJ mol^{-1} and some of whom may in turn do a mental conversion to yet another system, say eV.

How does this process work within the end-user customization electronic journal? The author is free to choose whatever energy unit is most convenient to him, so the article is written in kcal mol^{-1}. Upon acceptance by the electronic journal, the article is parsed and energy units are identified and tagged. This tagged document is then mounted on the journal's Web server. When a reader enters the journal Web site, he makes a decision on what the default energy units will be in all articles he accesses. Suppose this reader selects that all energies be expressed in units of ergs. When the reader requests the article under discussion, the Web server scans through the document and when an energy tag is found, a substitution of the appropriate unit is made, in this case into units of ergs. The reader is then sent this article and finds only energy units of ergs. Sitting at the terminal next door is another reader who has requested the exact same article but wants the energies expressed in kJ mol^{-1}; this reader views the exact same content except for the energies which have been automatically (and transparently) converted to the appropriate base. The bottom line is that all authors and all readers can select their personal preferences without imposing any conditions on anyone else; this freedom is handled behind the scene of view of all parties.

IMPLEMENTATION

End-user customization was a critical goal of the journal. Although end-user customization of Web pages are typically handled by utilizing CGI scripts that are called by the Web server, we opted not to follow this procedure. Since nearly every single request of the server would involve some customization component, continually spawning CGI procedures will degrade the performance of the server to an unacceptable level. Therefore, we decided to modify the Web server itself to handle the customization features directly, without calling an external script.

To provide end-user customization of articles, we must provide a mechanism for marking up the articles in such a way so as to enable substitution of the specific modifications requested by each individual user. We do this by placing a custom tag within the HTML file of an article. This tag is not a standard HTML tag; though formatted like one, it is not sent to a browser. When the file is requested, this tag is interpreted by the server and is replaced with standard HTML. The following example will suffice to describe this implementation. We have chosen to describe an example that is directly applicable to chemistry; however, the *IJC* system allows for layout customizations (number of frames per window, color of text, inline icons, etc.) and can be extended to handle any type of formatting or layout need.

The author of an article writes the following line within his article, which is submitted as part of an HTML file:

The ring strain energy of 10 is 23 kcal mol^{-1}, significantly less than that of 11.

The energy unit used here (kcal mol^{-1}) is arbitrary; it is simply the unit most convenient to the author but perhaps not to a reader. How does the *IJC* server allow for this unit to be substituted with another unit chosen by the reader?

First, the authors' submitted HTML text must be parsed and the unit recognized. The parsing is handled automatically via an "intelligent parser", written in Perl, which incorporates sophisticated syntax analysis and elements of artificial intelligence to identify components of the submitted article. This is an important point. Authors submit ordinary HTML code without any special formatting or tags; the parser creates the customization features. When an energy unit is found (kcal mol^{-1} in this case), a custom tag is generated to flag this unit. Most custom tags are formatted as <*--name?field1=value2;field2=value2...--*>. This format follows the standard for placing a comment within an HTML page. The "units" tag has the format <**--unit?from=*type*; value=*value*--**>, where *type* can be a variety of standard units in different categories (currently, energy, distance, and temperature but easily extendible to other measurements), and *value* is the quantity. The above example now formatted with the units tag is shown below:

The ring strain of 10 is <--unit?from=kcal/mol;value=23-->, significantly less than that of 11.

These custom tags are not meant to be read or interpreted by a browser. When a reader requests a page, the server does not simply deliver this HTML. The custom tags format is not absolutely arbitrary; they follow the standard HTTP practice for in-page comments. The units tag shown above would be interpreted by a standard browser as a comment and would, therefore, be completely ignored. In part, this is a precaution. If for any reason at runtime one or more custom tags were mishandled and find their way into the delivered HTML code, they will not cause a browser failure. In order for these tags to be properly handled, the server scans each page for the presence of any custom tags, such as the units tag. If a custom tag is found, the server will generate standard HTML to replace the custom tag so that the correct option and feature can be presented on any Web browser.

In the example above, let us suppose the reader has selected that all energy values are to be displayed in units of kJ mol^{-1}. Therefore, the server will have to convert the

23 kcal mol^{-1} to 96.23 kJ mol^{-1}. In addition, the *IJC* server will create a hyperlink attached to this value so that when this hyperlink is selected, the server will deliver a page with this energy value converted into a number of different units.

Continuing with our example, when the reader makes a request for the page with the example described above, the following steps are taken. Initially, the server scans the file for any custom tags. When a custom tag is located, the server matches the tag name with the appropriate solution, which is either a straightforward template (text) substitution or a server-side executable program. We have implemented most of these solutions as Java classes for the sake of consistency and cross-platform portability, but the native interface allows these solutions to be any executable code, for example, Perl scripts or C programs. In our example, the units tag is handled by the units Java class. Before executing this class, the server compiles a working environment that it provides to each solution. The following information is always passed:

(a) The parameter list from the tag itself.

