

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau



(43) International Publication Date
14 February 2002 (14.02.2002)

PCT

(10) International Publication Number
WO 02/13035 A1

(51) International Patent Classification⁷: **G06F 15/16**, (74) Agents: JABLON, Clark, A. et al.; Akin, Gump, Strauss, 15/173, 13/00, 13/14 Hauer & Feld, L.L.P., One Commerce Square, 2005 Market Street, Suite 2200, Philadelphia, PA 19103-7986 (US).

(21) International Application Number: PCT/US01/24675

(22) International Filing Date: 7 August 2001 (07.08.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/223,394 7 August 2000 (07.08.2000) US

(71) Applicant: ACTIVE DATA EXCHANGE, INC.
[US/US]; 190 Brodhead Road, Suite 300, Bethlehem, PA 18017 (US).

(72) Inventors: MARTIN, Richard, D.; 4646 Virginia Drive, Bethlehem, PA 18017 (US). WETZEL, John, E.; 313 S. Egg Road, Bath, PA 18014 (US).

(81) Designated States (national): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

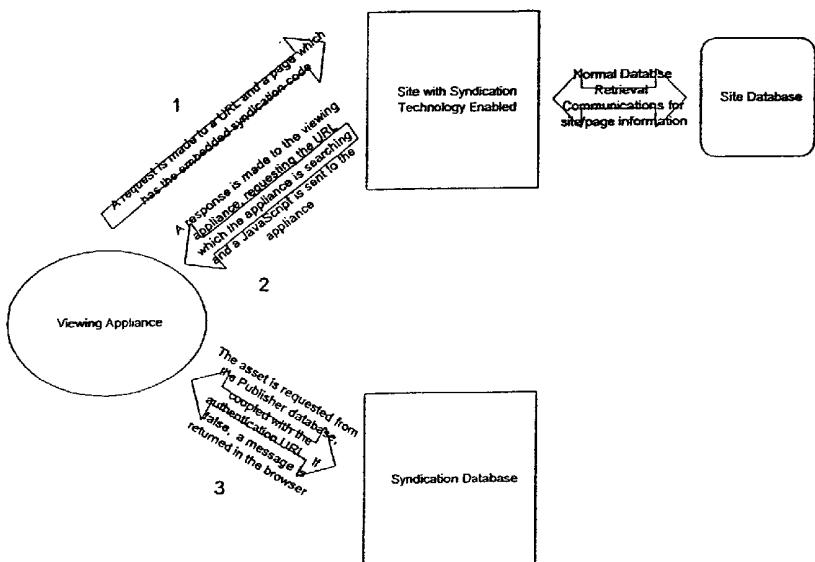
(84) Designated States (regional): ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— with international search report

[Continued on next page]

(54) Title: SYNDICATION METHODOLOGY TO DYNAMICALLY PLACE DIGITAL ASSETS ON NON-RELATED WEB SITES



WO 02/13035 A1

(57) Abstract: Digital assets are syndicated by constructing a web page and inserting into the web page JavaScript associated with a digital asset that is desired to be part of a fully rendered web page. The content of the digital asset is not initially part of the web page. The script, when executed by a browser, requests the content of the digital asset from a remote site (1). The request includes a uniform resource identifier (URI) of the web page and a unique identifier of the selected content. The remote site receives the request and authenticates whether the URI is authorized to receive the selected content (3). If so, the remote site locates the selected content and sends the selected content to the web browser (3). The web browser assembles the initially requested web page using the selected content obtained from the remote site (2).

WO 02/13035 A1



For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

TITLE OF THE INVENTION

SYNDICATION METHODOLOGY TO DYNAMICALLY PLACE
DIGITAL ASSETS ON NON-RELATED WEB SITES

CROSS-REFERENCE TO RELATED APPLICATIONS

5 This application claims the benefit of U.S. Provisional Application No. 60/223,394 filed August 7, 2000, entitled "SYNDICATION METHODOLOGY TO DYNAMICALLY PLACE DIGITAL ASSETS ON NON-RELATED WEB SITES."

COPYRIGHT NOTICE AND AUTHORIZATION

10 Portions of the documentation in this patent document contain material that is subject to copyright protection. The copyright owner has no objection to the facsimile reproduction by anyone of the patent document or the patent disclosure as it appears in the Patent and Trademark Office file or records, but otherwise reserves all copyright rights whatsoever.

BACKGROUND OF THE INVENTION

15 The present invention is directed to the process of Active Data Syndication and its use in web development and content management. More specifically, the present invention establishes the mechanism, the framework, the interfaces and the ancillary methodology to allow multiple content creators to offer for targeted syndication or network distribution and retrieval of digital assets across the web and to have those assets present within the constraints 20 of the receiving web site, and to automate that process, where warranted.

Conventional web sites consist of a wide variety of articles and information compiled and entered independently by a webmaster. Tools abound to facilitate the production of such web sites, from textual HTML editors to visual page designers like Adobe GoLive. Programs like LinkBOT exist to validate links among pages. However, the conventionality of 25 this form of web site generation becomes merely a shell for the advent of the more advanced, dynamic and interactive web site.

Site management tools allow for the collaborative efforts of site creation, but are significantly limited in their ability to share information and digital assets across the boundaries of web sites.

Syndication or distribution of digital assets across the web heretofore involved the 5 direct replication of those assets to a database server under the control of the receiving web site manager, and away from the control of the supplier of the digital asset. Examples of implementations where content replication is the primary form of asset syndication include those from Vignette, Kinecta, Interwoven, ICE, and ArcadiaOne.

Accordingly, there is a need for a system which (1) unifies the way digital assets, in 10 any form, are shared from one site to another, (2) unifies the distribution of those digital assets across multiple platforms, (3) allows the owners of digital asset a management tool for tracking the business relationships surrounding the use of the digital assets, (4) establishes mechanisms, interfaces and methodology for the secure flow of information through information distribution networks, (5) facilitates the webmaster to have updated digital content present on the web site, 15 and (6) automates the update of web content. The present invention fulfills these needs.

Prior to the present invention, it was not possible for web sites governed by any number of normal content management tools to proactively share information with independent and unrelated web sites. The lightweight and transportable tool of the present invention allows for a web site to be programmed to receive digital asset information in a multiplicity of formats, 20 and to display those assets within the style and constraints of the receiving web site. The web sites need not be collocated, nor do they need to be created using the same tools. They simply need to render HTML in any web browser, capable of interpreting Java and JavaScript. The power of this tool is such that it can enhance, if not revolutionize, the bi-directional communications infrastructure using the web. These managed information distribution 25 networks do for the Internet what faxes and overnight mail did for corporate correspondence two decades ago.

SUMMARY OF THE INVENTION

The present invention provides a scheme of obtaining selected content for a web page, wherein the selected content itself is not initially part of the web page. The web page includes script, such as JavaScript, associated with the selected content. The scheme operates as follows:

1. A web browser requests a web page that includes script associated with the selected content. The selected content may be only a portion of the web page. The selected content may be a digital asset or an executable file. In the preferred embodiment, the web page is constructed using HTML, and the script is embedded therein.
2. The web browser interprets the script and formats a request to obtain the selected content from a remote site. The request includes a uniform resource identifier (URI) of the web page and a unique identifier of the selected content. The URI may be a URL.
3. A remote site, such as a web server, receives the request and authenticates whether the URI is authorized to receive the selected content. If so, then the remote site locates the selected content and sends the selected content to the web browser. The selected content may be stored in a content repository connected to the web server. If the URI is not authorized to receive the selected content, then the remote site sends a signal to the web browser that the selected content is not available, and the web browser assembles the web page without the selected content.
4. The web browser assembles the initially requested web page using the selected content obtained from the remote site.

The assembled web page may include one or more content sets from the syndicator, each having its own script for implementing the steps above.

In one preferred embodiment, the script includes a subscriber identifier and a content identifier which are both used to create the unique identifier of the selected content.

The present invention also provides a scheme for syndicating digital assets. A web page is constructed, and script, such as JavaScript, associated with at least one digital asset that is desired to be part of the fully rendered web page is inserted into the web page. The script,

when executed by a browser, performs modest authentication regarding the URI and requests the content of the digital asset from content repository. The request includes a uniform resource identifier (URI) of the web page and a unique identifier of the selected content. In one preferred embodiment, the script includes a subscriber identifier and a content identifier, which, 5 together, create the unique identifier of the selected content. The selected content may be an executable file.

BRIEF DESCRIPTION OF THE DRAWINGS

The following detailed description of preferred embodiments of the present invention would be better understood when read in conjunction with the appended drawings.

10 For the purpose of illustrating the present invention, the drawings show embodiments of the present invention which are presently preferred. However, the present invention is not limited to the precise arrangements and instrumentalities shown. In the drawings:

Fig. 1 is a schematic block diagram that provides an overview of one preferred embodiment of the asset syndication scheme of the present invention;

15 Fig. 2 is a database schema for one preferred embodiment of the present invention;

Fig. 3 is an authentication schema for one preferred embodiment of the present invention;

Figs. 4-13B are user interface displays (administrative screen shots) for one preferred embodiment of the present invention;

20 Fig. 14 is a database schema for the second preferred embodiment of the present invention;

Figs. 15A and 15B, taken together, are overall schemas for the second preferred embodiment of the present invention;

Fig. 16 is a schematic block diagram of the second preferred embodiment of the 25 present invention;

Figs. 17-19 are JavaScript source code snippets for implementing a web application embodiment of the present invention; and

Fig. 20 is a database schema for the web application embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

Certain terminology is used herein for convenience only and is not to be taken as a limitation on the present invention. In the drawings, the same reference letters are employed for designating the same elements throughout the several figures.

5 I. DEFINITIONS

Content: Any textual, visual, and audio materials or the combination thereof, including animated images, video clips, executable files, or digital assets targeted for presentation.

Content Repository: a database of file structure which contains the syndicatable asset.

Syndicated Asset: Any content which is available to an authenticated receiver.

10 Authenticated Receiver: A receiver of content that is authorized to retrieve and present that content.

Subscriber: Synonym to Authenticated Receiver.

Domain: The name which appears in the URL between the “www” and the end of the three letter extension (e.g., .com, .net , .org). Example: www.regiononline.com, the domain is
15 regiononline.com.

II. OVERVIEW OF PRESENT INVENTION

Referring to Fig. 1, the following steps are performed:

1. A request is made from a web appliance to a URL, which contains a web page that has embedded the syndication code.

20 2. The web page, while rendering other data present on the page, such as navigation, headers and footers, and other assets not related to the syndicated content, returns JavaScript which tests the appliance for the site URL requested. The JavaScript effectively functions as a dynamic content retrieval agent.

25 3. This information is then used to authenticate the content presence on the site as valid, and then to retrieve from the syndication database those assets for display, and render those syndicated assets through the appliance.

The present invention introduces the notion of web real estate and the apportionment of content sections to a particular web page. A visually unified site is substantially divided into sub-sites, or content sections, that are independently maintained by section managers or through syndicated content offerings. As a direct and targeted information management delivery tool, 5 the present invention is useful to the large corporation, a community site, or any site which enables the sharing of targeted information. Any content or information rich site will find this present invention very helpful.

10 The present invention may be used to complement web pages created in MS Frontpage, or any number of content management tools. The present invention does not perform visual layout, and may be considered a post-design tool.

The present invention may be used with any number of programming languages such as Cold Fusion, ASP, C++, Java, Visual Basic or Perl. The present invention may serve as an extension to the web page, or an add-on component to any number of content management tools.

15 The present invention may be used on any site that renders HTML, and is intended to be programming language neutral, through the use of JavaScript, servlets, and Java features.

III. DETAILED DESCRIPTION OF ONE PREFERRED EMBODIMENT

In one preferred embodiment of the present invention as described herein, the user communicates through a browser with the necessary web site via an electronic network, such as 20 the Internet. However, the scope of the invention includes other types of user interfaces and electronic networks that are capable of performing the desired functions.

The present invention is described in the context of a commercially available software product called Active Data Syndicator™, available from Active Data Exchange, Inc., Bethlehem, Pennsylvania.

25 A. DETAILED EXPLANATION OF FIGURES AND APPENDICES

Fig. 2 is a self-explanatory database schema for one preferred embodiment of the present invention, and Fig. 3 is a self-explanatory authentication schema for one preferred embodiment of the present invention.

Fig. 4 is an administrative entry screen for beginning the process.

Fig. 5 is a user interface display that allows for the selection from an existing syndication or the creation of a new syndication.

Referring to Fig. 6, if the administrator chooses the “go get it” button from the Fig. 5 display, then a listing of the syndication offers are presented. The “clickable” first field 5 allows for the modification of that offering.

Figs. 7 and 8 are user interface displays a for a “New Setup.” The administrator completes the fields shown in these displays.

Fig. 9 is a user interface display that allows for the viewing of the necessary syndication code for placement into the HTML of the subscriber/receiver web site.

10 Fig. 10A shows the actual code for placement on the subscriber/receiver HTML page for the fictitious client Attorney at Law. This code allows for the one time insertion of the code for presentation of information, news, events, or other digital assets on an ongoing basis.

15 Fig. 10B shows the same code, modified to present the latest asset, as well as the creation of several hypertext links for viewable archives of older digital assets. In this case, there is the ability to present for viewing up to 10 viewable archives from the database.

Figs. 11 and 12 show additional administrative functions for the deletion of a subscriber.

Figs. 13A and 13B show the list of subscribers/receivers for overall administrative purposes.

20 Appendix A is the syndication source code for the embodiment of the present invention shown in Figs. 2-13B.

Appendix B is the servlet package source code for the embodiment of the present invention shown in Figs. 2-13B.

B. INHERENT SECURITY

25 All web sites do not use the same syndication code. Subtle differences in the code are “keyed” to the receiving domain. It is through this “key” that the integrity of the business relationship and the placement of the digital asset are preserved. The process includes a security module that looks at the browser URL of the appliance to determine which URL 30 (domain) it is calling. If the domain does not match an existing client URL (coupled with the content authorized for use), a message is returned to the appliance, which states that the content is unavailable.

C. WEB SITE SETUP FOR RECEIVING SYNDICATION OFFERINGS

To set up a web site page for syndication, a few parameters must be understood and defined:

1. The receiving URL, as authentication or validation of the receiver/subscriber is performed to the domain level.
- 5 2. The asset section to be syndicated. This refers to a content section in a content management system. One instant example would be to syndicate "What's News" from a company or organization. This section of the site is a hypothetical region where all information regarding press releases and latest organizational information is to be found. In this installation, the site only needs to be set up with the HTML snippet one time. Content changes occur dynamically.
- 10 3. The number of displayed archives the receiver would like to be able to view. This is a title listing of previously syndicated content for this section.
- 15 4. The database location of the digital asset.
5. The organizational name (for administrative purposes).
6. The start and end time for the syndication. This is important if the syndicated asset is time dependant or whether or not the subscription to the syndicated digital asset is on a monthly or other time dependent basis.

20 Through these parameters, metrics can be derived which display, from the syndication-offered standpoint, several management reports useful in understanding the offerings and their use. Metrics include:

1. Number of times a digital asset is accessed by a subscriber (useful in fee per use models as well as for tracking relative worth of the asset).
- 25 2. Places where the digital asset can be viewed/authorized domains.
3. Listing of the offerings available for release through the syndication model.
4. Through an oblique methodology, a measure of the physical overhead of the device housing that content.

D. DATABASE INTERRELATIONSHIPS

30 As discussed above, the present invention is platform independent and program neutral. It functions equally as well on operating systems written in Microsoft, as it does from

that of Red Hat Linux. Furthermore, the database from which it pulls its content may either be MS SQL, Informix, Sybase or Oracle, as it uses command phrases which are non-vendor specific (ANSI SQL). The hardware, however, must be of sufficient strength to power databases that conceivably will receive hundreds of thousands of requests per second. Fig. 2, 5 described above, illustrates one preferred database schema for the present invention.

E. ARCHITECTUAL OVERVIEW

The implementation of this business process and its underlying software is hardware independent, providing that the following applications/services are available: An application server running a servlet engine such as Allaire's JRUN or TomCat, and a web server such as 10 Internet Information Server (IIS) from Microsoft or an Apache web server should the operating system of the server be RedHat Linux or Unix. The database server should have a strong relational database such as MS SQL, Oracle, or Sybase, overtop of the appropriate operating system.

The location of the servers used in the present invention can be anywhere within the 15 infrastructure of the existing client network. One suitable scheme is to locate the servers behind a redundant firewall.

The invention environment provides for all system hardware requisite to the invention work. It includes an application/web server that controls the look of web pages, serves content for the web pages, and provides the environment for the administration of those 20 tasks. The standard operating system for this invention is Linux. However, a client may choose Windows NT or UNIX as an alternate operating system (OS). Coupled with the OS, a web server needs to be chosen. In the example of Red Hat Linux, one suitable choice would be Apache. However, a Windows NT environment would most likely use IIS. Since the present invention is a Java-based application, a servlet engine needs to be incorporated into the server 25 operation. One suitable configuration to support the invention includes the following elements:

Operating System (Application Server)

Windows NT 4.0 with Service Pack 6A

Red Hat Linux 6.2

Databases (Database Server)

30 Sybase 11.02 [Windows NT, Linux]

Oracle 8i Server 8.1.5 [Windows NT, Solaris, Linux]

Microsoft SQL Server 6.5 with Service Pack 6A [Windows NT Only]

Microsoft SQL Server 7.0 [Windows NT Only]

JDBC Drivers

5 Microsoft SQL Server 6.5, 7.0: SPRINTA JDBC 2.0 Driver version 2000 [Type 4]

Oracle 8.0.5.0. Production release JDBC [Windows NT, Solaris, Linux]

HTTP Servers (Web Server Types) Operating JRUN 2.3.3 or greater

Netscape Enterprise Server 3.0, 3.6 [Windows NT, Solaris, Linux, AIX]

Microsoft Internet Information Server Version 4.0 [Windows NT Only]

10 Apache and Stronghold Web Server 2.4.2 [NT, AIX, Solaris, Linux]

Java Development Kit (JDK)

Recommendation: Sun Compliant JDK minimum version 1.2.2

Example Configurations:

1. Linux Application Server with NT OS running the Database server

15 Application/Web Server: Red Hat Linux version 6.2 with an Apache web server.

Database Server: Microsoft NT 4.0 with Service Package 6A and MS SQL version 7 as the database.

2. Linux Application Server with Linux OS running the Database server

Application/Web Server: Red Hat Linux version 6.2 with an Apache web server.

20 Database Server: Red Hat Linux version 6.2 with Oracle 8i as the database.

3. NT Application Server with NT OS running the Database server

Application/Web Server: Microsoft NT 4.0 with Service Package 6A and IIS version 4 with full options as the Web Server.

Database Server: Microsoft NT 4.0 with Service Package 6A and MS SQL version 7 as the database.

25

The Database Server becomes more critical as database size increases over time. It is feasible for an installation to use a different OS for the application server than that of the database server. Different database engines serve data at different rates (some faster than others) and some networks operate with corporate standards which mandate the use of certain configurations, to standardize application environments for ease of maintenance. For those

30

reasons, the publishing database was developed platform independent. It is functionally transparent whether or not it is installed over Oracle, Sybase, MS SQL or Informix.

The present invention may be installed as part of a related product from Active Data Exchange, Inc., called Active Data Publisher™/Web Server, or it may be a functionally independent device. The setup of the environment of the present invention is the same as the Application/Web server environment.

F. EXAMPLES OF SYNDICATION USE AND APPLICATION

Case One: Large organization with multiple web sites (Public and Private)

A large organization has multiple web sites to manage, both public (visible to all) and private (internal and departmental in nature). The sites are repositories for a wide variety of information specific to their department of corporate division. There are, however, many assets which are frequently reproduced and shared among those sites. If a directive from Senior Management needs to be presented on each of the sites, conventional content management tools would require the HTML editors to re-post the directive on each and every site. This repetitive process creates organizational inefficiencies.

Using the present invention, an HTML snippet is placed on the page in the place where information bulletins will be seen. (See Figs. 10A and 10B which illustrate an HTML snippet.) Every time a new information bulletin is released, the web page is automatically updated, without technology intervention. Furthermore, the bulletin is created in the native environment frequently used by the author or their assistant, and posted once through an intuitive, easy to use interface module which converts the document to HTML, enters the data into the syndication database, and prepares it for view on the site.

Case Two: Supply Chain Information Management

A manufacturer manages information streams from raw materials suppliers (upstream providers) to distributors (end users). The management objective is to assure that the end user receives the completed goods at a fair price in a timely manner, and that the raw materials suppliers notify the manufacturer of delays in shipments, which, in turn, affect product availability. If one believes that the main differentiation between suppliers of similar materials is their ability to enhance communications with clients and vendors, and that issue singularly is what is going to set apart one from another, then the implementation of the tools

provided by the present invention is the linchpin in corporate success. In this example, it is helpful from a supply management role to understand the needs of the client and the ability for the vendor to supply product to create the materials for the client. Subscribing to the notion of apportioned web real estate, and having particular areas designated for information relevant to 5 those in the supply chain, a new communications modality is created called a Digital Information Network that is linking the supplier and the end user in such a fashion as to be able to enhance the decision process, increase productivity, and enhance the digital economy.

IV. DETAILED DESCRIPTION OF A SECOND PREFERRED EMBODIMENT

The second preferred embodiment may be used with Active Data Syndicator v.3.

10 A. DETAILED EXPLANATION OF FIGURES AND APPENDIX

Fig. 14 is a self-explanatory database schema for the second preferred embodiment.

Fig. 14 shows only the portion of the database schema that relates to the present invention.

Figs. 15A and 15B, taken together, are self-explanatory overall schema for the second preferred embodiment.

15 Fig. 16 is a schematic block diagram of the second preferred embodiment.

Appendix C is the combined syndication and servlet package source code for the embodiment of the present invention shown in Figs. 14-16.

B. IMPLEMENTATION OF SECOND EMBODIMENT

Fig. 16 shows an overview of the second preferred embodiment. The basic elements

20 include web pages located at a plurality of different URLs, a viewing browser, an application web server that hosts the syndication product, and a content repository. Each of these elements may be interconnected by any suitable communication medium, such as the Internet. The process operates as follows:

1. A user at the viewing browser requests a web page from a particular web site. The requested web page contains HTML elements, as well as at least 25 one snippet of JavaScript associated with the syndicated digital asset.

Alternatively, the requested web page does not have to reside on a web site. The requested web page may also be a simple HTML file stored on the hard drive of a user's local computer in cases where an authenticating URL is not required.

30 2. The snippet of JavaScript is received by the browser.

3. When the JavaScript is received, it is not immediately used by the browser on the part of the web page on which it resides. Instead, the “src” attribute of the JavaScript tag is used to make a call to a Java servlet. More specifically, the JavaScript forms an HTTP (or HTTPS) request that includes a URI (e.g., a URL) of the requested web page as obtained from the browser, and a unique identifier contained within the JavaScript. In one preferred embodiment, the unique identifier is a combination of a subscriber identifier (sub ID) and a content identifier (content ID). In effect, the HTTP request is saying that URI [xyz] is requesting content [123], and is asking if it is okay to deliver it. The HTTP request is sent to an application web server that is designated by an address located within the JavaScript.
4. At the application web server, the syndication product receives the URI and the unique identifier,
5. The syndication product then looks in an authentication table to determine if there is a matching URI and unique identifier. If so, then the unique identifier is parsed to obtain the sub ID and content ID. The sub ID is used to check the current account status of the subscriber, and the content ID is used to locate the content in the content repository.
6. Assuming that a match is found in the authentication table, the subscriber ID is properly authorized, and the content is located in the content repository, then the content is retrieved from the content repository and sent by the syndication product to the browser for insertion at the appropriate location during the rendering of the web page. To facilitate this process, the syndication product contains content filtering and parsing methods (called “parseContent” and “swapStrings” in the example source code) which are used to prepare the content so that it can be rendered appropriately in the web page via a JavaScript “document.write” statement. If no match is found in the authentication table, and/or if the subscriber ID is not properly authorized, then a message is returned indicating that requested content cannot be received. Alternatively, no message is returned and the user merely does not receive the requested

content. If the content is a text article, the web page may have a blank portion where the requested content would have appeared. If the content is multimedia-oriented, such as an audio file, then such content is not experienced.

- 5 7. The syndication content manager updates its records to reflect the activity.

The manager may track content retrievals and charge subscriber accounts (if any exist) for such content retrievals. The content manager may remove an entry from the authentication table based upon expiration dates, number of retrievals, or any other suitable factor.

- 10 The process described above preferably occurs seamlessly in near real-time. Thus, the user is not aware that content (which is typically only a portion of the web page, but could be the entire content of the web page) is being requested and delivered from a remote content repository during the rendering of the web page.

The term "web application" as used herein refers to dynamic HTML web site content which varies depending upon user input, includes one or more interactive forms, involves the use of a web server programming/scripting language (e.g., Java, Perl, Cold Fusion, Active Server Pages, etc.), and may also make use of a backend database server for data storage. Some common examples of web applications include guestbooks, forums and shopping carts. Web applications typically execute on the same server as the hosting web server. This arrangement can place significant strain on the web server, especially when a large number of users are simultaneously requesting service and/or many web sites and applications are running concurrently on the same server. (This situation often occurs with ISP's.) The present invention leverages the JavaScript capabilities of the user's browser to execute the web application at a remote server independent of the subscriber's hosting web server, thereby reducing the potential load on the subscriber's web server and greatly simplifying the process by which a web application can be incorporated into a subscribing web site. That is, simply include the JavaScript snippet of the application using the present invention's syndication methodology in the HTML on the subscriber's web site. No further programming is then needed on the subscriber's web server. The user's browser effectively invokes the remote web application that runs and makes the subscriber's web page dynamic.

The present invention may be used for flat (static) sites and flat sites having one or more dynamic sections, as well as for fully dynamic sites.

The present invention is further advantageous because the traffic for creating the web pages uses port 80 (for http traffic) and port 443 (for https traffic), and thereby can pass 5 through most server firewalls.

C. WEB APPLICATION EXAMPLE

Active Data Randomizer, available from Active Data Exchange, Inc., Bethlehem, Pennsylvania, is an example of a simple web application which incorporates the syndication methodology of the present invention. Randomizer produces two syndicated assets from two 10 different JavaScript snippets.

Fig. 17 shows the first JavaScript snippet which renders in the browser as an administrator area (see Fig. 18) for the subscribing web site administrator to use in the configuration and entry of groups of HTML blurbs.

Fig. 19 shows the second JavaScript snippet which is generated from the 15 administrative area and renders in the browser as a randomly selected HTML burl from a group of HTML blurbs specified during the generation of the snippet.

Appendix D is a User Guide for this embodiment, and Appendix E shows sample source code for this embodiment.

The present invention may be implemented with any combination of hardware and 20 software. If implemented as a computer-implemented apparatus, the present invention is implemented using means for performing all of the steps and functions described above.

The present invention can be included in an article of manufacture (e.g., one or more 25 computer program products) having, for instance, computer useable media. The media has embodied therein, for instance, computer readable program code means for providing and facilitating the mechanisms of the present invention. The article of manufacture can be included as part of a computer system or sold separately.

It will be appreciated by those skilled in the art that changes could be made to the embodiments described above without departing from the broad inventive concept thereof. It is understood, therefore, that this invention is not limited to the particular embodiments disclosed,

but it is intended to cover modifications within the spirit and scope of the present invention as defined by the appended claims.

What is claimed is:

Appendix A

Syndication Source Code

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.util.Vector; //most methods of Vector are synchronized
import java.net.*;

/**
 * Servlet to retrieve content from Publisher server
 * Takes 7-9 parameters:
 *
 * db, contentServer, linkBase, SectionID, ArticleID, PartID, mode,
 * unit, number
 *
 * @author Technology
 * @author ActivedataX.com
 * Last changed 11/9/99
 */

public class ContentNOWjs extends HttpServlet implements Constants
{
    private static int itemsToKeep = 100, minutesToLive = 10; // init
may set the values later
    private static long lastModified;
    private static Vector contents = new Vector();
    private Sentinel sentinel = new Sentinel(itemsToKeep, minutesToLive);
    private static final ReadURL serverURL = new ReadURL();

    public void doGet(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException
    {
        String contentID, detail;
        boolean newContent = false;
        Content cont; //needed for lookup
        Params params = new Params(req); //decipher the request
        String ContentID = params.contentServer //req.getParameter("Server")
+ "?db=" + params.db
+ "&SectionID=" + params.SectionID
+ "&ArticleID=" + params.ArticleID
+ "&PartID=" + params.PartID
+ "&ln=" + params.linkBase
+ "&mode=" + params.mode;
        if (params.unit != null)
            {ContentID = ContentID + "&unit=" + params.unit
            + "&number=". + params.number;
        }
        if (params.templateURL == null){
            params.templateURL =
"http://www.lehighvalleynow.com/rol/template4.asp";
        }
    }
}
```

```
ContentID = ContentID + "&tpl=" + params.templateURL;
// parameter tpl will be ignored by serverURL.getContent, used for
internal ID only
    cont = lookup(ContentID);
    if (cont != null)
        detail = cont.detail;
    else
    {
        try
        {
            System.out.println(ContentID);
            detail = serverURL.getContent(ContentID);
            detail = convertTojs(detail, params);
            newContent = true;
        }
        catch (Exception ex) {detail = ex.toString();}
    }

    //PrintWriter out = resp.getWriter(); // ServletOutputStream
bombs with \u0018: CharConversionException
    //out.print(detail);
    //System.out.println(detail);
    OutputStream out = resp.getOutputStream();
    out.write(detail.getBytes());
    System.out.println(detail);
    out.flush();
    out.close();

    if (newContent)
    {
        lastModified = System.currentTimeMillis();
        cont = new Content(ContentID, detail, lastModified);
        contents.addElement(cont);
    }
}

public long getLastModified(HttpServletRequest req)
{
    return lastModified/1000*1000;
}

private String convertTojs(String htmlText, Params params){
    htmlText = htmlText.replace('\r', ' ');
    htmlText = htmlText.replace('\n', ' ');
    htmlText = swapStrings(htmlText, "\\", backSlash);
    htmlText = swapStrings(htmlText, "\'", apostrophe);
    htmlText = swapStrings(htmlText, "\"", doubleQuote);

    if (params.ArticleID.equals("0")){
        htmlText = JavaScript.openArticle() + makeJavascriptHrefs(htmlText,
params);
    }

    return "document.write (" + htmlText + ");";
}
// pre-pend JavaScript.expandHeader() here for use by all header
```

```

items.//

//+ the above line is skipped mysteriously
// specific full article, just print
// parse out href=, get ArticleID, replace with
javascript:expandHeader(ArticleID)

private String makeJavascriptHrefs(String htmlText, Params params){
    //find "<a", find "href=", find "ArticleID=", find '>'
    //replace href= with javascript:expandHeader(ArticleID)
    int indexBeginA, indexHref, indexArticleID,
        ArticleIDEndsAt, indexEndA, indexClosingA;
    String ArticleID = null;
    String htmlTextLowercase = htmlText.toLowerCase();
    StringBuffer newHtml = new StringBuffer();
    int indexStart = 0;
    indexBeginA = htmlTextLowercase.indexOf("<a ", indexStart);
    indexHref = htmlTextLowercase.indexOf("href=", indexBeginA + 2);
    indexArticleID = htmlTextLowercase.indexOf("articleid=", indexHref +
5) + 10;
    indexEndA = htmlTextLowercase.indexOf(">", indexArticleID);
    indexClosingA = htmlTextLowercase.indexOf("</a>", indexEndA) + 4;
    while (indexBeginA >= 0 && indexBeginA < indexHref
        && indexHref < indexArticleID
        && indexArticleID < indexEndA
        && indexEndA < indexClosingA){
        ArticleIDEndsAt = htmlTextLowercase.indexOf("&",
indexArticleID);
        //ArticleID = 'a' + htmlTextLowercase.substring(indexArticleID,
ArticleIDEndsAt);
        ArticleID = htmlTextLowercase.substring(indexArticleID,
ArticleIDEndsAt);
        newHtml.append(htmlText.substring(indexStart,
indexHref)).append("href=");

        newHtml.append("\\\\\"javascript:openArticle\\\\\" +
params.templateURL + "?SectionID="
        + params.SectionID + "&ArticleID=" + ArticleID +
"&PartID=2&db=" + params.db + "\\\");\\\"");
        //newHtml.append("\\\\\"javascript:expandHeader(" + ArticleID +
");\\\"");
        newHtml.append(htmlText.substring(indexEndA, indexClosingA)); // done with one anchor.

//newHtml.append(JavaScript.appendArticleMain(params.requestURL,
params.db, params.SectionID, ArticleID) );
    //new loop starts over
    indexStart = indexClosingA;
    indexBeginA = htmlTextLowercase.indexOf("<a ", indexStart);
    indexHref = htmlTextLowercase.indexOf("href=", indexBeginA);
    indexArticleID = htmlTextLowercase.indexOf("articleid=",
indexHref + 5) + 10;
    indexEndA = htmlTextLowercase.indexOf(">", indexArticleID);
    indexClosingA = htmlTextLowercase.indexOf("</a>", indexEndA) +
```

```
4;           //else use old value to get the tail.
        }
    //newHtml.append(htmlText.substring(indexClosingA));
    return newHtml.toString();
}

Content lookup(String contentID)
{
    Content cont;
    int index = -1;
    for (int i=0; i<contents.size(); ++i)
    {
        cont = (Content)contents.elementAt(i);
        if (contentID.equals(cont.contentID))
        {
            return (Content)contents.elementAt(i);
        }
    }
    return null;
}

public void destroy()
{
    sentinel.quit = true;
    contents = null;
}

class Content
{
    String contentID, detail;
    long timestamp;
    Content(String contentID, String detail, long timestamp)
    {
        this.contentID = contentID;
        this.detail = detail;
        this.timestamp = timestamp/60000; //converts millis to min.
    }
}

/**
 * Monitors the content collection, trims and refreshes.
 */

private class Sentinel extends Thread
{
    int itemsToKeep;
    int minutesToLive;
    boolean quit = false;

    Sentinel(int itemsToKeep, int minutesToLive)
    {
        this.itemsToKeep = itemsToKeep;
        this.minutesToLive = minutesToLive;
    }
}
```

```
        setPriority(Thread.MIN_PRIORITY);
        start();
    }

    /**
     * The only and single thread
     */

    public void run()
    {
        int itemsToRemove;
        while( !quit ) // this keeps cleanup alive until servlet is
destroyed.
        {
            freshen();
            if (contents.size() > itemsToKeep)
                trim();
            //System.gc();
            try{
                Thread.sleep(5000);
            }catch (InterruptedException ex) {}
        }
    }

    private void trim()
    {
        int itemsToRemove = contents.size() - itemsToKeep;
        int trimmed = 0;
        for (int i=0; i<itemsToRemove; ++i)
        {
            contents.removeElementAt(i);
            Thread.yield();
            trimmed++;
        }
        if (trimmed > 10) contents.trimToSize();
    }

    private void freshen()
    {
        long currentTime = System.currentTimeMillis()/60000;
        Content cont;
        for (int i=0; i<contents.size(); ++i)
        {
            cont = (Content) contents.elementAt(i);
            if (cont.timestamp + minutesToLive < currentTime)
            {
                contents.removeElementAt(i);
                Thread.yield();
            }
        }
    }
}
```

```
public String getServletInfo()
{
    return "Retrieves content from PublishNOW server";
}

public void init(ServletConfig config) throws ServletException
{
    super.init (config);
    if ( getInitParameter("itemsToKeep") != null )
    {
        try{
            this.itemsToKeep =
Integer.parseInt(getInitParameter("itemsToKeep"));
        }
        catch (NumberFormatException e){
            this.itemsToKeep = 100;
        }
    }
    else this.itemsToKeep = 100;

    if ( getInitParameter("minutesToLive") != null )
    {
        try{
            this.minutesToLive =
Integer.parseInt(getInitParameter("minutesToLive"));
        }
        catch (NumberFormatException e){
            this.minutesToLive = 15;
        }
    }
    else this.minutesToLive = 15;
}

String swapStrings(String TextIn, String OldString, String NewString)
{
    int OldStringStartAt;
    int OldStringLength = OldString.length();
    int NewStringLength = NewString.length();
    String TempText = TextIn;
    OldStringStartAt = indexOfIgnoreCase(TempText, OldString, 0);
    while (OldStringStartAt != -1)
    {
        TempText = TempText.substring( 0, OldStringStartAt) +
                    NewString + TempText.substring(OldStringStartAt +
OldStringLength);
        OldStringStartAt = indexOfIgnoreCase(TempText,
                                         OldString,
                                         OldStringStartAt + NewStringLength);
    }
    return TempText;
}

int indexOfIgnoreCase(String TempText, String subString,
                     int StartAt)
```

```
{  
    return TempText.toLowerCase().indexOf( subString.toLowerCase(),  
    StartAt);  
}  
  
}  
  
class ReadURL  
{  
    public String getContent(String SourceURL) throws Exception  
    {  
        URL page;  
        BufferedReader in;  
        int len;  
  
        page = new URL(SourceURL);  
        in = new BufferedReader(new InputStreamReader(page.openStream()));  
        long time = System.currentTimeMillis();  
        CharArrayWriter chbuffer= new CharArrayWriter();  
        char[] cbuf = new char[1024];  
        while ((len = in.read(cbuf, 0, 1024)) != -1)  
        {  
            chbuffer.write(cbuf, 0, len);  
        }  
        in.close();  
        time = (int)(System.currentTimeMillis() - time);  
        return chbuffer.toString() + "<etime " + time + "ms/>";  
    }  
}  
  
class JavaScript implements Constants{  
    static String expandHeader(){  
        return "<script language=\\\"JavaScript\\\">function  
expandHeader(ArticleID) "+  
        "{ArticleID.style.display = (ArticleID.style.display ==  
\\\"none\\\")? \\\\\"\\\":\\\"none\\\"}<\\script>";  
    }  
  
    static String openArticle(){  
        return "<script language=\\\"JavaScript\\\">function  
openArticle(url) "+  
        "{var kk = window.open(url,  
\\\"Regiononline\\\",\\\"resizable=yes,menubar=yes,scrollbars=yes,width=  
640,height=400\\\"};}<\\script>";  
    }  
  
    static String appendArticleMain(String contentServer,  
        String db, String SectionID, String ArticleID){  
        return "<div ID=\\\""+ ArticleID + "\\\""  
style=\\\"display:none\\\">"  
        + "<script language=\\\"JavaScript\\\" "  
        + "src=\\\""+ contentServer //req.getParameter("Server")  
        + "?db=" + db
```

```
+ "&SectionID=" +..SectionID  
+ "&ArticleID=" + ArticleID.substring(1)  
+ "&PartID=2\\\"><\\script>"  
    .+ endDiv;  
}  
}  
  
interface Constants{  
    static final String apostrophe = "\\'\\\'"; //used in document.write()  
    static final String doubleQuote = "\\\"\\\""; //used in document.write()  
    static final String backSlash = "\\\\\\"; //used in document.write()  
()  
    static final String beginScript = "<script language=\"javascript\">";  
    static final String endScript = "<\\script>";  
    //static final String beginDiv = "<div  
    static final String endDiv = "</div>";  
    static final String newline = "\r\n";  
}
```

Appendix B

```

package com.rnci.products.PublishNow;

import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.util.*;
import java.sql.*;
import java.net.*;
import com.rnci.products.DataModules.ePress;
import com.rnci.products.DataModules.HttpServletDb;

public class jSyndicate extends HttpServletDb {

    private static String controlDb = "rolcontrol";
    private static String controlServer = "stoudt";
    private static String defaultURL =
"http://jSyndicate.rnci.com/servlet/com.rnci.products.PublishNow.jContent";
    private static String defaultServer = "stoudt";
    private static String defaultDb = "epublish";
    private static String defaultSectionID = "bnews";
    private static String defaultDocumentURL = "http://www.lehighvalleynow.com/rol/template4.asp";

    private static int itemsToKeep = 200, minutesToLive = 3; //may be set by init() later
    private static long lastModified;
    private static Vector contents = new Vector();
    private Sentinel sentinel = new Sentinel(itemsToKeep, minutesToLive);

    public void init(ServletConfig config) throws ServletException {
        super.init(config);
        if (getInitParameter("itemsToKeep") != null) {
            try {
                this.itemsToKeep = Integer.parseInt(getInitParameter("itemsToKeep"));
            } catch (NumberFormatException e) {
                this.itemsToKeep = 100;
            }
        } else this.itemsToKeep = 100;

        if (getInitParameter("minutesToLive") != null) {
            try {
                this.minutesToLive = Integer.parseInt(getInitParameter("minutesToLive"));
            } catch (NumberFormatException e) {
                this.minutesToLive = 5;
            }
        } else this.minutesToLive = 5;
    }

    public void destroy() {
        sentinel.quit = true;
        contents = null;
    }