(b) The network environment which includes all the information that is contained within the http request (i.e., the user's IP address, the browser configuration, etc.).

(c) The current user profile which contains all options selected by the individual.

The class produces HTML code as its output which is used by the server to completely replace the custom tag. This HTML substitution might be a completed piece of code or it may contain additional embedded custom tags which will be recursively handled. When all of the custom tags have been processed, the resulting HTML file is sent to the browser.

For our example, the HTML delivered to the reader's browser is as follows:

The ring strain of 10</10> is <a onMouseOver="window.status"='Convert this unit'; return true;" target=other href=convert.unit.html?value=23&from=kcal/mol&>96.23 kJ/mol, significantly less than that of 11.

How has this code been created? First, one user option is to have a units conversion table made available. This table will display the original value and unit followed by the equivalent value in a number of alternative units. This conversion table can be displayed within a frame or within a new window. The user specifies the options he desires using a preferences page, and these options are stored within the user's profile. (A schematic diagram of the article options preference page is given in Figure 1.) The units Java class reads this user profile and finds (in this case) that the user wishes to have this table displayed in a frame via a hyperlink access. Therefore, a standard anchor tag (the <*a*...>... combination) is created with the target specifying that the table be displayed within the frame named "other". Next, the units class checks the network environment to determine if the browser supports JavaScript. Finding the answer to be yes, the class then creates the code for displaying the text "Convert this unit" in the browser message bar. The "href" portion designates which HTML file will be delivered when the hyperlink is selected, in this case the file **convert.unit.html** with options after the question mark. These options pass the unit and value to a new class that will create the final HTML code for the units conversion table. Next, the units class converts the value to the appropriate unit (kcal mol^{-1} into kJ mol^{-1}) and places this text within the anchor tags. This HTML code is then the output of the units class and is returned to the server. The server replaces the custom units tag with this generated HTML code, ready to be sent to the reader's browser. The goal is now achieved—the reader views the article customized to his specifications, while the author has drafted the article in units he desires without having to create any out-of-the-ordinary HTML code.

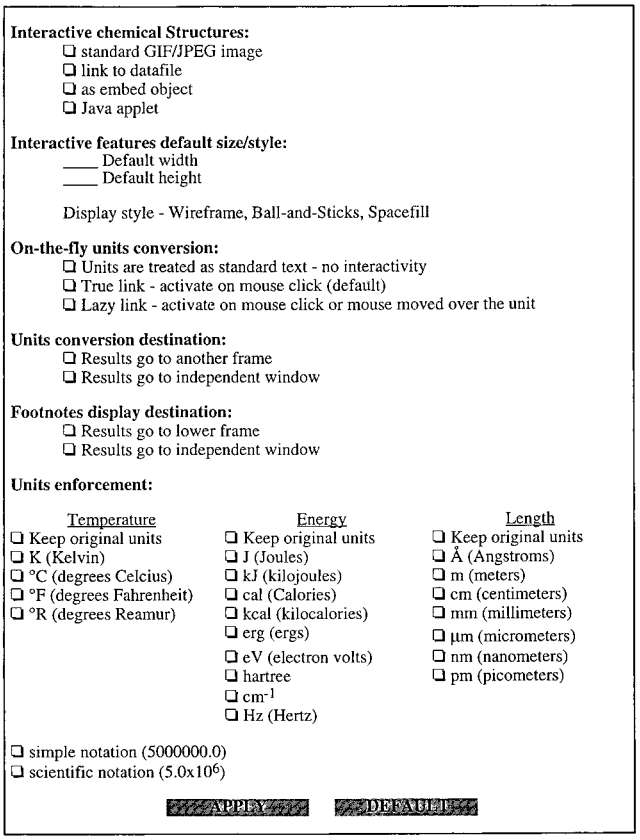

Figure 1. Schematic diagram of the Articles Options Preferences Page of the *Internet Journal of Chemistry*.

A couple of caveats need to be made here concerning the extent and scope of unit substitution allowed. First, unit substitution could potentially alter the meaning of the authors. As an example, suppose an author writes "**The absorption at 420 nm is longer than the expected 350 nm"**. Suppose a reader requested a replacement of wavelength with frequency (the point here is an inverse rather than linear conversion). The statement would then read "**The absorption at 7.14 × 10^{14} s^{-1} is longer than the expected 8.56 × 10^{14} s^{-1}"**. This is clearly a nonsensical statement. Instead of just a simple unit conversion, the sentence would have to be parsed and the comparison word ("longer") must be identified and then inverted. We avoid this problem by allowing only linear unit conversions at this time.