    //Process the HTTP Get request
    public void doGet(HttpServletRequest req, HttpServletResponse res) throws ServletException,
IOException {

        String HTMLtext, overRide, SectionID, server, db, templateURL, refererURL, documentURL, redirectURL,
unit, contentID, newWindow;
        int position, ArticleID, PartID, mode, number;
        boolean newContent = false, overrideDefaults = true;
        Content cont;
    }
}

```

```
refererURL = req.getHeader("REFERER");

// default is to use URL params
overRide = req.getParameter("overrideDefaults");
if (overRide != null) {
    if (overRide.startsWith("f")) overrideDefaults = false;
}

//Document URL
documentURL = req.getParameter("documentURL");

else {
    //if (unit.length() < 3) unit = "<p>";
    position = unit.lastIndexOf(",");
    if (position >= 0) unit = (overrideDefaults) ? unit.substring(0, position) : unit.substring(position + 1);
}

//The number of teaser delimiters to count
String numbers = req.getParameter("number");
if (numbers == null) numbers = "4";
else {
    position = numbers.lastIndexOf(",");
    if (position >= 0) numbers = (overrideDefaults) ? numbers.substring(0, position) : numbers.substring(position + 1);
}
try { number = Integer.parseInt(numbers); } catch (Exception e) { number = 4; }

Connection cnn = null;
contentID = newWindow + "," + server + "," + db + "," + SectionID + "," + ArticleID + "," + mode +
"," + unit + "," + number + "," + templateURL;

String validated = validateRequest(templateURL, documentURL, refererURL, db, SectionID);
if (validated.equals("R")) {
    if (documentURL.indexOf("?") != -1) {
        redirectURL = documentURL + "&redirected=yes";
    } else {
        redirectURL = documentURL + "?redirected=yes";
    }
    HTMLtext = " location.href = \"\" + redirectURL + '\"';"
}
else if (validated.equals("E")) {
    HTMLtext = " document.write('<BR><B>A network error occurred in connecting to the database.  
Please try again.</B>'); ";
}
else if (validated.equals("N")) {
    HTMLtext = " document.write('<BR><B>This web site is not authorized to display the content  
requested.</B>'); ";
}
else {
    cont = lookup(contentID);
    if (cont != null) {
        HTMLtext = cont.detail;
    } else {
        newContent = true;

        if ((SectionID != null) && (db != null) && (server != null)) {
            try {
                cnn = dataConnMan.getConnection(server, db);
                ePress thisOne = new ePress(cnn);

                thisOne.setLink(templateURL);
                thisOne.setDb(db);
            }
        }
    }
}
```

```

    if ((unit != null) && (number > 0)) {      //getTeaser
        HTMLtext = thisOne.getTeasers(SectionID, ArticleID, PartID, mode, unit, number);
    }
    else {                                     //getContent
        HTMLtext = thisOne.getContent(SectionID, ArticleID, PartID, mode);
    }
    HTMLtext = swapStrings(HTMLtext, "&db=", "&server=" + server + "&templateURL=" +
URLEncoder.encode(templateURL) + "&db=");
    if (newWindow.toLowerCase().startsWith("y")) { // add the "target=_blank" to article Hrefs
        HTMLtext = makeArticleHrefs(HTMLtext);
    }
    dataConnMan.freeConnection(server, db, cnn);
}
catch (Exception e) {
    HTMLtext = "<!-- Error connecting to database: " + e.toString() + "-->" + server + "<-->";
}
} else {
    HTMLtext = "<!-- The SectionID, server, and db parameters are required. --->";
}
HTMLtext = convertTojs(HTMLtext, documentURL, refererURL);
}

if (documentURL == null) documentURL = defaultDocumentURL;

//Should part 1 links bring up a new browser window?
newWindow = req.getParameter("newWindow");
if (newWindow == null) newWindow = "no";
position = newWindow.lastIndexOf(",");
if (position >= 0) newWindow = (overrideDefaults)? newWindow.substring(0, position):
newWindow.substring(position + 1);

//The Requested SectionID
SectionID = req.getParameter("SectionID");
if (SectionID == null) SectionID = defaultSectionID;
position = SectionID.lastIndexOf(",");
if (position >= 0) SectionID = (overrideDefaults)? SectionID.substring(0, position):
SectionID.substring(position + 1);

//The Requested ArticleID (0 for most recent article)
String ArticleIDs = req.getParameter("ArticleID");
if (ArticleIDs == null) ArticleIDs = "0";
position = ArticleIDs.lastIndexOf(",");
if (position >= 0) {
    if (overrideDefaults) ArticleIDs = ArticleIDs.substring(0, position);
    else ArticleIDs = ArticleIDs.substring(position + 1);
}
try { ArticleID = Integer.parseInt(ArticleIDs); } catch (Exception e) { ArticleID = 0; }

//The Requested Part (1 for title, 2 for main content)
String PartIDs = req.getParameter("PartID");
if (PartIDs == null) PartIDs = "1";
position = PartIDs.lastIndexOf(",");
if (position >= 0) {
    if (overrideDefaults) PartIDs = PartIDs.substring(0, position);
    else PartIDs = PartIDs.substring(position + 1);
}
try { PartID = Integer.parseInt(PartIDs); } catch (Exception e) { PartID = 2; }

//Database Server
server = req.getParameter("server");
if (server == null) server = defaultServer;
position = server.lastIndexOf(",");
if (position >= 0) server = (overrideDefaults)? server.substring(0, position):
server.substring(position + 1);

```

```

//PublishNow Database
db = req.getParameter("db");
if (db == null) db = defaultDb;
position = db.lastIndexOf(",");
if (position >= 0) db = (overrideDefaults)? db.substring(0, position): db.substring(position + 1);

//Forward Link Template URL
templateURL = req.getParameter("templateURL");
if (templateURL == null) templateURL = documentURL; //if no templateURL is supplied, set
templateURL equal to the current document
position = templateURL.lastIndexOf(",");
if (position >= 0) templateURL = (overrideDefaults)? templateURL.substring(0, position):
templateURL.substring(position + 1);
//try { templateURL = URLDecoder.decode(templateURL); }
//catch (Exception e) { }
if (templateURL.indexOf("SectionID=") != -1) { //if article detail page is same as part 1 links
page
    templateURL = templateURL.substring(0,documentURL.indexOf("SectionID=") - 1);
}

//The iCount Parameter in ePress.java
String modes = req.getParameter("mode");
if (modes == null) modes = "1";
position = modes.lastIndexOf(",");
if (position >= 0) {
    if (overrideDefaults) modes = modes.substring(0, position);
    else modes = modes.substring(position + 1);
}
try { mode = Integer.parseInt(modes); } catch (Exception e) { mode = 1; }

//The delimiting string for Teaser requests
unit = req.getParameter("unit");
if (unit == null) {}

}

PrintWriter out = res.getWriter();
res.setContentType("text/html");
//res.setHeader("Cache-Control","no cache"); //only seems to effect images
out.println(HTMLtext);

if (newContent) {
    lastModified = System.currentTimeMillis();
    cont = new Content(contentID, HTMLtext, lastModified);
    contents.addElement(cont);
}
}

private String validateRequest(String templateURL, String documentURL, String refererURL, String db,
String SectionID) {
    //returns "Y", "N", or "E" (connection error)

    String documentHOST = "", SQL = "", refererHOST = "", retchar="N";
    if (refererURL != null) refererHOST = parseHostFromURL(refererURL);
    if (documentURL != null) documentHOST = parseHostFromURL(documentURL);
    if (documentURL != null) {
        SQL = "SELECT authorized_url FROM Syndicated_Content WHERE pubnow_section_name = '" + SectionID +
" AND active_flg = 'Y' AND pubnow_dbname = '" + db + "' AND authorized_url LIKE '%" + documentHOST +
"%' AND datetime_up < getdate() AND datetime_down > getdate();";
        Connection cnn = null;
        try {
            cnn = dataConnMan.getConnection(controlServer,controlDb);
            Statement stmt = cnn.createStatement();
            ResultSet rst = stmt.executeQuery(SQL);
            retchar = "N";
            while (rst.next()) retchar = "Y";
            rst.close();
            stmt.close();
            dataConnMan.freeConnection(controlServer,controlDb,cnn);
        }
    }
}

```

```

        catch (Exception e) {
            retchar = "E";
        }
    }

    return retchar;
}

private String parseHostFromURL(String targetURL) {
    StringTokenizer tokenizer = new StringTokenizer(targetURL, "/");
    int tokencount = tokenizer.countTokens();
    int i;
    String returnHost = "";
    if (tokencount > 1) {
        for (i = 1; i < 3; i++) returnHost = tokenizer.nextToken();
    } else {
        returnHost = "";
    }
    return returnHost;
}

private String makeArticleHrefs(String HTMLtext) {
    int i;
    StringBuffer sb = new StringBuffer();
    for (i = 0; i < HTMLtext.length(); i++) {
        if ((HTMLtext.substring(i,i+1).equals("<"))
            && (HTMLtext.substring(i+1,i+2).equalsIgnoreCase("a"))
            && (HTMLtext.substring(i+3,i+4).equals("H"))
            && (HTMLtext.substring(i+4,i+5).equals("r")))) {
            sb.append("<a target=\"_blank\"");
            i++;
        } else {
            sb.append(HTMLtext.charAt(i));
        }
    }
    return sb.toString();
}

public long getLastModified(HttpServletRequest req) {

    return lastModified/1000*1000;
}

private String convertTojs(String HTMLtext, String documentURL, String refererURL) {
    HTMLtext = HTMLtext.replace('\r', ' ');
    HTMLtext = HTMLtext.replace('\n', ' ');

    try {
        HTMLtext = swapStrings(HTMLtext, "\\", "\\");
        HTMLtext = swapStrings(HTMLtext, "\'", "\\\"");
        HTMLtext = swapStrings(HTMLtext, "\\"", "\\\"");
    }
    catch(Exception e) {
        System.out.println(e.getMessage());
    }

    if (refererURL != null) {
        HTMLtext = " if (location.href.indexOf(\"" + documentURL + "\") != -1) { document.write ('" + HTMLtext + "'); } else { document.write ('<BR><B>This web site is not authorized to display the content requested.</B>'); } ";
    } else {
        HTMLtext = " document.write ('" + HTMLtext + "');");
    }
    return HTMLtext;
}

```

```
String swapStrings(String TextIn, String OldString, String NewString)
{
    int OldStringStartAt;
    int OldStringLength = OldString.length();
    int NewStringLength = NewString.length();
    String TempText = TextIn;
    OldStringStartAt = indexOfIgnoreCase(TempText, OldString, 0);
    while (OldStringStartAt != -1)
    {
        TempText = TempText.substring( 0, OldStringStartAt) +
                    NewString + TempText.substring(OldStringStartAt + OldStringLength);
        OldStringStartAt = indexOfIgnoreCase(TempText,
                                             OldString,
                                             OldStringStartAt + NewStringLength);
    }
    return TempText;
}

int indexOfIgnoreCase(String TempText, String subString,
                     int StartAt)
{
    return TempText.toLowerCase().indexOf( subString.toLowerCase(), StartAt);
}

Content lookup(String contentID) {
    Content cont;
    int index = -1;
    for (int i=0; i<contents.size(); ++i) {
        cont = (Content)contents.elementAt(i);
        if (contentID.equals(cont.contentID)) return (Content)contents.elementAt(i);
    }
    return null;
}

class Content {
    String contentID, detail;
    long timestamp;
    Content(String contentID, String detail, long timestamp) {
        this.contentID = contentID;
        this.detail = detail;
        this.timestamp = timestamp/60000; //converts milliseconds to minutes
    }
}

private class Sentinel extends Thread {
    int itemsToKeep;
    int minutesToLive;
    boolean quit = false;

    Sentinel(int itemsToKeep, int minutesToLive) {
        this.itemsToKeep = itemsToKeep;
        this.minutesToLive = minutesToLive;
        setPriority(Thread.MIN_PRIORITY);
        start();
    }

    public void run() {
        int itemsToRemove;
        while (!quit) { // this keeps cleanup alive until servlet is destroyed
            freshen();
            if (contents.size() > itemsToKeep) trim();
            //System.gc();
            try {
                Thread.sleep(5000);
            }
            catch (InterruptedException ex) { }
            //System.out.println("Sentinel ran at:" + System.currentTimeMillis());
        }
    }
}
```

```
private void trim() {
    int itemsToRemove = contents.size() - itemsToKeep;
    int trimmed = 0;
    for (int i=0; i<itemsToRemove; ++i) {
        contents.removeElementAt(i);
        //System.out.println("Removed: " + contents.elementAt(i).toString());
        Thread.yield();
        trimmed++;
    }
    if (trimmed > 10) contents.trimToSize();
}

private void freshen() {
    long currentTime = System.currentTimeMillis()/60000;
    Content cont;
    for (int i=0; i<contents.size(); ++i) {
        cont = (Content) contents.elementAt(i);
        if (cont.timestamp + minutesToLive < currentTime) {
            contents.removeElementAt(i);
            //System.out.println("Expired: " + cont.contentID);
            Thread.yield();
        }
    }
}

//Get Servlet information
public String getServletInfo() {
    return "Retrieves content from PublishNow 2.x databases.";
}
```

Appendix C

```
/*
 * Title: Active Data Syndicator - ADPExport Servlet
 * Description: Copyright (c) 2001 Active Data Exchange. All rights reserved.
 * Any use without the express written permission of the copyright
 * owner is unlawful.
 * All information contained in the file is confidential to Active Data Exchange
 * Copyright: Active Data Exchange
 * Company: Active Data Exchange
 * Author John E. Wetzel
 * @creation date 10 February 2001
 * @last revision date 13 June 2001
 * @version 3.01
 */

package com.activedatax.products.syndicator.export;

import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.Connection;
import java.util.StringTokenizer;
import java.util.Properties;
import com.activedatax.utils.adxTools;
import java.net.URLEncoder;
import com.activedatax.sql.HttpServletDb;

public class ADPExport extends HttpServletDb {
    private static String sServer;
    private static String sDbName;
    private static String sNoAuthMsg;

    private static final String CONTENT_TYPE = "text/html";

    /**Initialize global variables*/
    public void init(ServletConfig config) throws ServletException {
        super.init(config);
        File f = new File("SyndicatorBG.properties");
        if(!f.exists()) {
            f = new File("I:/SyndicatorBG/SyndicatorBG.properties");
        }
        Properties dbProps = new Properties();
        try {
            InputStream is = new FileInputStream(f);
            dbProps.load(is);
        }
        catch (Exception e) {
            System.err.println("Can't read the properties file from ADPExport servlet. " +
                "Make sure SyndicatorBG.properties is in the CLASSPATH");
            return;
        }
        sServer = dbProps.getProperty("syndicator.database.server");
        sDbName = dbProps.getProperty("syndicator.database.name");
        sNoAuthMsg = dbProps.getProperty("syndicator.adp.noauthmsg");
    }

    /**Process the HTTP Get/Post request*/
    public void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

        String sServerRoot = adxTools.getServerRoot(request);
        String sDocumentURL = "";
        String sTargetURL = "";
        String sOfferContentSetIds = "0";
        String sItemIds = "0";
        String sFullNums = "0";
        String sArcNums = "0";

        long lOfferContentsetId = 0;
        long lSerial = 0;
        long lSubId = 0;
    }
}
```

```

long lItemId = 0;

int nFullNum = 0;
int nArcNum = 0;

boolean bNewWindow = false;
boolean bShowDates = false;

//Serial Number Parameter (required)
try {
    lSerial = Long.parseLong(request.getParameter("j3serial"));
} catch(Exception e) {
    lSerial = 0;
}

//Subscriber Id Parameter (required)
try {
    lSubId = Long.parseLong(request.getParameter("j3sub"));
} catch(Exception e) {
    lSubId = 0;
}

//Offer Content Set Id Parameter (required)
sOfferContentSetIds = request.getParameter("j3ocset");
if (sOfferContentSetIds == null) sOfferContentSetIds = "0";

//Document URL Parameter (required)
sDocumentURL = request.getParameter("j3durl");
if (sDocumentURL == null) sDocumentURL = "";
sDocumentURL =
ADPExporter.removeParamsFromURL(sDocumentURL,"j3serial,j3sub,j3ocset,j3durl,j3itemid,j3fullnum,j3
arcnum,j3turl,j3newwindow,j3showdates,j3rand");

//Target URL for Archive Links Parameter (optional)
sTargetURL = request.getParameter("j3turl");
if (sTargetURL == null) sTargetURL = sDocumentURL; //defaults to the requesting document

//Item Id Parameter per Content Set (optional)
sItemIds = request.getParameter("j3itemid");
if (sItemIds == null) sItemIds = "0";

//Number of Full Articles to Display per Content Set Parameter (optional)
sFullNums = request.getParameter("j3fullnum");
if (sFullNums == null) sFullNums = "1"; //default to 1

//Number of Article Archive Links to Display per Content Set Parameter (optional)
sArcNums = request.getParameter("j3arcnum");
if (sArcNums == null) sArcNums = "0"; //default to none

//Popup new window when an archive link is clicked? True or False (optional)
try {
    bNewWindow = (new Boolean(request.getParameter("j3newwindow"))).booleanValue();
} catch(Exception e) {
    bNewWindow = false; //default to false
}

//Show dates along with archive links? True or False (optional)
try {
    bShowDates = (new Boolean(request.getParameter("j3showdates"))).booleanValue();
} catch(Exception e) {
    bShowDates = false; //default to false
}

response.setContentType(CONTENT_TYPE);
PrintWriter out = response.getWriter();

//Expecting these tokenizers to have the same number of tokens.
StringTokenizer tokenizer = new StringTokenizer(sOfferContentSetIds,":");
StringTokenizer tFull = new StringTokenizer(sFullNums,":");
StringTokenizer tArc = new StringTokenizer(sArcNums,":");

```

```

 StringTokenizer tItems = new StringTokenizer(sItemIds,":");
 if(tItems.countTokens() != tokenizer.countTokens()) {
     sItemIds = "0";
     for(int a=1;a<tokenizer.countTokens();a++) {
         sItemIds = sItemIds + ":" + "0";
     }
     tItems = new StringTokenizer(sItemIds,":");
 }

 Connection cnn = dataConnMan.getConnection(this.sServer,this.sDbName);
 ADPExporter ADP = new ADPExporter(cnn, sServerRoot, lSerial, lSubId, 0, 0, sDocumentURL,
 sTargetURL, 1, 0, bNewWindow, bShowDates, request.getRemoteAddr());

 if(tokenizer.countTokens() > 1) {
     ADP.setAtomicQueryString(sItemIds,"&j3ocset=" + URLEncoder.encode(sOfferContentSetIds) +
     "&j3arcnum=" + URLEncoder.encode(sArcNums) + "&j3fullnum=" + URLEncoder.encode(sFullNums));
 }

 int nTk = 0;
 while(tokenizer.hasMoreTokens()) {
     try {
         lOfferContentsetId = Long.parseLong(tokenizer.nextToken());
         try { nFullNum = Integer.parseInt(tFull.nextToken()); } catch(Exception ex) { nFullNum =
 1; }
         try { nArcNum = Integer.parseInt(tArc.nextToken()); } catch(Exception ex) { nArcNum =
 0; }
         try { lItemId = Long.parseLong(tItems.nextToken()); } catch(Exception ex) { lItemId =
 0; }

         ADP.setOfferContentsetId(lOfferContentsetId);
         ADP.setFullNum(nFullNum);
         ADP.setArcNum(nArcNum);
         ADP.setItemId(lItemId);
         ADP.setAtomicPosition(nTk);

         if(ADP.validateADPRequest()) {
             out.println(ADP.getADPContent());
             out.println("document.write('<P>');");
         } else {
             out.println("document.write('" + this.sNoAuthMsg + "');");
         }
     } catch(Exception e) {
         out.println("document.write('<!---- An error occurred while parsing offer content set
ids. --->');");
     }
     nTk++;
 }

 dataConnMan.freeConnection(this.sServer,this.sDbName,cnn);
 ADP = null;
 out.flush();
 out.close();
}

public void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    doGet(request, response);
}

/**Clean up resources*/
public void destroy() {
}
}

```

```

/**
 * Title:      Active Data Syndicator - ADPExporter Class
 * Description: Copyright (c) 2001 Active Data Exchange. All rights reserved.
 * Any use without the express written permission of the copyright
 * owner is unlawful.
 * All information contained in the file is confidential to Active Data Exchange
 * Copyright:   Active Data Exchange
 * Company:    Active Data Exchange
 * @author     John E. Wetzel
 * @date       10 February 2001
 * @last revision date 21 June 2001
 * @version    3.01
 */

package com.activedatax.products.syndicator.export;

import com.activedatax.utils.adxURLParser;
import java.util.StringTokenizer;
import java.net.MalformedURLException;
//import com.imaginary.lwp.Identifier;
import com.activedatax.products.syndicator.content.OutputType;

import java.io.OutputStream;
import java.net.URLEncoder;
import java.io.FileOutputStream;
import java.io.IOException;
import java.util.List;
import java.rmi.RemoteException;
import java.util.Iterator;
import java.util.Date;
import java.util.Collection;
import java.util.StringTokenizer;
import java.util.Hashtable;
import java.sql.*;
import java.text.SimpleDateFormat;
import javax.servlet.http.HttpServletResponse;
import com.activedatax.utils.DigraphEncrypt;
import com.activedatax.utils.adxTools;

import gnu.regexp.*;
/*
 * This class is a helper class for the ADPExport servlet.
 */
class ADPExporter {

    // class variables/objects
    private Hashtable hItems;
    private Hashtable hItemContent;
    private Connection cnn;
    private String sServerRoot, sDocumentURL, sTargetURL, sAtomicQueryString, sIPAddress,
    sItemTokens;
    private long lSerial, lSubId, lOfferContentSetId, lContentSetId, lItemId;
    private int nPos;
    private boolean bNewWindow, bShowDates;
    int nFullNum, nArcNum, nItemHashSize;

    ADPExporter(Connection pCnn, String psServerRoot, long plSerial, long plSubId, long plItemId,
               long plOfferContentSetId, String psDocumentURL, String psTargetURL,
               int pnFullNum, int pnArcNum, boolean pbNewWindow, boolean pbShowDates,
               String psIPAddress) {

        // init class variables/objects
        this.cnn = pCnn;
        this.lSerial = plSerial;
        this.lSubId = plSubId;
        this.lItemId = plItemId;
        this.sServerRoot = psServerRoot;
        this.lOfferContentSetId = plOfferContentSetId;
        this.lContentSetId = this.lOfferContentSetId;
        this.sDocumentURL = psDocumentURL;
    }
}

```

```
this.sTargetURL = psTargetURL;
this.nFullNum = pnFullNum;
this.nArcNum = pnArcNum;
this.bNewWindow = pbNewWindow;
this.bShowDates = pbShowDates;
this.sIPAddress = psIPAddress;
this.sAtomicQueryString = "";
this.sItemTokens = "";
this.nPos = -1;
}

public void setFullNum(int pnFullNum) {
    this.nFullNum = pnFullNum;
}

public void setArcNum(int pnArcNum) {
    this.nArcNum = pnArcNum;
}

public void setItemId(long plItemId) {
    this.lItemId = plItemId;
}

public void setOfferContentSetId(long plOfferContentSetId) {
    this.lOfferContentSetId = plOfferContentSetId;
}

public void setAtomicQueryString(String psItemIds, String psAtomicQueryString) {
    this.sAtomicQueryString = psAtomicQueryString;
    this.sItemTokens = psItemIds;
}

public void setAtomicPosition(int pnPos) {
    this.nPos = pnPos;
}

/**
 * This method will check to see if the web site making the content request is
 * authorized to receive the content.
 */
public boolean validateADPRequest() {
    String sSQL = "";
    String sURL = parseHostFromURL(this.sDocumentURL);
    boolean bAuthorized = false;
    PreparedStatement stmt;
    ResultSet rs;

    //A non-subscriber is requesting the content.
    //The value in pOfferContentSetId refers to the "content_set_id" column of the content_set
    :able.
    if((this.lSerial == -1) && (this.lSubId == -1)) {
        this.lContentSetId = this.lOfferContentSetId;
        sSQL = "SELECT DISTINCT url.url_id, url.crt_class FROM url, content_set "
            + "WHERE content_set.content_set_id = ? "
            + "AND UPPER(url.url) LIKE ? "
            + "AND url.url_id NOT IN (SELECT DISTINCT url_id FROM subscriber_url) "
            + "AND url.site_id = content_set.site_id ";
        try {
            stmt = cnn.prepareStatement(sSQL);
            stmt.clearParameters();
            stmt.setLong(1, this.lContentSetId);
            stmt.setString(2, "%" + sURL.toUpperCase() + "%");
            stmt.setMaxRows(1);
            rs = stmt.executeQuery();
            while(rs.next()) {
                bAuthorized = true;
            }
            rs.close();
            stmt.close();
        } catch(Exception e) {
```

```

        e.printStackTrace();
    }

    //A subscriber is requesting the content.
} else {
    sSQL = "SELECT DISTINCT url.url_id, url.crt_class FROM url, offer_content_set, "
    + "subscription_subdetails, subscriber, subscriber_url "
    + "WHERE (offer_content_set.offer_content_set_id = ? "
    + "AND subscription_subdetails.serial_no = ? "
    + "AND subscriber.subscriber_id = ? "
    + "AND subscriber_url.subscriber_id = ? "
    + "AND url.url like ?) "
    + "AND offer_content_set.offer_id = subscription_subdetails.offer_id "
    + "AND subscription_subdetails.subscriber_id = subscriber.subscriber_id "
    + "AND subscriber.subscriber_id = subscriber_url.subscriber_id "
    + "AND subscriber_url.url_id = url.url_id";
    try {
        stmt = cnn.prepareStatement(sSQL);
        stmt.clearParameters();
        stmt.setLong(1,this.lOfferContentSetId);
        stmt.setLong(2,this.lSerial);
        stmt.setLong(3,this.lSubId);
        stmt.setLong(4,this.lSubId);
        stmt.setString(5,"%"+sURL.toUpperCase()+"%");
        stmt.setMaxRows(1);
        rs = stmt.executeQuery();
        while(rs.next()) {
            bAuthorized = true;
        }
        rs.close();
        stmt.close();
    } catch(Exception e) {
        e.printStackTrace();
    }
}
return bAuthorized;
}

/**
 * This method retrieves the requested content as defined by this classes' parameters.
 */
private void getItemHashData() {
    this.nItemHashSize = Math.abs(this.nFullNum) + Math.abs(this.nArcNum);
    this.hItems = new Hashtable(this.nItemHashSize);
    this.hItemContent = new Hashtable(this.nItemHashSize);
    PreparedStatement stmt;
    ResultSet rs;
    String sSQL = "", sItemReview = "N", sItemIds = "";
    String[] sTemp = {"","","","","","","","","","",""};
    long lThisItemId;
    int nAllowedArcNum = 0;
    int nFullCount = 0;
    int nResultCount = 0;

    if(!((this.lSerial == -1) && (this.lSubId == -1))) { // A subscriber has requested the
content.
        //get the necessary content_set_id with an additional query
        sSQL = "SELECT offer_content_set.content_set_id, subdetails_cs.archives,
subdetails_cs.item_review "
        + "FROM offer_content_set, subdetails_cs "
        + "WHERE subdetails_cs.serial_no = ? "
        + "AND subdetails_cs.offer_content_set_id = ? "
        + "AND subdetails_cs.output_type_id = ? "
        + "AND subdetails_cs.offer_content_set_id = offer_content_set.offer_content_set_id";

        try {
            stmt = cnn.prepareStatement(sSQL);
            stmt.clearParameters();
            stmt.setLong(1,this.lSerial);

```

```

stmt.setLong(2, this.lOfferContentSetId);
stmt.setInt(3, OutputType.ADP);
stmt.setMaxRows(1);
rs = stmt.executeQuery();
while(rs.next()) {
    this.lContentSetId = rs.getLong(1);
    nAllowedArcNum = rs.getInt(2);
    sItemReview = rs.getString(3);
}
rs.close();
stmt.close();
} catch(Exception e) {
    System.err.println("An ERROR occurred in ADPExporter ...");
    e.printStackTrace();
    return;
}

//make sure the subscriber hasn't tried to request more archives than allowed
if(nAllowedArcNum < this.nItemHashSize) this.nItemHashSize = nAllowedArcNum;
}

if(sItemReview.equals("Y")) {
    return; //will need to code a block here when item_review feature is added
}

String sCsId = "AND item_content_set.content_set_id = ? ";
if(this.lContentSetId == -1) sCsId = "AND item.item_id = ? ";

//now get the content set items (not including the actual content column)
sSQL = "SELECT item.item_id, item.title, item.subtitle, item.author, item.byline, "
+ "item.link_text, item.meta_data, item.comments, item.inctitle, item.last_updated_on "
+ "FROM item, item_content_set "
+ "WHERE UPPER(item.status) != ? AND item.timeup <= ? "
+ "AND item.timedown >= ? "
+ "AND item.item_id = item_content_set.item_id "
+ sCsId
+ "ORDER BY item.timeup DESC ";

try {
    stmt = cnn.prepareStatement(sSQL);
    stmt.clearParameters();
    stmt.setString(1, "D");
    stmt.setTimestamp(2, new Timestamp((new Date()).getTime()));
    stmt.setTimestamp(3, new Timestamp((new Date()).getTime()));
    if(this.lContentSetId > -1)
        stmt.setLong(4, this.lContentSetId);
    else
        stmt.setLong(4, this.lItemId);
    stmt.setMaxRows(this.nItemHashSize);
    rs = stmt.executeQuery();
    while(rs.next()) {
        lThisItemId = rs.getLong(1);
        System.out.println("FOUND ITEM ID: " + lThisItemId);
        if(this.nFullNum > nFullCount) {
            if(nFullCount > 0) sItemIds = sItemIds + ",";
            sItemIds = sItemIds + String.valueOf(lThisItemId);
            nFullCount++;
        }
        sTemp[0] = String.valueOf(lThisItemId); //item_id
        sTemp[1] = rs.getString(2); //title
        sTemp[2] = rs.getString(3); //subtitle
        if(sTemp[2] == null) sTemp[2] = "";
        sTemp[3] = rs.getString(4); //author
        if(sTemp[3] == null) sTemp[3] = "";
        sTemp[4] = rs.getString(5); //byline
        if(sTemp[4] == null) sTemp[4] = "";
        sTemp[5] = rs.getString(6); //linktext
        if(sTemp[5] == null) sTemp[5] = "";
        sTemp[6] = rs.getString(7); //metadata
    }
}

```

```

        if(sTemp[6] == null) sTemp[6] = "";
        sTemp[7] = rs.getString(8); //comments
        if(sTemp[7] == null) sTemp[7] = "";
        sTemp[8] = rs.getString(9); //inctitle
        sTemp[9] = rs.getTimestamp(10).toString(); //displaydate

        this.hItems.put(String.valueOf(nResultCount), sTemp.clone());
        nResultCount++;
    }
    rs.close();
    stmt.close();
} catch(Exception e) {
    System.err.println("An ERROR occurred in ADPExporter ...");
    e.printStackTrace();
}

//finally get nFullNum number of item "content" fields or a specific item
if((sItemIds.length() > 0) || (this.lItemId > 0)) {
    if(this.lItemId > 0) { //a specific item link was clicked on
        sSQL = "SELECT item_id, content FROM item WHERE item_id =" +
String.valueOf(this.lItemId);
    } else {
        sSQL = "SELECT item_id, content FROM item WHERE item_id IN(" + sItemIds + ")";
    }
    try {
        stmt = cnn.prepareStatement(sSQL);
        stmt.clearParameters();
        rs = stmt.executeQuery();
        while(rs.next()) {
            this.hItemContent.put(new Long(rs.getLong(1)), rs.getString(2));
        }
        rs.close();
        stmt.close();
    } catch(Exception e) {
        System.err.println("An ERROR occurred in ADPExporter ...");
        e.printStackTrace();
    }
}

public String getADPContent() {
    getItemHashData();
    String sURL = (this.sTargetURL.indexOf("?") == -1) ? "?" : "&";
    String sNewWindow = (this.bNewWindow) ? " TARGET=_new\" " : "";
    String sLink, sContent, sDisplayDate, sItemIdParameter;
    StringBuffer cb = new StringBuffer(""); //content buffer
    StringBuffer lb = new StringBuffer(""); //link buffer
    String[] sTemp;
    int nFull = this.hItemContent.size();
    int nItems = this.hItems.size();

    boolean bNegativeArcNum = (this.nArcNum < 0);

    //
j3serial,j3sub,j3ocset,j3durl,j3itemid,j3fullnum,j3arcnum,j3turl,j3newwindow,j3showdates,j3rand
    if(this.sAtomicQueryString.length() > 0) {
        sURL += "j3serial=" + URLEncoder.encode(String.valueOf(this.lSerial)) + "&j3sub=" +
URLEncoder.encode(String.valueOf(this.lSubId)) +
        this.sAtomicQueryString + "&j3newwindow=" +
URLEncoder.encode(String.valueOf(this.bNewWindow)) +
        "&j3showdates=" + URLEncoder.encode(String.valueOf(this.bShowDates));

    } else {
        sURL += "j3serial=" + URLEncoder.encode(String.valueOf(this.lSerial)) + "&j3sub=" +
URLEncoder.encode(String.valueOf(this.lSubId)) +
        "&j3ocset=" + URLEncoder.encode(String.valueOf(this.lOfferContentSetId)) + "&j3arcnum=" +
        URLEncoder.encode(String.valueOf(this.nArcNum)) + "&j3newwindow=" +
URLEncoder.encode(String.valueOf(this.bNewWindow)) +

```

```

    "&j3showdates=" + URLEncoder.encode(String.valueOf(this.bShowDates));
}

for (int i=0; i<nItems; i++) {
    sTemp = (String[])this.hItems.get(String.valueOf(i));
    sContent = (String)this.hItemContent.get(new Long(sTemp[0]));
    if(sContent != null) {
        sContent = replaceItemBinaryRefs(sContent);
        //sContent = adxTools.swapStrings(sContent, "\u00f3\u00f3", this.sServerRoot +
        "com.activedatax.products.syndicator.export.ADPRetrieveItemBinary?binary_id=");
    } else {
        sContent = "";
    }
    sLink = (sTemp[5].trim().length() > 0) ? sTemp[5].trim() : sTemp[1].trim();
    sDisplayDate = sTemp[9];
    if(this.bShowDates || bNegativeArcNum) {
        sDisplayDate = sDisplayDate.substring(5,7) + "/" + sDisplayDate.substring(8,10) + "/" +
        sDisplayDate.substring(2,4);
        sLink += "(" + sDisplayDate + ")";
    }

    //sTemp[0] = item_id
    //sTemp[1] = title
    //sTemp[2] = subtitle
    //sTemp[3] = author
    //sTemp[4] = byline
    //sTemp[5] = linktext
    //sTemp[6] = metadata
    //sTemp[7] = comments
    //sTemp[8] = inctitle
    //sTemp[9] = displaydate

    sItemIdParameter = sTemp[0]; //the current item_id
    if(this.sAtomicQueryString.length() > 0) {
        StringTokenizer tItems = new StringTokenizer(this.sItemTokens, ":");
        if(this.nPos == 0) {
            sItemIdParameter = sTemp[0];
            tItems.nextToken();
        } else {
            sItemIdParameter = tItems.nextToken();
        }
        int nq = 1;
        while(tItems.hasMoreTokens()) {
            if(this.nPos == nq) {
                sItemIdParameter += ":" + sTemp[0];
                tItems.nextToken();
            } else {
                sItemIdParameter += ":" + tItems.nextToken();
            }
            nq++;
        }
    }

    if(sContent.length() > 0) {
        cb.append(sContent + "<P>");
        if(Math.abs(this.nArcNum) > 0)
            if((!this.bNewWindow) || ((this.bNewWindow) && (this.lItemId==0))) lb.append(sLink +
            "<BR>");
        } else {
            if(Math.abs(this.nArcNum) > 0)
                if((!this.bNewWindow) || ((this.bNewWindow) && (this.lItemId==0))) lb.append("<A HREF=\"" + this.sTargetURL + sURL + "&j3itemid=" + URLEncoder.encode(sItemIdParameter) + "\" " + sNewWindow + ">" + sLink + "</A><BR>");
        }
    } // end for
    return parseContent(cb.toString() + "<P>" + lb.toString() + "<P>", "");
}

private String replaceItemBinaryRefs(String psContent) {
    StringBuffer sb = new StringBuffer(psContent.length() + 300);

```

```

String sId;
int nPos = 0;
int nLoc = psContent.indexOf("\u00f3\u00f3");
while(nLoc != -1) {
    sb.append(psContent.substring(nPos,nLoc));
    sId = adxTools.convertStringToHexString(DigraphEncrypt.encryptDiGraph(this.sIPAddress +
"TFB" + psContent.substring(nLoc+2,nLoc+22)).trim());
    sb.append(this.sServerRoot +
"com.activatedatax.products.syndicator.export.ADPRetrieveItemBinary?binary_id=" + sId);
    nPos = nLoc + 22;
    nLoc = psContent.indexOf("\u00f3\u00f3",nPos);
}
sb.append(psContent.substring(nPos));
return sb.toString();
}

/**
* This method will remove specified querystring parameters from a supplied URL
* and return the resulting URL
*
* Example: For the method parameters
* psURL = "http://www.mysite.com/page.asp?param1=x&param2=y&param3=z"
* psQueryStringParams = "param1,param3"
*
* The returned URL will be:
* "http://www.mysite.com/page.asp?param2=y"
*/
public static String removeParamsFromURL(String psURL, String psQueryStringParams) {
//System.out.println(URL);
String sTempURL = psURL;
 StringTokenizer tokenizer = new StringTokenizer(psQueryStringParams, ", ");
adxURLParser myParser = new adxURLParser();
try {
    while(tokenizer.hasMoreTokens()) {
        myParser.setOriginalURL(sTempURL);
        myParser.setParamToRemove(tokenizer.nextToken());
        sTempURL = myParser.getRemovedParamURL();
    }
} catch (MalformedURLException mue) {
    mue.printStackTrace();
    //return an error string that will need to be handled by the calling method.
    sTempURL = "error";
}
return sTempURL;
}

/**
* This method will parse out and return the host/domain name from a supplied URL
*
* Example: For the URL:
* psURL = "http://www.mysite.com/page.asp?param1=x&param2=y&param3=z"
*
* The returned String will be:
* "www.mysite.com"
*/
public static String parseHostFromURL(String psURL) {
    StringTokenizer tokenizer = new StringTokenizer(psURL, "/");
    int tokencount = tokenizer.countTokens();
    int i;
    String sReturnHost = "";
    try {
        if (tokencount > 1) {
            for (i = 1; i < 3; i++) sReturnHost = tokenizer.nextToken();
        } else {
            sReturnHost = "error";
        }
    } catch(Exception e) {
}
}

```

```

    //return an error string that will need to be handled by the calling method.
    sReturnHost = "error";
}
return sReturnHost;
}

public static int indexOfIgnoreCase(String sTempText, String sSubString, int nStartAt) {
    return sTempText.toLowerCase().indexOf( sSubString.toLowerCase(), nStartAt);
}