The larger problem related to this is that authors may refer to a given measurement in a variety of places within a manuscript and that some of these may not be placed with the *text* portion of an article. For example, a measurement may be part of a plot or scheme, which is contributed as a graphic image, and therefore not easily machine-recognized as a measurement with an associated unit. Therefore, if the

author has labeled the unit within the plot and also within the text, the user could alter the unit in the text, but this would no longer match the value in the figure, which would have remained unchanged. This situation can occur within the current operation of the journal, but we do not feel that this is a major deterrent. The reader can always opt for the units to appear in their default (author-defined) version and then there is perfect agreement. Alternatively, the reader can click on the measurement which will bring up a conversion table that provides a value in the same unit as in the plot.

The best solution, however, requires some effort from the author. Instead of supplying stagnant graphic images of plots and schemes, authors could supply the actual data along with a Java applet or ready for input into a spreadsheet plugin for generation of the plot. The reader could then manipulate the data (for example, by performing a unit conversion) and replot the figure. This is in fact an ideal situation in that real data is placed within the hands of the readers, who can then not only reproduce the work of the authors but also continue to explore it for new insights. However, until more efficient ways of including this type of data within a document are developed (see the CML section below), unit disagreements between text and graphic may occur.

Another example of a custom tag is the "chemmodel" tag, which designates an available 2-D and/or 3-D structure. At present, we are using this tag to allow the reader to designate how 3-D structures should be presented. For example, the reader can opt to have the structure embedded within the document using Chime[12] or using ChemSymphony[13] (a Java applet) or to have a static image presented instead. A few of the fields associated with this tag are

- available formats which specify all immediately available data representations for the structure (MDL molfile, xyz coordinates, Gaussian output, etc.),
- available images indicating that a static graphic may be substituted,
- preferred width and height of the embedded object.

Extension of this tag will include adding a SMILES field, allowing for chemical structure searching and indexing.

Some information about an article are of global interest; they are applicable to the entire article rather than to any specific part. This includes, for example, the article title, authors' names, mailing address, and keywords. These items could have been identified by creating a custom tag such as <--authorName?name=*names*--> within the actual article. Instead, we have opted to create an information file for each article. This file contains these custom tags, formatted without the leading or trailing angle brackets. The advantage of this procedure is that each article has associated with it a commonly named file that contains information that can be conveniently collected and processed. For example, when the server is initialized it creates the table of contents, author index, and keyword index (among other global collections) by gathering the tagged information from these information files. These collections are readily accessible to various solutions and can be formatted and inserted into generated HTML pages. The ability to regenerate a majority of commonly used data structures during the initial server startup can drastically improve its later performance.

THE JOURNAL AS A COLLECTION OF OBJECTS

In a sense, we are dramatically remaking the concept of a scientific research journal. The traditional print model has a single delivery mode of text and graphics, forever fixed upon the page, delivered in immutable form to isolated readers. In our model, the journal becomes a large interconnected collection of objects, cross-linked and cross-referenced into a single web. Each reader is an object, in that the interaction of the reader with the journal is through a URL request, which is explicitly handled as a request object and a user profile (yet another object). The journal is one large object. The journal object is made up of article objects, individual request objects, a server object, solution objects, etc. As we continue down the hierarchy, we see that each article object contains page objects, which contain molecular structures, figures, charts, and references objects, which contains the individual records objects. This set of objects interacts with each other and the network and hardware environment to create the "journal" delivered to each reader.

This object model is one of the major reasons why we chose to construct the *IJC* server in Java, an inherently object-oriented programming language. Additionally, Java allows us to deliver the journal on a variety of platforms with little reconfiguring for the specifics of the hardware. This feature has already proved useful as the server has been moved twice between different UNIX boxes without recompiling a single line of code. The strong exception handling ability and automated garbage collection of Java provides a server which is extremely stable; a class or object may fail without crashing the entire server, an important consideration for a journal that will be constantly accessed from around the world.

RELATION TO CML

There is a strong similarity between the concept described here and Chemical Markup Language (CML).[6,14] CML is a specific example of XML, extensible markup language. The XML concept is to expand the functionality of HTML into areas beyond page layout issues. XML defines a new set of tags, similar to HTML tags in their format, which markup (identify) elements of a file by giving it a semantic context. An example of this might be to markup a mathematical expression with tags that indicate, for example, fractions, matrices, integrals, etc., both from a content and presentation perspective. The CML proposal involves marking up documents to identify chemical content, such as units, molecular structure, elemental structure, spectral properties, etc. These tags are similar in format to the custom tags we have employed.