/*
public static String swapStrings(String sTextIn, String sOldString, String sNewString) {
    int nOldStringStartAt;
    int nOldStringLength = sOldString.length();
    int nNewStringLength = sNewString.length();
    String sTempText = sTextIn;
    nOldStringStartAt = indexOfIgnoreCase(sTempText, sOldString, 0);
    while (nOldStringStartAt != -1) {
        sTempText = sTempText.substring( 0, nOldStringStartAt) + sNewString +
sTempText.substring(nOldStringStartAt + nOldStringLength);
        nOldStringStartAt = indexOfIgnoreCase(sTempText, sOldString, nOldStringStartAt +
nNewStringLength);
    }
    return sTempText;
}
*/

public static String parseContent(String sContent, String sServerRoot) {
    String sClientScripts = "";

    int nStartSearchAt = 0;
    int nScriptLoc = indexOfIgnoreCase(sContent, "<SCRIPT", nStartSearchAt);
    if(nScriptLoc > -1) { //Check for client scripts and separate them from regular HTML
        try {
            int nEndScriptLoc;
            int nArticleLength = sContent.length();
            StringBuffer sbArticle = new StringBuffer(nArticleLength + 1);
            StringBuffer sbScripts = new StringBuffer();
            while(nScriptLoc > -1) {
                nEndScriptLoc = indexOfIgnoreCase(sContent, "</SCRIPT>", nScriptLoc);
                sbArticle.append(sContent.substring(nStartSearchAt,nScriptLoc));

                sbScripts.append(sContent.substring((indexOfIgnoreCase(sContent, ">", nScriptLoc)+1),nEndScriptLoc));
            };
            sbScripts.append('\n');
            sbScripts.append('\r');
            //System.out.println("The HTML: " + sContent.substring(nStartSearchAt,nScriptLoc));
            //System.out.println("The Script: " +
sContent.substring((indexOfIgnoreCase(sContent, ">", nScriptLoc)+1),nEndScriptLoc));

            nStartSearchAt = nEndScriptLoc + 9;
            nScriptLoc = indexOfIgnoreCase(sContent, "<SCRIPT", nStartSearchAt);
        }
        //sbArticle.append(sContent.substring(nStartSearchAt,nArticleLength - 1));
        sbArticle.append(sContent.substring(nStartSearchAt,nArticleLength)); //changed by JW
        //System.out.println(sbArticle.toString());
        //System.out.println(sbScripts.toString());
        sClientScripts = sbScripts.toString(); //contains just the scripts' code
        sContent = sbArticle.toString(); //contains just the article HTML
    }
    catch(Exception e) {
        System.err.println(e.getMessage());
        e.printStackTrace();
    }
}

sContent = sContent.replace('\r', ' ');
sContent = sContent.replace('\n', ' ');
try {
    sContent = adxTools.swapStrings(sContent, "\\", "\\");
}

```

```
sContent = adxTools.swapStrings(sContent, "\\", "\\\"");
sContent = adxTools.swapStrings(sContent, "\", "\\\"");
}
catch(Exception e) {
    System.err.println(e.getMessage());
    e.printStackTrace();
}

return sClientScripts + " document.write ('" + sContent + "'); " + '\r
}

} // end class ADPExporter
```

Appendix D

YOUR CONTENT. ANY PLACE. ANY TIME.

ACTIVE DATA RANDOMIZER User Guide

Active Data Randomizer gives Users the power to keep a Website fresh and inviting to visitors. This convenient, easy-to-use tool facilitates the collecting, grouping, modifying and removal of messages, either graphic or text, that are displayed within a page each a time it is loaded, reloaded or refreshed by the viewer.

This User Guide is an overview of how Active Data Randomizer works and what it does. For a complete demonstration, contact your Active Data Exchange representative.



Active Data Exchange, Inc.
190 Brodhead Road, Suite 300
Bethlehem, PA
610.997.8100
www.activedatax.com

Copyright 2001, Active Data Exchange, Inc. All rights reserved. Duplication, electronic or otherwise, is prohibited without written permission of the publisher.



Table of Contents

Overview.....	3
Step-by-step guide for Active Data Randomizer	3
Getting Started.....	3
Step 1 Site Set Up	4
To Set up a new site.....	4
Step 2 Randomizer Groups	4
Step 3 HTML Blurbs	6
<i>To add another HTML blurb:</i>	7
Step 4 Java Script Code	7
Step 5 Verify or Modify Blurb Groupings	9
Step 6 Rotating Blurbs for Preview.....	10
Step 7 Exit	10

Table of Figures

Figure 1 Site Configuration Screen	3
Figure 2 Add a New Site Configuration.....	4
Figure 3 Add a New Group Name.....	5
Figure 4 New Group Confirmation Message.....	5
Figure 5 New Active Data Randomizer Blurb Group	6
Figure 6 New Active Data Randomizer Blurb HTML Confirmation	7
Figure 7 Active Data Randomizer Blurb Code	8
Figure 8 Sample HTML Editor.....	8
Figure 9 Confirmation and View Message Window	9



YOUR CONTENT. ANY PLACE. ANY TIME.
Active Data Randomizer User Guide

Active Data Randomizer

Overview

A practical tool, Active Data Randomizer gives Users the power to keep their Website viewers attentive to the screen by changing a specific part of a page message each time a viewer refreshes or returns to a previously viewed screen. An easy to use tool that requires no additional hardware or installation, Active Data Randomizer is platform independent and facilitates the grouping and presenting of graphic or text messages on a Website page.

The Active Data Randomizer allows for direct, targeted message updates, without the complexities of publishing completely new pages or documents, from any location with Internet access, at any time. Short, catchy visuals and text can be displayed on a page with little effort on the part of the client after setup is complete. The random display of messages keeps your page continually altering and fresh to viewers as they browse through a site. Using Active Data Randomizer to rotate images or messages will decrease the static look and feel of a Web page.

Step-by-step guide for Active Data Randomizer

Getting Started

The Active Data Randomizer is accessible through a Universal Resource Locator, URL. Open your Internet browser and enter this URL into the proper area:

<http://publisher3.activedatax.com/servlet/com.activedatax.products.Blurb.AdminMenu>

You will be taken to the main screen for Active Data Randomizer.

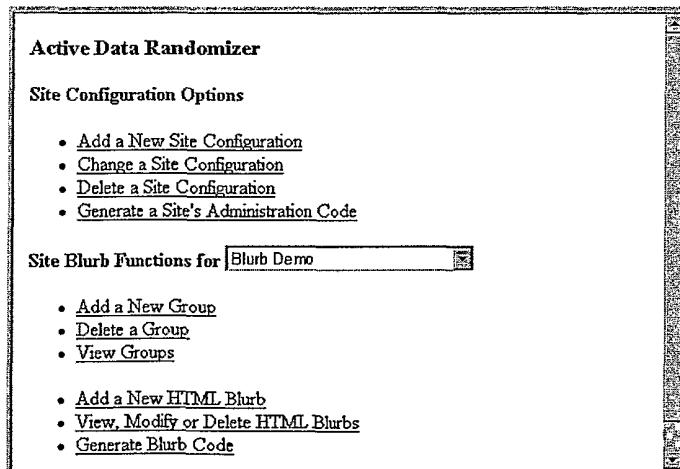


Figure 1 Site Configuration Screen



YOUR CONTENT. ANY PLACE. ANY TIME.
Active Data Randomizer User Guide

The main screen for the Active Data Randomizer has links to each and every functionality offered by this time-saving tool. Once you have accessed the Active Data Randomizer tool, you will see all the site configuration options and functions that are usable for screen refresh messages. Click on any of the bulleted items to access that functionality.

Step 1 Site Set Up

You will arrive at this window when Add a New Site Configuration is chosen from the Site Configuration Screen. From here you are able to set up a new location to display Active Data Randomizer messages or return to the Menu.

Site Name	Blurbdemo
Description	Demonstration of Active Data Randomizer
Site URL	http://asppreview.activedatax.com/blurbdemo

Figure 2 Add a New Site Configuration

To Set up a new site

1. Site Name—enter your new site name.
2. Descriptions—enter a brief explanation.
3. Enter the URL to locate the site and the page where you want your dynamic blurbs to be displayed
4. Select the Submit Configuration button

The new site is automatically configured, and you will be returned to the Administration screen.

NOTE—the success of your new site set-up is confirmed and expressed by the red text in the upper portion of the screen.

Step 2 Randomizer Groups

You will arrive at this window when Add a New Group is chosen from the Site Configuration screen. From here you are able to set up a new group of Active Data Randomizer blurbs or return to the Menu.



YOUR CONTENT. ANY PLACE. ANY TIME.
Active Data Randomizer User Guide

Active Data Randomizer Administration

Add a New Blurb Group to the Site *Blurb Demo*

Group Name	02blurb Test
------------	--------------

[Go Back to Menu](#) [Submit Group](#)

Figure 3 Add a New Group Name

1. Enter a Group Name for your first collection of different blurbs that will be used for a specific location. These are the dynamic blurbs that will be rotated sequentially every time a viewer returns to that page and it is reloaded, or when the viewer selects the refresh icon. The blurbs you want to set up are virtually limitless and can be graphic or textual based. The sample group name in this case is 02blurb test.
2. Click **Submit Group**.

Active Data Randomizer Administration

The new blurb group 02blurb test was added to the Blurb Demo site.

Site Configuration Options

- [Add a New Site Configuration](#)
- [Change a Site Configuration](#)
- [Delete a Site Configuration](#)
- [Generate a Site's Administration Code](#)

Site Blurb Functions for Blurb Demo

- [Add a New Group](#)
- [Delete a Group](#)
- [View Groups](#)
- [Add a New HTML Blurb](#)
- [View, Modify or Delete HTML Blurbs](#)
- [Generate Blurb Code](#)

Figure 4 New Group Confirmation Message

3. You will be returned to the main menu. A confirmation that your group has been created is shown in red text. It repeats the Group Name and the Site Name that it has been added to.
4. To continue click on Add a New HTML Blurb, located in the lower group of bulleted items.



YOUR CONTENT. ANY PLACE. ANY TIME.
Active Data Randomizer User Guide

Step 3 HTML Blurbs

You will arrive at this window when Add a New HTML Blurb is chosen from the New Group Confirmation screen. From here you are able to set up a new location to display Active Data Randomizer messages or return to the Menu.

The screenshot shows a window titled "Active Data Randomizer Administration". Inside, it says "Add a New Blurb to the Site Blurb Demo". A dropdown menu under "Group Name" has three options: "02blurb test", "02blurb test", and "blurb1". Below the dropdown is a large text area labeled "Blurb HTML". At the bottom are two buttons: "Go Back in Menu" and "Submit Blurb".

Figure 5 New Active Data Randomizer Blurb Group

1. From the Add a New randomizer screen, select the Group Name from the pull down menu you want this new HTML blurb added to.
2. Enter or Copy and Paste the HTML code for the picture or text or both that you want to show in the refreshed window. This information is retrieved from your original document's source code from whatever HTML editor program you are using.



YOUR CONTENT. ANY PLACE. ANY TIME.
Active Data Randomizer User Guide

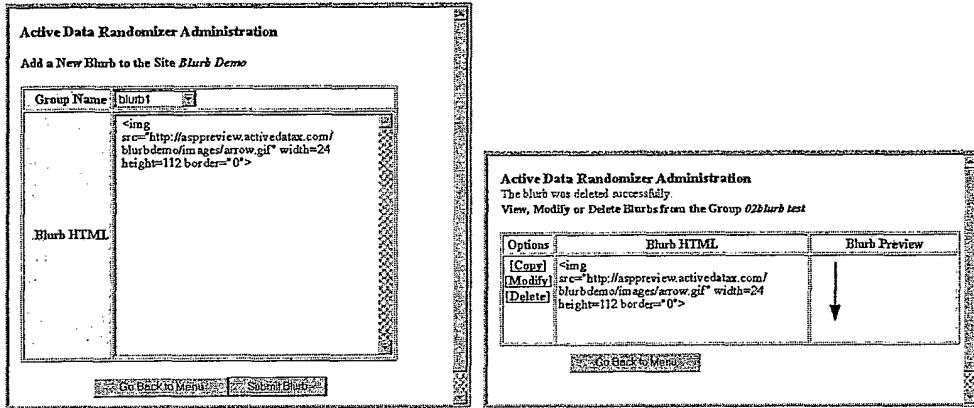


Figure 6 New Active Data Randomizer Blurb HTML Confirmation

3. Click on **Submit Blurb**. A confirmation screen will show that the blurb has been added. You are still able to modify or delete your blurb in the future if necessary. Any action you take at this step will be confirmed within the next window in red text.

To add another HTML blurb:

1. Click on **[Copy]** and Randomizer will automatically duplicate your last HTML blurb.
2. Either insert new HTML text or alter your HTML text, as in the case of changing the image only, you can just update the name of the source image, without having to re-enter all new HTML.
3. Select the **Submit Blurb** button after each entry to save the changes
Each entry will be confirmed and added to your list of HTML blurbs.
4. Repeat this process until you have added all your HTML blurbs.
5. Select **Go Back to Menu**.

Step 4 Java Script Code

Once you have returned to the Site Configuration main screen,

1. Select Generate Blurb Code from the lower level bulleted options.
2. Active Data Blurb Admin will automatically write the Java Script needed to instruct your site to sequentially rotate the HTML sources. This Java will include all necessary instructions to refresh the image in the window each time it is reloaded or manually refreshed.



YOUR CONTENT. ANY PLACE. ANY TIME.
Active Data Randomizer User Guide

Active Data Randomizer Administration

Blurb Machine Code for the Group 02blurb test of the Site Blurb Demo

(copy and paste into target site's HTML)

```
<SCRIPT LANGUAGE="javascript">document.write
(''<SCR> IPT LANGUAGE="javascript"
SRC="http://publisher3.activedatax.com/servlet/com
.activedatax.products.Blurb.JBlurb?
jbsid=860582&jbgid=271221&jbdurl=' + escape
(location.href) + '&jbrand=' + escape(Math.random
()) + '')<'/SCR> IPT'');</SCRIPT>
```

Go Back to Menu

Figure 7 Active Data Randomizer Blurb Code

3. Highlight and copy this Java Script
4. Return to your primary source document using your HTML Editing program and select the HTML source view.

```

<TR><TD WIDTH=640 COLSPAN=3><A href="home.htm"><IMG border=0 height=27 src="images/logoa.gif"
width=239></a><a href="pacnews.htm"><IMG alt="" border=0 height=27 src="images/12pacnews.gif"
width=131></a><a href="portfolionews.htm"><IMG alt="" border=0 height=27
src="images/12portfolionews.gif" width=131></a><IMG border=0 src="images/12news.gif"></TD></TR>

<TR><TD WIDTH=239 VALIGN=top><A href="home.htm"><IMG alt="PA Early Stage" border=0 height=50
src="images/logob.gif" width=239 ></a><BR>

<table width=239 CELLPADDING=0 CELLSPACING=0 BORDER=0><tr><td><td VALIGN=top WIDTH=38><IMG alt=""
border=0 height=27 src="images/fade2.gif" width=38 VALIGN="TOP"></td><td WIDTH=201
VALIGN=top><BR><script LANGUAGE="javascript">document.write('<SCR> IPT LANGUAGE="javascript"
SRC="http://publisher3.activedatax.com/servlet/com.activedatax.products.Blurb.JBlurb?jbsid=321186
&jbgid=116364&jbdurl=' + escape(location.href) + '&jbrand=' + escape(Math.random()) +
'')<'/SCR> IPT'');</script>
<script LANGUAGE="javascrip">document.write('<SCR> IPT LANGUAGE="javascrip"
SRC="http://publisher3.activedatax.com/servlet/com.activedatax.products.Blurb.JBlurb?jbsid=321186
&jbgid=105242&jbdurl=' + escape(location.href) + '&jbrand=' + escape(Math.random()) +
'')<'/SCR> IPT'');</script>
</td></tr></table></td>

<TD WIDTH=10 VALIGN=top><IMG border=0 height=1 src="images/clear.gif" width=10></TD>

<TD VALIGN=top><BR><font face="Arial, Helvetica, sans-serif" size=-1><IMG alt="" border=0
height=27 src="images/hmcv2.gif" width=217></font>
<font color="#2d4491"><a href="pacnews.htm">PA Early Stage News</a></font><font
color="#2d4491">December 4, 2000</font><font color="#2d4491"><a href="pr_ajumlah120400.htm">PA Early Stage
Company, Ajunto, Inc., Form Strategic Alliance with VerticalNet</a></font></font>
<font color="#2d4491">Ajunto, Inc. has been selected as the Preferred IT Consulting Hub for the SJ VerticalNet Market</font>

```

Figure 8 Sample HTML Editor

5. Paste the Active Data Randomizer Java script you have copied into your original source document for the Web page.



YOUR CONTENT. ANY PLACE. ANY TIME.
Active Data Randomizer User Guide

NOTE—Make sure you are within the proper location on your Web page. Paste the script in the exact position where you want the message to be displayed on the page.

6. Exit (or close) your HTML editing program and return to the Active Data Randomizer Administration screen.
7. Click the **Return to Menu** button.

Step 5 Verify or Modify Blurb Groupings

From the main Active Data Randomizer Administration window, you may view your group of messages. Active Data Randomizer presents both the HTML and a visual of the output.

1. Choose the [View, Modify, Delete HTML Blurb](#) to view and confirm your message or messages.

Active Data Randomizer Administration

View, Modify or Delete Blurbs from the Group *In the News*

Options	Blurb HTML	Blurb Preview
[Copy] [Modify] [Delete]		
[Copy] [Modify] [Delete]		
[Copy] [Modify] [Delete]	 Charlie Clark (VerticalNet), Michael Carter (Ajunto), Mary Naylor (VIP Desk), Mike Bolton	 Charlie Clark (VerticalNet), Michael Carter (Ajunto), Mary Naylor (VIP Desk), Mike Bolton

[Go Back to Menu](#)

Figure 9 Confirmation and View Message Window

2. A preview image of the actual image that will be shown on a viewer's screen. Select **Go Back to Menu** to pretest the messages.



YOUR CONTENT. ANY PLACE. ANY TIME.
Active Data Randomizer User Guide

Step 6 Rotating Blurs for Preview

It is best to always pretest the site to make sure your messages are being presented properly.

1. Open any separate browser window and point to the page you want to check.
2. Choose *refresh* to view the new message in the Group you have been assigned to that page. Continue to choose *refresh* or move to another page and return to the original page to verify that your messages have changed. At times it is normal for Active Data Randomizer to repeat a message or image.
3. Exit

Step 7 Exit

Users may exit Active Data Randomizer Administration by simply leaving or closing your browser window.

About Active Data Exchange

Active Data Exchange is a leader in syndication software solutions that empower companies to get the right content to the right place at the right time. Active Data Exchange helps create highly effective information delivery chains with partners, customers, vendors, distributors, investors, and other target groups and affects commerce with communications tools better than existing email technologies. The company is active in several industry standards committees including the Information and Content Exchange Authoring Group and the W3C XML Protocol Standards Committee. Clients include Crown, Cork and Seal, MainStreet Networks, Turner Construction, Penn Mutual Life Insurance, DeSales and Lehigh University.

Appendix E

```

/**
 * Title:      Active Data Blurb Machine<p>
 * Description: Dynamic HTML content generation without the need for a client install.<p>
 * Copyright:  Copyright (c) John E. Wetzel<p>
 * Company:    Active Data Exchange, Inc.<p>
 * @author:    John E. Wetzel
 * @version:   1.0a
 */

package com.activedatax.products.Blurb;

import javax.servlet.*;
import javax.servlet.http.*;
import com.activedatax.products.Blurb.dbmodules.*;
import com.activedatax.products.Blurb.html.*;
import com.activedatax.sql.HttpServletDb;
import com.activedatax.utils.HTMLQuoter;
import java.io.*;
import java.net.*;
import java.util.*;
import java.sql.*;
import gnu-regexp.*;
import java.net.URL;

public class AdminMenu extends HttpServlet {
    private BlurbCommon mycommon;
    private String server = "";
    private String dbname = "";
    private String thisServlet = "";
    private String serverRoot = "";
    private String fLink = "";
    private String fPost = "";

    //Initialize global variables
    public void init(ServletConfig config) throws ServletException {
        super.init(config);

        //USE BlurbCommon.class TO GET PROPERTIES -- SERVER & DATABASE NAME
        mycommon = new BlurbCommon();
        Properties p = mycommon.getProps();
        serverRoot = p.getProperty("server.root");
        server = p.getProperty ("default.server");
        dbname = p.getProperty ("default.dbname");

        //store the alias in
        p = mycommon.getAliases();
        thisServlet = p.getProperty("AdminMenu");

        fLink = serverRoot + thisServlet;
        fPost = serverRoot + thisServlet;
    }

    public void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        String outputHTML = doBlurb(request, response, this.fLink, this.fPost, false);
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        //System.out.println(request.getParameter("jbact"));
        out.println(outputHTML);
        return;
    }

    public void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        String outputHTML = doBlurb(request, response, this.fLink, this.fPost, false);
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println(outputHTML);
        return;
    }
}

```

```

}

public String doBlurb(HttpServletRequest request, HttpServletResponse response, String
forwardLink, String forwardPost, boolean syndicated) throws ServletException, IOException {
    int jbsid, jbgid, jbbid;
    String outputHTML = "";
    String param = "";
    String action = request.getParameter("jbact");

    try {
        jbsid = Integer.parseInt(request.getParameter("jbsid").trim());
    } catch(Exception e) { jbsid = -1; }

    try {
        jbgid = Integer.parseInt(request.getParameter("jbgid").trim());
    } catch(Exception e) { jbgid = -1; }

    try {
        jbbid = Integer.parseInt(request.getParameter("jbbid").trim());
    } catch(Exception e) { jbbid = -1; }

    if(jbsid == -1) { //get the blurb_site_id of the first site in the drop-down list box
        jbsid = getFirstSiteId();
    }

    if ((action == null) || (action.length() == 0)) {
        outputHTML =
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "",syndicat
ed);
    }
    else {
        if (action.equalsIgnoreCase("add")) {
            outputHTML = AdminMenuHTML.siteAddOrModify(forwardLink,-1,"","","http://","");
        } else if (action.equalsIgnoreCase("addpost")) {
            outputHTML = doSiteAdd(request, response, syndicated);
        } else if (action.equalsIgnoreCase("getpostdata")) {
            outputHTML =
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "",syndicat
ed);
        } else if (action.equalsIgnoreCase("addbg")) {
            outputHTML =
AdminMenuHTML.groupAdd(jbsid,-1,forwardLink,forwardPost,"",getSiteNameFromId(jbsid), "");
        } else if (action.equalsIgnoreCase("addbgpost")) {
            outputHTML = doGroupAdd(request, response, forwardLink, forwardPost, syndicated);
        } else if (action.equalsIgnoreCase("addbb")) {
            outputHTML = addBlurb(jbsid,forwardLink,forwardPost,syndicated);
        } else if (action.equalsIgnoreCase("copybb")) {
            outputHTML = copyBlurb(jbsid,jbgid,jbbid,forwardLink,forwardPost,syndicated);
        } else if (action.equalsIgnoreCase("chgb")) {
            outputHTML = changeBlurb(jbsid,jbgid,jbbid,forwardLink,forwardPost,syndicated);
        } else if (action.equalsIgnoreCase("chgbpost2")) {
            outputHTML = doBlurbChange(request, response, forwardLink, forwardPost, syndicated);
        } else if (action.equalsIgnoreCase("addbbpost")) {
            outputHTML = doBlurbAdd(request, response, forwardLink, forwardPost, syndicated);
        } else if (action.equalsIgnoreCase("chg")) {
            outputHTML = changeSiteSelect(syndicated);
        } else if (action.equalsIgnoreCase("gensitecode")) {
            outputHTML = genSiteCodeSelect(syndicated);
        } else if (action.equalsIgnoreCase("gensitecodepost1")) {
            outputHTML =
AdminMenuHTML.generateSiteAdminCode(jbsid,forwardLink,this.serverRoot,getSiteNameFromId(jbsid));
        } else if (action.equalsIgnoreCase("chgbpost1")) {
            outputHTML = changeSiteDetails(request, response, syndicated);
        } else if (action.equalsIgnoreCase("chgbpost2")) {
            outputHTML = doSiteChange(request, response, syndicated);
        } else if (action.equalsIgnoreCase("del")) {
            outputHTML = deleteSiteSelect(syndicated);
        } else if (action.equalsIgnoreCase("delbg")) {
            outputHTML = deleteGroupSelect(jbsid,forwardLink,forwardPost,syndicated);
        }
    }
}

```

```

        } else if (action.equalsIgnoreCase("delbb")) {
            outputHTML = viewGroupBlurbs(jbsid, jbgid, jbbid, forwardLink, forwardPost, "Are you
<B>sure</B> you want to delete the blurb shown?", syndicated);
        } else if (action.equalsIgnoreCase("viewbb")) {
            outputHTML = viewBlurbSelect(jbsid, forwardLink, forwardPost, syndicated);
        } else if (action.equalsIgnoreCase("viewbbpost1")) {
            outputHTML = viewGroupBlurbs(jbsid, jbgid, -1, forwardLink, forwardPost, "", syndicated);
        } else if (action.equalsIgnoreCase("delbgpost1")) {
            outputHTML =
deleteGroupDetails(jbsid, forwardLink, forwardPost, request, response, syndicated);
        } else if (action.equalsIgnoreCase("delbgpost2")) {
            outputHTML = doGroupDelete(jbsid, forwardLink, forwardPost, request, response, syndicated);
        } else if (action.equalsIgnoreCase("delbbpost1")) {
            outputHTML = doBlurbDelete(jbsid, jbgid, jbbid, forwardLink, forwardPost, syndicated);
        } else if (action.equalsIgnoreCase("delpost1")) {
            outputHTML = deleteSiteDetails(request, response, syndicated);
        } else if (action.equalsIgnoreCase("delpost2")) {
            outputHTML = doSiteDelete(request, response, syndicated);
        } else if (action.equalsIgnoreCase("genbcode")) {
            outputHTML = generateBlurbCode(jbsid, forwardLink, forwardPost, syndicated);
        } else if (action.equalsIgnoreCase("viewwbg")) {
            outputHTML = viewGroups(jbsid, forwardLink, forwardPost, syndicated);
        } else if (action.equalsIgnoreCase("genbcodepost1")) {
            outputHTML =
AdminMenuHTML.generateBlurbCode(jbsid, jbgid, forwardLink, this.serverRoot, getSiteNameFromId(jbsid),
getGroupNameFromId(jbgid));
        }
    }

    return outputHTML;
}

private synchronized String doSiteAdd(HttpServletRequest request, HttpServletResponse
response, boolean syndicated) {
    boolean errFlag = false;
    int id = 0;
    int count = 1;
    double tempval;
    Connection conn;
    Vector rs = null;
    BlurbSiteDB bs = new BlurbSiteDB();
    String blurb_site_name = request.getParameter("blurb_site_name").trim();
    String description = request.getParameter("description").trim();
    String url = request.getParameter("url").trim();

    /*
    Test if site, desc, and http:// are valid
    */
    try {
        if (blurb_site_name.trim().equalsIgnoreCase("")) {
            return AdminMenuHTML.siteAddOrModify(this.serverRoot + this.thisServlet, -1,
blurb_site_name, description, url, "An error occurred. An invalid <B>Site Name or Site
Description</B> was entered.");
        }
        if (description.trim().equalsIgnoreCase("")) {
            return AdminMenuHTML.siteAddOrModify(this.serverRoot + this.thisServlet, -1,
blurb_site_name, description, url, "An error occurred. An invalid <B>Site Name or Site
Description</B> was entered.");
        }
    }
    catch(Exception e) {
        BlurbCommon.handleException(e);
        return AdminMenuHTML.siteAddOrModify(this.serverRoot + this.thisServlet, -1,
blurb_site_name, description, url, "An error occurred. An invalid <B>Site Name or Site
Description</B> was entered.");
    }

    /*
    Test for a valid URL
    */
}

```

```

if (url.length() > 9) {
    if (url.substring(0,7).equalsIgnoreCase("http://")) {
    } else if(url.substring(0,8).equalsIgnoreCase("https://")) {
    } else {
        return AdminMenuHTML.siteAddOrModify(this.serverRoot +
this.thisServlet,-1,blurb_site_name,description,url,"An error occurred. An invalid <B>Site
URL</B> was entered.");
    }
} else {
    return AdminMenuHTML.siteAddOrModify(this.serverRoot +
this.thisServlet,-1,blurb_site_name,description,url,"An error occurred. An invalid <B>Site
URL</B> was entered.");
}
if (!url.substring(0,5).equalsIgnoreCase("https")) {
    try {
        URL testURL = new URL(url);
    }
    catch(Exception e) {
        BlurbCommon.handleException(e);
        return AdminMenuHTML.siteAddOrModify(this.serverRoot +
this.thisServlet,-1,blurb_site_name,description,url,"An error occurred. An invalid <B>Site
URL</B> was entered.");
    }
}

/*
Pick a random id for blurb_site and make sure that a record with
that id doesn't already exist.
*/
conn = dataConnMan.getConnection(server, dbname);
while (count > 0) {
    tempval = Math.floor(1000000 * Math.random());
    id = (int)tempval;
    try {
        rs = bs.getAll(conn, "blurb_site_id = " + id);
    } catch(Exception e) {
        BlurbCommon.handleException(e);
        break;
    }
    count = rs.size();
}
dataConnMan.freeConnection(server, dbname, conn);
if (count > 0)
    return AdminMenuHTML.getAdminMenu(getFirstSiteId(), this.serverRoot + this.thisServlet,
getSiteSelectList(getFirstSiteId(),true,syndicated),"An error occurred. The new site
configuration was not added.",syndicated);

/*
Make sure that blurb_site_name doesn't already exist in the database.
*/
conn = dataConnMan.getConnection(server, dbname);
try {
    //blurb_site_name = new RE("").substituteAll(blurb_site_name, "'");
    rs = bs.getAll(conn, "blurb_site_name = '" + new RE("").substituteAll(blurb_site_name,
"''") + "'");
} catch(Exception e) {
    BlurbCommon.handleException(e);
    errFlag = true;
}
dataConnMan.freeConnection(server, dbname, conn);

if(rs.size() > 0)
    return AdminMenuHTML.siteAddOrModify(this.serverRoot +
this.thisServlet,-1,blurb_site_name,description,url,"The site <B>" + blurb_site_name + "</B>
already exists. Please try again.");
if(errFlag)
    return AdminMenuHTML.getAdminMenu(getFirstSiteId(), this.serverRoot + this.thisServlet,
getSiteSelectList(getFirstSiteId(),true,syndicated),"An error occurred. The new site
configuration was not added.",syndicated);

```

```

//attempt to do the database insert of the new blurb_site record;
bs.blurb_site_id = id;
bs.blurb_site_name = blurb_site_name;
bs.description = description;
bs.url = url;
bs.status = "Y";
bs.created_on = new java.util.Date();
bs.last_modified_on = new java.util.Date();

conn = dataConnMan.getConnection(server, dbname);
try {
    bs.insert(conn);
} catch (Exception e) {
    BlurbCommon.handleException(e);
    errFlag = true;
}
dataConnMan.freeConnection(server, dbname, conn);

if(errFlag)
    return AdminMenuHTML.getAdminMenu(getFirstSiteId(), this.serverRoot + this.thisServlet,
getSiteSelectList(getFirstSiteId(),true,syndicated),"An error occurred. The new site
configuration was not added.",syndicated);
else
    return AdminMenuHTML.getAdminMenu(getFirstSiteId(), this.serverRoot + this.thisServlet,
getSiteSelectList(getFirstSiteId(),true,syndicated),"The new site configuration for <B>" +
blurb_site_name + "</B> was added successfully.",syndicated);
}

private synchronized String doGroupAdd(HttpServletRequest request, HttpServletResponse
response, String forwardLink, String forwardPost, boolean syndicated) {
    boolean errFlag = false;
    int id = 0;
    int count = 1;
    double tempval;
    Connection conn;
    Vector rs = null;
    BlurbGroupDB bg = new BlurbGroupDB();
    int jbsid = Integer.parseInt(request.getParameter("jbsid").trim());
    String blurb_group_name = request.getParameter("blurb_group_name").trim();
    //String description = request.getParameter("description").trim();

    if (blurb_group_name == null) {
        return
AdminMenuHTML.groupAdd(jbsid,-1,forwardLink,forwardPost,blurb_group_name,getSiteSelectList(jbsid,
false, syndicated),"An Invalid blurb group name. Please try again.");
    } else if (blurb_group_name.trim().equalsIgnoreCase("")) {
        return
AdminMenuHTML.groupAdd(jbsid,-1,forwardLink,forwardPost,blurb_group_name,getSiteSelectList(jbsid,
false, syndicated),"An Invalid blurb group name. Please try again.");
    }

/*
Pick a random id for blurb_group and make sure that a record with
that id doesn't already exist.
*/
    conn = dataConnMan.getConnection(server, dbname);
    while (count > 0) {
        tempval = Math.floor(1000000 * Math.random()) ;
        id = (int)tempval;
        try {
            rs = bg.getAll(conn, "blurb_group_id = " + id);
        } catch(Exception e) {
            BlurbCommon.handleException(e);
            break;
        }
        count = rs.size();
    }
    dataConnMan.freeConnection(server, dbname, conn);
    if (count > 0)

```

```

        return
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(getFirstSiteId(),true,syndicated),
"An error occurred. The new blurb group was not added.",syndicated);

/*
Make sure that blurb_group_name doesn't already exist in the database.
*/
conn = dataConnMan.getConnection(server, dbname);
try {
    rs = bg.getAll(conn, "blurb_group_name = '" + new RE("'").substituteAll(blurb_group_name,
"''") + "'");
} catch(Exception e) {
    BlurbCommon.handleException(e);
    errFlag = true;
}
dataConnMan.freeConnection(server, dbname, conn);

if(rs.size() > 0)
    return
AdminMenuHTML.groupAdd(jbsid,-1,forwardLink,forwardPost,blurb_group_name,getSiteSelectList(jbsid,
false, syndicated),"The group <B>" + blurb_group_name + "</B> already exists. Please try
again.");
if(errFlag)
    return
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated),"An error
occurred. The new blurb group was not added.",syndicated);

//attempt to do the database insert of the new blurb_group record;
bg.blurb_site_id = jbsid;
bg.blurb_group_id = id;
bg.blurb_group_name = blurb_group_name;
//bg.description = description;
bg.status = "Y";
bg.created_on = new java.util.Date();
bg.last_modified_on = new java.util.Date();

conn = dataConnMan.getConnection(server, dbname);
try {
    bg.insert(conn);
} catch (Exception e) {
    BlurbCommon.handleException(e);
    errFlag = true;
}
dataConnMan.freeConnection(server, dbname, conn);

if(errFlag)
    return
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "An error
occurred. The new blurb group was not added.",syndicated);
else
    return
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "The new
blurb group <B>" + blurb_group_name + "</B> was added to the <B>" + getSiteNameFromId(jbsid) +
"</B> site.",syndicated);
}

private synchronized String doBlurbAdd(HttpServletRequest request, HttpServletResponse
response, String forwardLink, String forwardPost, boolean syndicated) {
    boolean errFlag = false;
    int id = 0;
    int count = 1;
    double tempval;
    Connection conn;
    Vector rs = null;
    BlurbDB bb = new BlurbDB();
    GregorianCalendar gc = new GregorianCalendar();
    int jbsid = Integer.parseInt(request.getParameter("jbsid").trim());
    int jbgid = Integer.parseInt(request.getParameter("jbgid").trim());

    String html = request.getParameter("html").trim();
}

```

```

/*
Pick a random id for blurb and make sure that a record with
that id doesn't already exist.
*/
conn = dataConnMan.getConnection(server, dbname);
while (count > 0) {
    tempval = Math.floor(1000000 * Math.random()) ;
    id = (int)tempval;
    try {
        rs = bb.getAll(conn, "blurb_id = " + id);
    } catch(Exception e) {
        BlurbCommon.handleException(e);
        break;
    }
    count = rs.size();
}
dataConnMan.freeConnection(server, dbname, conn);
if (count > 0)
    return
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "An error
occurred. The new blurb was not added.",syndicated);

//attempt to do the database insert of the new blurb record;
bb.blurb_site_id = jbsid;
bb.blurb_group_id = jbgid;
bb.blurb_id = id;
bb.html = html;
bb.status = "Y";

//bb.created_on = new java.util.Date();
bb.created_on = new Timestamp(gc.getTime().getTime());
bb.last_modified_on = new Timestamp(gc.getTime().getTime());

conn = dataConnMan.getConnection(server, dbname);
try {
    bb.insert(conn);
} catch (Exception e) {
    BlurbCommon.handleException(e);
    errFlag = true;
}
dataConnMan.freeConnection(server, dbname, conn);

if(errFlag)
    return
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "An error
occurred. The new blurb was not added.",syndicated);
else
    return viewGroupBlurbs(jbsid,jbgid,-1,forwardLink,forwardPost,"The new blurb was added
successfully.",syndicated);
}

private String doSiteChange(HttpServletRequest request, HttpServletResponse response, boolean
syndicated) {
    boolean errFlag = false;
    Connection conn;
    Vector rs = null;
    BlurbSiteDB bs = new BlurbSiteDB();
    int id = Integer.parseInt(request.getParameter("jbsid").trim());
    int tempid = id;
    String blurb_site_name = request.getParameter("blurb_site_name").trim();
    String description = request.getParameter("description").trim();
    String url = request.getParameter("url").trim();

    /*
    Test if site, desc, and http:// are valid
    */
    try {
        if (blurb_site_name.trim().equalsIgnoreCase("")) {

```

```

        return AdminMenuHTML.siteAddOrModify(this.serverRoot + this.thisServlet, id,
blurb_site_name, description, url, "An error occurred. An invalid <B>Site Name or Site
Description</B> was entered.");
    }
    if (description.trim().equalsIgnoreCase("")) {
        return AdminMenuHTML.siteAddOrModify(this.serverRoot + this.thisServlet, id,
blurb_site_name, description, url, "An error occurred. An invalid <B>Site Name or Site
Description</B> was entered.");
    }
}
catch(Exception e) {
    BlurbCommon.handleException(e);
    return AdminMenuHTML.siteAddOrModify(this.serverRoot + this.thisServlet, id,
blurb_site_name, description, url, "An error occurred. An invalid <B>Site Name or Site
Description</B> was entered.");
}

/*
Test for a valid URL
*/
if (url.length() > 9) {
    if (url.substring(0,7).equalsIgnoreCase("http://")) {
    } else if(url.substring(0,8).equalsIgnoreCase("https://")) {
    } else {
        return AdminMenuHTML.siteAddOrModify(this.serverRoot +
this.thisServlet,id,blurb_site_name,description,url,"An error occurred. An invalid <B>Site
URL</B> was entered.");
    }
} else {
    return AdminMenuHTML.siteAddOrModify(this.serverRoot +
this.thisServlet,id,blurb_site_name,description,url,"An error occurred. An invalid <B>Site
URL</B> was entered.");
}
if (!url.substring(0,5).equalsIgnoreCase("https")) {
    try {
        URL testURL = new URL(url);
    }
    catch(Exception e) {
        BlurbCommon.handleException(e);
        return AdminMenuHTML.siteAddOrModify(this.serverRoot +
this.thisServlet,id,blurb_site_name,description,url,"An error occurred. An invalid <B>Site
URL</B> was entered.");
    }
}

/*
Make sure that the new blurb_site_name doesn't already exist in the database.
*/
conn = dataConnMan.getConnection(server, dbname);
try {
    //blurb_site_name = new RE("").substituteAll(blurb_site_name, "'");
    rs = bs.getAll(conn, "blurb_site_name = '" + new RE("").substituteAll(blurb_site_name,
"') + "'");
} catch(Exception e) {
    BlurbCommon.handleException(e);
    errFlag = true;
}
dataConnMan.freeConnection(server, dbname, conn);

if(errFlag)
    return AdminMenuHTML.getAdminMenu(getFirstSiteId(), this.serverRoot + this.thisServlet,
getSiteSelectList(getFirstSiteId(),true,syndicated), "An error occurred. The site configuration
change was not completed.",syndicated);

if(rs.size() > 0) {
    Enumeration e = rs.elements();
    while (e.hasMoreElements()) {
        BlurbSiteDB record = (BlurbSiteDB) e.nextElement();
        tempid = record.blurb_site_id;
    }
}

```

```

        if(tempid != id)
            return AdminMenuHTML.siteAddOrModify(this.serverRoot +
this.thisServlet,id,blurb_site_name,description,url,"The site <B>" + blurb_site_name + "</B>
already exists. Please try again.");
    }