The key to XML is having a browser that can then process the XML tags. A number of different options are being investigated for this solution. A customized XML-aware browser can be implemented. A plugin to the standard browser can handle the XML code. A third option is to have a set of Java applets interpret the XML. A standard, widely distributed CML browser is still unavailable, though a prototype version has been developed.[15]

We felt that implementation of CML into the journal is premature; however, the concept can be exploited now. Instead of having the tags interpreted by the browser (which is the procedure for any XML implementation), the tags are

interpreted by the server and converted to standard HTML that can be read by any of the commercial or freeware browsers.

FUTURE

Full implementation of custom journal presentations awaits the time when the CML standard is fixed and adopted and when CML-aware browser solutions are available. Our custom tags are sufficiently similar in format to CML that conversion to true CML in the future should be straightforward. In this CML-world, a user will be able to select the specific tools he or she wants to display structures, spectra, chemical formulas and reactions, amino acid sequences, etc. These tools will allow direct user interaction with the chemical content of articles. Our solution is a bridge to carry us until full CML implementation is a reality. The power of end-user customization is so compelling, however, that this bridge solution is certainly worth bringing to the chemistry community now. The days of immutable journal presentations are numbered, and the age of tremendous reader control over information delivery and presentation has dawned.

Acknowledgment is made to InfoTrust Ltd. for continuing support of this research. Early developments were supported by a Henry and Camille Dreyfus Chemical Informatics Award. The authors thank Drs. Henry Rzepa and Peter Murray-Rust for assistance and guidance.

REFERENCES AND NOTES

(1) Rzepa, H. S.; Whitaker, B. J.; Winter, M. J. *J. Chem. Soc., Chem. Commun.* **1994**, 1907−1910. URL: http://www.ch.ic.ac.uk/rzepa/RSC/CC/4_02963A.html.

(2) Casher, O.; Chandarmohan, G. K.; Hargreaves, M. J.; Leach, C.; Murray-Rust, P.; Rzepa, H. S.; Sayle, R.; Whitaker, B. J. *J. Chem. Soc., Perkin Trans. 2* **1995**, 7−11. URL: http://www.ch.ic.ac.uk/rzepa/RSC/P2/4_05970K.html.

(3) Bachrach, S. M. In *Proceedings of the 1996 International Chemical Information Conference*; Collier, H., Ed.; Infonortics: Calne, England, 1996; pp 135−140.

(4) Rzepa, H. S.; Casher, O.; Whitaker, B. J. In *Proceedings of the 1996 International Chemical Information Conference*; Collier, H., Ed.; Infonortics: Calne, England, 1996; pp 141−148.

(5) Bachrach, S. M.; Murray-Rust, P.; Rzepa, H. S.; Whitaker, B. J. *Network Science* **1996**, *2*; URL: http://www.netsci.org/Science/Special/feature07.html.

(6) Murray-Rust, P.; Rzepa, H. S.; Whitaker, B. J. *Chem. Soc. Rev.* **1997**, 1−10; URL: http://www.rsc.org/is/journals/current/chsocrev/csr398.htm.

(7) Bachrach, S. M., URL: http://www.chem.niu.edu/IJC/Text/.

(8) Bachrach, S. M.; Burleigh, D. C.; Krassavine, A. *Issues Sci. Technology Librarianship* **1998**, *17*, URL: http://www.library.ucsb.edu/istl/98-winter/article1.html.

(9) See, for example: the *J. Molecular Modeling* (URL: http://www.organik.uni-erlangen.de/info/JMOLMOD/) and the Proceedings of the Second and Third Electronic Compuational Chemistry Conferences, published in the *J. Mol. Struct. (THEOCHEM)* (URL: http://www.elsevier.nl/homepage/saa/eccc2/ and http://www.elsevier.nl/homepage/saa/eccc3/) and Supporting Information for the *J. Am. Chem. Soc*. (URL: http://pubs.acs.org/subscribe/journals/jacsat/supmat/index.html).

(10) InfoTrust Ltd., *Internet J. Chem.*, URL: http://www.ijc.com/.

(11) NIST, *NIST Webbook*, URL: http://webbook.nist.gov/.

(12) Molecular Designs Ltd., *Chime*, URL: http//www.mdli.com/chemscape/chime/.

(13) Cherwell Scientific, *ChemSymphony*, URL: http://www.cherwell.com/chemsymphony/.

(14) Murray-Rust, P., URL: http://www.venus.co.uk/omf/cml/doc/index.html.

(15) Murray-Rust, P. *JUMBO*, URL: http://www.venus.co.uk/omf/cml/download.html.

CI9800864