    //attempt to do the database update of the modified blurb_site record;
    bs.blurb_site_id = id;
    bs.blurb_site_name = blurb_site_name;
    bs.description = description;
    bs.url = url;
    bs.status = "Y";
    bs.created_on = new java.util.Date();
    bs.last_modified_on = new java.util.Date();

    conn = dataConnMan.getConnection(server, dbname);
    try {
        bs.update(conn);
    } catch (Exception e) {
        BlurbCommon.handleException(e);
        errFlag = true;
    }
    dataConnMan.freeConnection(server, dbname, conn);

    if(errFlag)
        return AdminMenuHTML.getAdminMenu(getFirstSiteId(), this.serverRoot + this.thisServlet,
getSiteSelectList(getFirstSiteId(),true,syndicated), "An error occurred. The site configuration
was not changed.",syndicated);
    else
        return AdminMenuHTML.getAdminMenu(getFirstSiteId(), this.serverRoot + this.thisServlet,
getSiteSelectList(getFirstSiteId(),true,syndicated), "The site configuration for <B>" +
blurb_site_name + "</B> was changed successfully.",syndicated);
}

private String doBlurbChange(HttpServletRequest request, HttpServletResponse response, String
forwardLink, String forwardPost, boolean syndicated) {
    boolean errFlag = false;
    Connection conn;
    Vector rs = null;
    BlurbDB bb = new BlurbDB();
    int jbsid = Integer.parseInt(request.getParameter("jbsid").trim());
    int jbgid = Integer.parseInt(request.getParameter("jbgid").trim());
    int id = Integer.parseInt(request.getParameter("jbbid").trim());
    int tempid = id;
    String html = request.getParameter("html").trim();

    //attempt to do the database update of the modified blurb record;
    bb.blurb_id = id;
    bb.blurb_group_id = jbgid;
    bb.blurb_site_id = jbsid;
    bb.html = html;
    bb.status = "Y";
    bb.created_on = new java.util.Date();
    bb.last_modified_on = new java.util.Date();

    conn = dataConnMan.getConnection(server, dbname);
    try {
        bb.update(conn);
    } catch (Exception e) {
        BlurbCommon.handleException(e);
        errFlag = true;
    }
    dataConnMan.freeConnection(server, dbname, conn);

    if(errFlag)
        return
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "An error
occurred. The blurb was not changed.",syndicated);
    else

```

```

    //return AdminMenuHTML.getAdminMenu(jbsid, this.serverRoot + this.thisServlet,
    getSiteSelectList(jbsid,true), "The blurb was changed successfully.");
    return viewGroupBlurbs(jbsid,jbgid,-1,forwardLink,forwardPost,"The blurb was changed
successfully.",syndicated);
}

private String doSiteDelete(HttpServletRequest request, HttpServletResponse response, boolean
syndicated) {
    boolean errFlag = false;
    Connection conn;
    Vector rg = null, rb = null;
    BlurbSiteDB bs = new BlurbSiteDB();
    BlurbDB bb = new BlurbDB();
    BlurbGroupDB bg = new BlurbGroupDB();
    int id = Integer.parseInt(request.getParameter("jbsid").trim());

    //attempt to delete the blurb_site record and all associated blurb_group and blurb records;
    bs.blurb_site_id = id;
    conn = dataConnMan.getConnection(server, dbname);
    try {
        rb = bb.getAll(conn,"blurb_site_id=" + id);
        rg = bg.getAll(conn,"blurb_site_id=" + id);
        if(rb.size() > 0) {
            Enumeration q = rb.elements();
            while (q.hasMoreElements()) {
                BlurbDB b_record = (BlurbDB) q.nextElement();
                b_record.delete(conn);
            }
        }
        if(rg.size() > 0) {
            Enumeration x = rg.elements();
            while (x.hasMoreElements()) {
                BlurbGroupDB g_record = (BlurbGroupDB) x.nextElement();
                g_record.delete(conn);
            }
        }
        bs.delete(conn);
    } catch (Exception e) {
        BlurbCommon.handleException(e);
        errFlag = true;
    }
    dataConnMan.freeConnection(server, dbname, conn);

    if(errFlag)
        return AdminMenuHTML.getAdminMenu(getFirstSiteId(), this.serverRoot + this.thisServlet,
getSiteSelectList(getFirstSiteId(),true,syndicated), "An error occurred. The site configuration
was not deleted.",syndicated);
    else
        return AdminMenuHTML.getAdminMenu(getFirstSiteId(), this.serverRoot + this.thisServlet,
getSiteSelectList(getFirstSiteId(),true,syndicated), "The site configuration was deleted
successfully.",syndicated);
}

private String doGroupDelete(int jbsid, String forwardLink, String forwardPost,
HttpServletRequest request, HttpServletResponse response, boolean syndicated) {
    boolean errFlag = false;
    Connection conn;
    Vector rb = null;
    BlurbDB bb = new BlurbDB();
    BlurbGroupDB bg = new BlurbGroupDB();
    int id = Integer.parseInt(request.getParameter("jbgid").trim());

    //attempt to delete the blurb_group record and all associated blurb records;
    bg.blurb_group_id = id;
    conn = dataConnMan.getConnection(server, dbname);
    try {
        rb = bb.getAll(conn,"blurb_site_id=" + jbsid + " and blurb_group_id = " + id);
        if(rb.size() > 0) {

```

```

Enumeration q = rb.elements();
while (q.hasMoreElements()) {
    BlurbDB b_record = (BlurbDB) q.nextElement();
    b_record.delete(conn);
}
bg.delete(conn);

} catch (Exception e) {
    BlurbCommon.handleException(e);
    errFlag = true;
}
dataConnMan.freeConnection(server, dbname, conn);

if(errFlag)
    return
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "An error
occurred. The group was not deleted.",syndicated);
else
    return
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "The group
was deleted successfully.",syndicated);
}

private String doBlurbDelete(int jbsid, int jbgid, int jbbid, String forwardLink, String
forwardPost, boolean syndicated) {
    boolean errFlag = false;
    Connection conn;
    Vector v = null;
    BlurbDB bb = new BlurbDB();

    //attempt to delete the blurb record
    bb.blurb_id = jbbid;
    conn = dataConnMan.getConnection(server, dbname);
    try {
        bb.delete(conn);
    } catch (Exception e) {
        BlurbCommon.handleException(e);
        errFlag = true;
    }
    dataConnMan.freeConnection(server, dbname, conn);

    if(errFlag)
        return
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "An error
occurred. The blurb was not deleted.",syndicated);
else
    return viewGroupBlurbs(jbsid,jbgid,-1,forwardLink,forwardPost,"The blurb was deleted
successfully.",syndicated);
}

//private String getSiteSelectList(int jbsid, boolean syndicated) {
//    //return getSiteSelectList(jbsid,false,syndicated);
//}

private int getFirstSiteId() {
    boolean errFlag = false;
    boolean first = true;
    int id = -1;
    BlurbSiteDB bs = new BlurbSiteDB();
    Vector v = null;
    Connection conn = dataConnMan.getConnection(server, dbname);
    try {
        v = bs.getAll(conn, "1=1 order by blurb_site_name asc");
    } catch(Exception e) {
        BlurbCommon.handleException(e);
        errFlag = true;
    }
    dataConnMan.freeConnection(server, dbname, conn);
}

```

```

if(errFlag)
    return -1;

Enumeration e = v.elements();
if(v.size() > 0) {
    while (e.hasMoreElements()) {
        BlurbSiteDB record = (BlurbSiteDB) e.nextElement();
        if(first)
            id = record.blurb_site_id;
        first = false;
    }
} else {
    id = -1;
}
return id;
}

private String getSiteNameFromId(int jbsid) {
    boolean errFlag = false;
    String blurb_site_name = "";
    BlurbSiteDB bs = new BlurbSiteDB();
    Vector v = null;
    Connection conn = dataConnMan.getConnection(server, dbname);
    try {
        v = bs.getAll(conn, "blurb_site_id =" + jbsid);
    } catch(Exception e) {
        BlurbCommon.handleException(e);
        errFlag = true;
    }
    dataConnMan.freeConnection(server, dbname, conn);

    if(errFlag)
        return "error";

    Enumeration e = v.elements();
    if(v.size() > 0) {
        while (e.hasMoreElements()) {
            BlurbSiteDB record = (BlurbSiteDB) e.nextElement();
            blurb_site_name = record.blurb_site_name;
        }
    } else {
        blurb_site_name = "error";
    }
    return blurb_site_name;
}

private String getGroupNameFromId(int jbgid) {
    boolean errFlag = false;
    String blurb_group_name = "";
    BlurbGroupDB bg = new BlurbGroupDB();
    Vector v = null;
    Connection conn = dataConnMan.getConnection(server, dbname);
    try {
        v = bg.getAll(conn, "blurb_group_id =" + jbgid);
    } catch(Exception e) {
        BlurbCommon.handleException(e);
        errFlag = true;
    }
    dataConnMan.freeConnection(server, dbname, conn);

    if(errFlag)
        return "error";

    Enumeration e = v.elements();
    if(v.size() > 0) {
        while (e.hasMoreElements()) {
            BlurbGroupDB record = (BlurbGroupDB) e.nextElement();
            blurb_group_name = record.blurb_group_name;
        }
    }
}

```

```

    } else {
        blurb_group_name = "error";
    }
    return blurb_group_name;
}

private String getBlurbHTMLFromId(int jbbid) {
    boolean errFlag = false;
    String html = "";
    BlurbDB bb = new BlurbDB();
    Vector v = null;
    Connection conn = dataConnMan.getConnection(server, dbname);
    try {
        v = bb.getAll(conn, "blurb_id =" + jbbid);
    } catch(Exception e) {
        BlurbCommon.handleException(e);
        errFlag = true;
    }
    dataConnMan.freeConnection(server, dbname, conn);

    if(errFlag)
        return "error";

    Enumeration e = v.elements();
    if(v.size() > 0) {
        while (e.hasMoreElements()) {
            BlurbDB record = (BlurbDB) e.nextElement();
            html = record.html;
        }
    } else {
        html = "error";
    }
    return html;
}

private String getSiteSelectList(int jbsid, boolean includeJavascript, boolean syndicated) {
    boolean errFlag = false;
    boolean first = true;
    BlurbSiteDB bs = new BlurbSiteDB();
    Vector v = null;

    if(syndicated)
        return getSiteNameFromId(jbsid);
    else {
        Connection conn = dataConnMan.getConnection(server, dbname);
        try {
            v = bs.getAll(conn, "1=1 order by blurb_site_name asc");
        } catch(Exception e) {
            BlurbCommon.handleException(e);
            errFlag = true;
        }
        dataConnMan.freeConnection(server, dbname, conn);

        if(errFlag)
            return "error";

        Enumeration e = v.elements();
        StringBuffer sb = new StringBuffer();
        StringBuffer jParams = new StringBuffer();
        StringBuffer js = new StringBuffer();

        if(v.size() > 0) {
            if(includeJavascript)
                sb.append("<SELECT NAME=\"jbsid\" onChange=\"jgo(this, 1, false)\">" + '\r');
            else
                sb.append("<SELECT NAME=\"jbsid\">" + '\r');
            while (e.hasMoreElements()) {
                BlurbSiteDB record = (BlurbSiteDB) e.nextElement();
                if(first)
                    jParams.append("\",\"" + this.serverRoot + this.thisServlet + "?jbsid=" +

```

```

record.blurb_site_id + "\"");
    else
        jParams.append(",\"" + this.serverRoot + this.thisServlet + "?jbsid=" +
record.blurb_site_id + "\"");

    if((first) && (includeJavascript))
        sb.append("<option value=\"\r>Select a site ></option>" + '\r');
    if(jbsid == record.blurb_site_id)
        sb.append("<option value=\"" + record.blurb_site_id + "\" selected>" +
record.blurb_site_name + "</option>" + '\r');
    else
        sb.append("<option value=\"" + record.blurb_site_id + "\">" + record.blurb_site_name
+ "</option>" + '\r');
        first = false;
    }
    sb.append("</SELECT>" + '\r');

    if(includeJavascript) {
        String redirParams = jParams.toString();
        js.append("<SCRIPT language=\"JavaScript1.2\">" + '\r');
        js.append("function buildArray() {" + '\r');
        js.append("var a = buildArray.arguments;" + '\r');
        js.append("for (i=0; i<a.length; i++) {" + '\r');
        js.append("this[i] = a[i];" + '\r');
        js.append("}" + '\r');
        js.append("this.length = a.length;" + '\r');
        js.append("}" + '\r');
        js.append("var urls1 = new buildArray(" + redirParams + ");" + '\r');
//        System.out.println(redirParams);
        js.append("function jgo(which, num, win) {" + '\r');
        js.append("n = which.selectedIndex;" + '\r');
        js.append("if (n != 0) {" + '\r');
        js.append("var url = eval(\"urls\" + num + \"[n]\")" + '\r');
        js.append("if (win) {" + '\r');
        js.append("openWindow(url);" + '\r');
        js.append("} else {" + '\r');
        js.append("location.href = url;" + '\r');
        js.append("}" + '\r');
        js.append("}" + '\r');
        js.append("}" + '\r');
        js.append("</SCRIPT>" + '\r');
    }

} else {
    return "none";
}
return js.toString() + sb.toString();
}
}

private String getGroupSelectList(int jbsid, int jbgid) {
    boolean errFlag = false;
    boolean first = true;
    BlurbGroupDB bg = new BlurbGroupDB();
    Vector v = null;
    Connection conn = dataConnMan.getConnection(server, dbname);
    try {
        v = bg.getAll(conn, "blurb_site_id = " + jbsid + " order by blurb_group_name asc");
    } catch(Exception e) {
        BlurbCommon.handleException(e);
        errFlag = true;
    }
    dataConnMan.freeConnection(server, dbname, conn);

    if(errFlag)
        return "error";

    Enumeration e = v.elements();
    StringBuffer sb = new StringBuffer();

```

```

        if(v.size() > 0) {
            sb.append("<SELECT NAME=\"jbgid\">" + '\r');
            while (e.hasMoreElements()) {
                BlurbGroupDB record = (BlurbGroupDB) e.nextElement();
                //if(first)
                //    sb.append("<option value=\"\">Select a site ></option>" + '\r');
                if(jbgid == record.blurb_group_id)
                    sb.append("<option value=\"" + record.blurb_group_id + "\" selected>" +
record.blurb_group_name + "</option>" + '\r');
                else
                    sb.append("<option value=\"" + record.blurb_group_id + "\">" + record.blurb_group_name
+ "</option>" + '\r');
                first = false;
            }
            sb.append("</SELECT>" + '\r');
        } else {
            return "none";
        }
        return sb.toString();
    }

    private String viewGroups(int jbsid, String forwardLink, String forwardPost, boolean
syndicated) {
        boolean errFlag = false;
        BlurbGroupDB bg = new BlurbGroupDB();
        BlurbDB bb = new BlurbDB();
        Vector v = null, z = null;
        Connection conn = dataConnMan.getConnection(server, dbname);
        try {
            v = bg.getAll(conn, "blurb_site_id = " + jbsid + " order by blurb_group_name asc");
        } catch(Exception e) {
            BlurbCommon.handleException(e);
            errFlag = true;
        }
        dataConnMan.freeConnection(server, dbname, conn);

        Enumeration e = v.elements();
        StringBuffer sb = new StringBuffer();

        if(v.size() > 0) {
            conn = dataConnMan.getConnection(server, dbname);
            while (e.hasMoreElements()) {
                BlurbGroupDB record = (BlurbGroupDB) e.nextElement();
                try {
                    z = bb.getAll(conn,"blurb_site_id = " + jbsid + " and blurb_group_id = " +
record.blurb_group_id);
                    sb.append("<tr><td align=left>&nbsp;&nbsp;&nbsp;" + record.blurb_group_name +
"</td><td align=left>&nbsp;&nbsp;&nbsp;" + z.size() + "</td></tr>" + '\r');
                } catch(Exception ex) {
                    BlurbCommon.handleException(ex);
                    errFlag = true;
                }
            }
            dataConnMan.freeConnection(server, dbname, conn);
        } else {
            return AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated),"No groups
are currently configured. Please add a new group.",syndicated);
        }
        if(errFlag)
            return AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated),"An error
occurred in connecting to the database.",syndicated);
        else
            return AdminMenuHTML.viewGroups(jbsid,forwardLink,forwardPost,sb.toString(),getSiteNameFromId(jbsid));
    }

    private String changeSiteDetails(HttpServletRequest request, HttpServletResponse response,
boolean syndicated) {

```

```

boolean errFlag = false;
int id = Integer.parseInt(request.getParameter("jbsid").trim());
String blurb_site_name = "";
String description = "";
String url = "";
Vector rs = null;
BlurbSiteDB bs = new BlurbSiteDB();

Connection conn = dataConnMan.getConnection(server, dbname);
try {
    rs = bs.getAll(conn, "blurb_site_id = " + id);
} catch(Exception e) {
    BlurbCommon.handleException(e);
    errFlag = true;
}
dataConnMan.freeConnection(server, dbname, conn);
if(errFlag)
    return AdminMenuHTML.getAdminMenu(getFirstSiteId(), this.serverRoot + this.thisServlet,
getSiteSelectList(getFirstSiteId(),true,syndicated), "An error occurred in connecting to the
database.",syndicated);

if(rs.size() > 0) {
    Enumeration e = rs.elements();
    while (e.hasMoreElements()) {
        BlurbSiteDB record = (BlurbSiteDB) e.nextElement();
        blurb_site_name = record.blurb_site_name;
        description = record.description;
        url = record.url;
    }
} else {
    return AdminMenuHTML.getAdminMenu(getFirstSiteId(), this.serverRoot + this.thisServlet,
getSiteSelectList(getFirstSiteId(),true,syndicated), "The site record selected no longer exists
in the database.",syndicated);
}
return AdminMenuHTML.siteAddOrModify(this.serverRoot +
this.thisServlet,id,blurb_site_name,description,url,"");
}

private String changeSiteSelect(boolean syndicated) {
    String param = getSiteSelectList(-1, false, syndicated);
    if(param.equals("error")) {
        return AdminMenuHTML.getAdminMenu(getFirstSiteId(), this.serverRoot + this.thisServlet,
getSiteSelectList(getFirstSiteId(),true,syndicated), "An error occurred in connecting to the
database.",syndicated);
    } else if(param.equals("none")) {
        return AdminMenuHTML.getAdminMenu(getFirstSiteId(), this.serverRoot + this.thisServlet,
getSiteSelectList(getFirstSiteId(),true,syndicated), "No sites are currently configured. Please
configure a new site.",syndicated);
    } else {
        return AdminMenuHTML.selectSite(this.serverRoot + this.thisServlet,"chg",param);
    }
}

private String genSiteCodeSelect(boolean syndicated) {
    String param = getSiteSelectList(-1, false, syndicated);
    if(param.equals("error")) {
        return AdminMenuHTML.getAdminMenu(getFirstSiteId(), this.serverRoot + this.thisServlet,
getSiteSelectList(getFirstSiteId(),true,syndicated), "An error occurred in connecting to the
database.",syndicated);
    } else if(param.equals("none")) {
        return AdminMenuHTML.getAdminMenu(getFirstSiteId(), this.serverRoot + this.thisServlet,
getSiteSelectList(getFirstSiteId(),true,syndicated), "No sites are currently configured. Please
configure a new site.",syndicated);
    } else {
        return AdminMenuHTML.selectSite(this.serverRoot + this.thisServlet,"gensitecode",param);
    }
}
/*
private String changeGroupSelect(boolean syndicated) {
    String param = getSiteSelectList(-1, false, syndicated);

```

```

        if(param.equals("error")) {
            return AdminMenuHTML.getAdminMenu(getFirstSiteId(), this.serverRoot + this.thisServlet,
getSiteSelectList(getFirstSiteId(),true,syndicated), "An error occurred in connecting to the
database.",syndicated);
        } else if(param.equals("none")) {
            return AdminMenuHTML.getAdminMenu(getFirstSiteId(), this.serverRoot + this.thisServlet,
getSiteSelectList(getFirstSiteId(),true,syndicated), "No sites are currently configured. Please
configure a new site.",syndicated);
        } else {
            return AdminMenuHTML.selectSite(this.serverRoot + this.thisServlet,"chg",param);
        }
    }
*/
private String deleteSiteSelect(boolean syndicated) {
    String param = getSiteSelectList(-1, false, syndicated);
    if(param.equals("error")) {
        return AdminMenuHTML.getAdminMenu(getFirstSiteId(), this.serverRoot + this.thisServlet,
getSiteSelectList(getFirstSiteId(),true,syndicated), "An error occurred in connecting to the
database.",syndicated);
    } else if(param.equals("none")) {
        return AdminMenuHTML.getAdminMenu(getFirstSiteId(), this.serverRoot + this.thisServlet,
getSiteSelectList(getFirstSiteId(),true,syndicated), "No sites are currently configured. Please
configure a new site.",syndicated);
    } else {
        return AdminMenuHTML.selectSite(this.serverRoot + this.thisServlet,"del",param);
    }
}

private String addBlurb(int jbsid, String forwardLink, String forwardPost, boolean syndicated)
{
    String param = getGroupSelectList(jbsid, -1);
    if(param.equals("error")) {
        return
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "An error
occurred in connecting to the database.",syndicated);
    } else if(param.equals("none")) {
        return
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "No blurb
groups exist for the current site. <BR>Please add a new blurb group before adding a
blurb.",syndicated);
    } else {
        return
AdminMenuHTML.blurbAddOrModify(jbsid,-1,-1,forwardLink,forwardPost,getSiteNameFromId(jbsid), "",pa
ram, "");
    }
}

private String copyBlurb(int jbsid, int jbgid, int jbbid, String forwardLink, String
forwardPost, boolean syndicated) {
    String param = getGroupSelectList(jbsid, jbgid);
    String html = getBlurbHTMLFromId(jbbid);
    if(param.equals("error")) {
        return
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "An error
occurred in connecting to the database.",syndicated);
    } else if(param.equals("none")) {
        return
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "No blurb
groups exist for the current site. <BR>Please add a new blurb group before adding a
blurb.",syndicated);
    } else {
        return
AdminMenuHTML.blurbAddOrModify(jbsid,jbgid,-1,forwardLink,forwardPost,getSiteNameFromId(jbsid),ht
ml,param, "");
    }
}

private String changeBlurb(int jbsid, int jbgid, int jbbid, String forwardLink, String
forwardPost, boolean syndicated) {
    String param = getGroupSelectList(jbsid, jbgid);
}

```

```

String html = getBlurbHTMLFromId(jbbid);
if(param.equals("error")) {
    return
}
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "An error
occurred in connecting to the database.",syndicated);
} else if(param.equals("none")) {
    return
}
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "No blurb
groups exist for the current site. <BR>Please add a new blurb group before adding a
blurb.",syndicated);
} else {
    return
}
AdminMenuHTML.blurbAddOrModify(jbsid,jbgid,jbbid,forwardLink,forwardPost,getSiteNameFromId(jbsid)
,html,param,"");
}

private String deleteGroupSelect(int jbsid, String forwardLink, String forwardPost, boolean
syndicated) {
    String param = getGroupSelectList(jbsid, -1);
    if(param.equals("error")) {
        return
}
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "An error
occurred in connecting to the database.",syndicated);
} else if(param.equals("none")) {
    return
}
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "No groups
are currently configured. Please add a new group.",syndicated);
} else {
    return
}
AdminMenuHTML.selectGroup(jbsid,forwardLink,forwardPost,"delbg",param,getSiteNameFromId(jbsid));
}

private String viewBlurbSelect(int jbsid, String forwardLink, String forwardPost, boolean
syndicated) {
    String param = getGroupSelectList(jbsid, -1);
    if(param.equals("error")) {
        return
}
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "An error
occurred in connecting to the database.",syndicated);
} else if(param.equals("none")) {
    return
}
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "No
groups are currently configured. Please add a new group.",syndicated);
} else {
    return
}
AdminMenuHTML.selectGroup(jbsid,forwardLink,forwardPost,"viewbb",param,getSiteNameFromId(jbsid));
}

private String viewGroupBlurbs(int jbsid, int jbgid, int jbbid, String forwardLink, String
forwardPost, String message, boolean syndicated) {
    boolean errFlag = false;
    BlurbDB bb = new BlurbDB();
    Vector v = null;
    Connection conn = dataConnMan.getConnection(server, dbname);
    String q = "?";
    if(forwardLink.indexOf("?") > -1)
        q = "&";
    try {
        if(jbbid > 0)
            v = bb.getAll(conn, "blurb_site_id = " + jbsid + " and blurb_group_id = " + jbgid + "
and blurb_id = " + jbbid + " order by created_on desc");
        else
            v = bb.getAll(conn, "blurb_site_id = " + jbsid + " and blurb_group_id = " + jbgid + "
order by created_on desc");
    } catch(Exception ex) {
        BlurbCommon.handleException(ex);
    }
}

```

```

        errFlag = true;
    }
    dataConnMan.freeConnection(server, dbname, conn);

    Enumeration e = v.elements();
    StringBuffer sb = new StringBuffer();

    if(v.size() > 0) {
        conn = dataConnMan.getConnection(server, dbname);
        while (e.hasMoreElements()) {
            BlurbDB record = (BlurbDB) e.nextElement();
            if(jbbid > 0) {
                sb.append("<tr><td valign=top align=center>N/A</td>");
            } else {
                sb.append("<tr><td valign=top align=center>[<A HREF=\"" + forwardLink + q +
"jbact=copybb&jbsid=" + jbsid + "&jbgid=" + jbgid + "&jbbid=" + record.blurb_id +
"\"><B>Copy</B></A>]<BR>");
                sb.append("[<A HREF=\"" + forwardLink + q + "jbact=chgbb&jbsid=" + jbsid + "&jbgid=" +
jbgid + "&jbbid=" + record.blurb_id + "\"><B>Modify</B></A>]<BR>");
                sb.append("[<A HREF=\"" + forwardLink + q + "jbact=delbb&jbsid=" + jbsid + "&jbgid=" +
jbgid + "&jbbid=" + record.blurb_id + "\"><B>Delete</B></A>]</td>");
            }
            sb.append("<td valign=top align=left>" + HTMLQuoter.toQuoted(record.html) + "</td><td
valign=top align=left>" + record.html + "</td></tr>" + '\r');
        }
        dataConnMan.freeConnection(server, dbname, conn);
    } else {
        return
    }
    AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "The
group <B>" + getGroupNameFromId(jbgid) + "</B> doesn't contain any HTML blurbs.",syndicated);
}
if(errFlag)
    return
AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated), "An error
occurred in connecting to the database.",syndicated);
else
    if(jbbid > 0)
        return
AdminMenuHTML.deleteBlurb(jbsid,jbgid,jbbid,forwardLink,forwardPost,sb.toString(),getSiteNameFrom
Id(jbsid),getGroupNameFromId(jbgid), message);
else
    return
AdminMenuHTML.viewGroupBlurbs(jbsid,jbgid,forwardLink,forwardPost,sb.toString(),getSiteNameFromId
(jbsid),getGroupNameFromId(jbgid), message);
}

private String generateBlurbCode(int jbsid, String forwardLink, String forwardPost, boolean
syndicated) {
    String param = getGroupSelectList(jbsid, -1);
    if(param.equals("error")) {
        return
    }
    AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated),"An error
occurred in connecting to the database.",syndicated);
    } else if(param.equals("none")) {
        return
    }
    AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated),"No groups
are currently configured. Please add a new group.",syndicated);
    } else {
        return AdminMenuHTML.selectGroup(jbsid,forwardLink,forwardPost,"genbcde",param,
getSiteNameFromId(jbsid));
    }
}

private String deleteSiteDetails(HttpServletRequest request, HttpServletResponse response,
boolean syndicated) {
    boolean errFlag = false;
    int id = Integer.parseInt(request.getParameter("jbsid").trim());
    String blurb_site_name = "";
    String description = "";
    String url = "";

```

```

Vector rs = null;
BlurbSiteDB bs = new BlurbSiteDB();

Connection conn = dataConnMan.getConnection(server, dbname);
try {
    rs = bs.getAll(conn,"blurb_site_id = " + id);
} catch(Exception e) {
    BlurbCommon.handleException(e);
    errFlag = true;
}
dataConnMan.freeConnection(server, dbname, conn);
if(errFlag)
    return AdminMenuHTML.getAdminMenu(getFirstSiteId(), this.serverRoot + this.thisServlet,
getSiteSelectList(getFirstSiteId(),true,syndicated), "An error occurred in connecting to the
database.",syndicated);

if(rs.size() > 0) {
    Enumeration e = rs.elements();
    while (e.hasMoreElements()) {
        BlurbSiteDB record = (BlurbSiteDB) e.nextElement();
        blurb_site_name = record.blurb_site_name;
        description = record.description;
        url = record.url;
    }
} else {
    return AdminMenuHTML.getAdminMenu(getFirstSiteId(), this.serverRoot + this.thisServlet,
getSiteSelectList(getFirstSiteId(),true,syndicated), "The site record selected no longer exists
in the database.",syndicated);
}
return AdminMenuHTML.siteDelete(this.serverRoot +
this.thisServlet,id,blurb_site_name,description,url,"");
}

private String deleteGroupDetails(int jbsid, String forwardLink, String forwardPost,
HttpServletRequest request, HttpServletResponse response, boolean syndicated) {
    boolean errFlag = false;
    int id = Integer.parseInt(request.getParameter("jbgid").trim());
    String blurb_group_name = "";
    Vector rs = null;
    BlurbGroupDB bg = new BlurbGroupDB();

    Connection conn = dataConnMan.getConnection(server, dbname);
    try {
        rs = bg.getAll(conn,"blurb_group_id = " + id);
    } catch(Exception e) {
        BlurbCommon.handleException(e);
        errFlag = true;
    }
    dataConnMan.freeConnection(server, dbname, conn);
    if(errFlag)
        return
    AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated),"An error
occurred in connecting to the database.",syndicated);

    if(rs.size() > 0) {
        Enumeration e = rs.elements();
        while (e.hasMoreElements()) {
            BlurbGroupDB record = (BlurbGroupDB) e.nextElement();
            blurb_group_name = record.blurb_group_name;
        }
    } else {
        return
    AdminMenuHTML.getAdminMenu(jbsid,forwardLink,getSiteSelectList(jbsid,true,syndicated),"The group
record selected no longer exists in the database.",syndicated);
    }
    return
    AdminMenuHTML.groupDelete(jbsid,id,forwardLink,forwardPost,blurb_group_name,"",getSiteNameFromId(
jbsid));
}

```

```
private void validateSite() {  
}  
}  
}
```

```

//a test comment

package com.activedatax.products.Blurb;

import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.util.*;
import java.sql.*;
import java.net.*;
import gnu-regexp.*;
import com.activedatax.sql.HttpServletDb;
import com.activedatax.products.Blurb.dbmodules.BlurbDB;

/**
 This is a rewrite of the PublishNow 2.x jSyndicate servlet for use with Active Data Publisher
 3.0 and Community Exchange (AffiliateNow)
 @author John E. Wetzel
 @version 1.0
 @date 15-MAY-2000
 */

public class JBlurb extends HttpServlet implements SingleThreadModel {
    private BlurbCommon mycommon;
    private String server = "";
    private String dbname = "";
    private String thisServlet = "";
    private String serverRoot = "";

    public void init(ServletConfig config) throws ServletException {
        super.init(config);

        //USE BlurbCommon.class TO GET PROPERTIES -- SERVER & DATABASE NAME
        mycommon = new BlurbCommon();
        Properties p = mycommon.getProps();
        serverRoot = p.getProperty("server.root");
        server = p.getProperty ("default.server");
        dbname = p.getProperty ("default.dbname");

        //store the alias in
        p = mycommon.getAliases();
        thisServlet = p.getProperty("JBlurb");
    }

    //Process the HTTP Get request
    public void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        String outputHTML = "";
        String param = "";
        String action = request.getParameter("jbact");

        outputHTML =
convertTojs(getRandomBlurb(Integer.parseInt(request.getParameter("jbsid").trim()),
Integer.parseInt(request.getParameter("jbgid").trim())));

        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println(outputHTML);
        return;
    }

    public String getRandomBlurb(int blurb_site_id, int blurb_group_id) {
        boolean errFlag = false;
        int count = 0;
        int tempid, blurb_id;
        double tempval;
        String blurbHTML = "", tempStr = "";

```

```

BlurbDB bb = new BlurbDB();
Vector rs = null;
Hashtable idHash = new Hashtable();

Connection conn = dataConnMan.getConnection(server, dbname);
String sql = "select blurb.blurb_id id from blurb_site, blurb_group, blurb "
+ " where upper(blurb_site.status) = 'Y' and upper(blurb_group.status) = 'Y' and
upper(blurb.status) = 'Y' "
+ " and blurb.blurb_site_id = blurb_group.blurb_site_id and blurb.blurb_site_id =
blurb_site.blurb_site_id"
+ " and blurb.blurb_group_id = blurb_group.blurb_group_id"
+ " and blurb.blurb_site_id = ? and blurb.blurb_group_id = ?";

try {
    PreparedStatement stmt = conn.prepareStatement(sql);
    stmt.clearParameters();
    stmt.setInt(1, blurb_site_id);
    stmt.setInt(2, blurb_group_id);
    ResultSet rr = stmt.executeQuery() ;

    while (rr.next()) {
        count++;
        idHash.put(String.valueOf(count), rr.getString("id"));
    }
    rr.close();
    stmt.close();
} catch (Exception e) {
    BlurbCommon.handleException(e);
    errFlag = true;
}
dataConnMan.freeConnection(server, dbname, conn);

if(count > 0) {
    tempval = Math.floor(count * Math.random());
    tempid = (int)tempval + 1;
//    System.out.println(tempid);
    tempStr = idHash.get(String.valueOf(tempid)).toString();
    blurb_id = Integer.parseInt(tempStr);
//    System.out.println(blurb_id);

    conn = dataConnMan.getConnection(server, dbname);
    try {
        rs = bb.getAll(conn, "blurb_id = " + blurb_id);
    } catch(Exception e) {
        BlurbCommon.handleException(e);
        errFlag = true;
    }
    dataConnMan.freeConnection(server, dbname, conn);

    Enumeration e = rs.elements();
    while (e.hasMoreElements()) {
        BlurbDB record = (BlurbDB) e.nextElement();
        blurbHTML = record.getHtml();
    }
}

if(errFlag)
    return "<!-- database error --->";
else
    return blurbHTML;
}

public String convertTojs(String HTMLtext) {

try {
    HTMLtext = new

```

```

RE("\\"+com.activedatax.products.Affiliate.Datamodules.RetrieveImage").substituteAll(HTMLtext,
"\\" + serverRoot + "com.activedatax.products.Affiliate.Datamodules.RetrieveImage") ;
    HTMLtext = new RE("[Ss][Rr][Cc] *"
*\"+com.activedatax.products.Affiliate.Datamodules.RetrieveImage").substituteAll(HTMLtext, "src=\""
+ serverRoot + "com.activedatax.products.Affiliate.Datamodules.RetrieveImage");
    HTMLtext = new RE("[Ss][Rr][Cc] *"
*\"+com.rnci.products.DataModules.RetrieveImage").substituteAll(HTMLtext, "src=\""+ serverRoot +
"com.rnci.products.DataModules.RetrieveImage");
    HTMLtext = new RE("\\"+com.rnci.products.DataModules.RetrieveAttachment").substituteAll(HTMLtext, "\\" +
serverRoot + "com.rnci.products.DataModules.RetrieveAttachment");
}
catch(Exception e) {
    BlurbCommon.handleException(e);
}

String scriptHTML = "";
int startSearchAt = 0;
int scriptLoc = indexOfIgnoreCase(HTMLtext, "<SCRIPT", startSearchAt);
if(scriptLoc > -1) { //Check for client scripts and separate them from regular HTML
    try {
        int endScriptLoc;
        int articleLength = HTMLtext.length();
        StringBuffer sbArticle = new StringBuffer(articleLength + 1);
        StringBuffer sbScripts = new StringBuffer();
        while(scriptLoc > -1) {
            endScriptLoc = indexOfIgnoreCase(HTMLtext, "</SCRIPT>", scriptLoc);
            sbArticle.append(HTMLtext.substring(startSearchAt, scriptLoc));

            sbScripts.append(HTMLtext.substring((indexOfIgnoreCase(HTMLtext, ">", scriptLoc)+1), endScriptLoc));
            sbScripts.append('\n');
            sbScripts.append('\r');
            //System.out.println("The HTML: " + HTMLtext.substring(startSearchAt, scriptLoc));
            //System.out.println("The Script: " +
HTMLtext.substring((indexOfIgnoreCase(HTMLtext, ">", scriptLoc)+1), endScriptLoc));

            startSearchAt = endScriptLoc + 9;
            scriptLoc = indexOfIgnoreCase(HTMLtext, "<SCRIPT", startSearchAt);
        }
        sbArticle.append(HTMLtext.substring(startSearchAt, articleLength - 1));
        //System.out.println(sbArticle.toString());
        //System.out.println(sbScripts.toString());
        scriptHTML = sbScripts.toString(); //contains just the scripts' code
        HTMLtext = sbArticle.toString(); //contains just the article HTML
    }
    catch(Exception e) {
        BlurbCommon.handleException(e);
    }
}

HTMLtext = HTMLtext.replace('\r', ' ');
HTMLtext = HTMLtext.replace('\n', ' ');
try {
    HTMLtext = swapStrings(HTMLtext, "\\", "\\\"");
    HTMLtext = swapStrings(HTMLtext, "\\", "\\\'");
    HTMLtext = swapStrings(HTMLtext, "\\", "\\\\\"");
}
catch(Exception e) {
    BlurbCommon.handleException(e);
}

return scriptHTML + " document.write ('" + HTMLtext + "'); "
}

public static String swapStrings(String TextIn, String OldString, String NewString) {
    int OldStringStartAt;
    int OldStringLength = OldString.length();
    int NewStringLength = NewString.length();

```

```
String TempText = TextIn;
OldStringStartAt = indexOfIgnoreCase(TempText, OldString, 0);
while (OldStringStartAt != -1) {
    TempText = TempText.substring(0, OldStringStartAt) + NewString +
TempText.substring(OldStringStartAt + OldStringLength);
    OldStringStartAt = indexOfIgnoreCase(TempText, OldString, OldStringStartAt +
NewStringLength);
}
return TempText;
}

public static int indexOfIgnoreCase(String TempText, String subString, int StartAt) {
    return TempText.toLowerCase().indexOf( subString.toLowerCase(), StartAt);
}
}
```

```

//A simple comment

package com.activedatax.products.Blurb;

import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.util.*;
import java.sql.*;
import com.activedatax.products.Blurb.*;
import com.activedatax.products.Blurb.dbmodules.*;
import gnu-regexp.*;
import com.activedatax.net.url.Util;
import java.net.MalformedURLException;
import com.activedatax.utils.HTMLQuoter;

public class JBlurbAdmin extends AdminMenu {
    private BlurbCommon mycommon;
    private String server = "";
    private String dbname = "";
    private String thisServlet = "";
    private String serverRoot = "";
    private String fPost = "";

    private static int itemsToKeep = 400, minutesToLive = 10; //will be set by init() later
    private static long lastModified;
    private static Vector contents = new Vector();
    private Sentinel sentinel = null;

    public void init(ServletConfig config) throws ServletException {
        super.init(config);

        System.out.println("Init Method executed");
        sentinel = new Sentinel(itemsToKeep, minutesToLive);
        //USE BlurbCommon.class TO GET PROPERTIES
        mycommon = new BlurbCommon();
        Properties p = mycommon.getProps();
        serverRoot = p.getProperty("server.root");
        server = p.getProperty ("default.server");
        dbname = p.getProperty ("default.dbname");

        try {
            this.itemsToKeep = Integer.parseInt(p.getProperty("cache.items"));
        } catch (NumberFormatException e) {
            System.err.println(e.getMessage());
            e.printStackTrace();
            this.itemsToKeep = 400;
        }
        try {
            this.minutesToLive = Integer.parseInt(p.getProperty("cache.minutes"));
        } catch (NumberFormatException e) {
            System.err.println(e.getMessage());
            e.printStackTrace();
            this.minutesToLive = 10;
        }

        //store the alias in
        p = mycommon.getAliases();
        thisServlet = p.getProperty("JBlurbAdmin");

        fPost = serverRoot + thisServlet;
    }

    /**Process the HTTP Get request*/
    public void doGet(HttpServletRequest request, HttpServletResponse response) throws
    ServletException, IOException {
        String outputHTML = "", documentURL = "", documentHost = "", jbact = "", jpostbact = "",
        jbts = "";
}

```

```

String contId;
boolean errFlag = false;
int jbsid = -1;
Content cont;

response.setContentType("text/html");
PrintWriter out = response.getWriter();

HttpSession session = request.getSession (false);
if (session == null) session = request.getSession(true);

documentURL = removeAllParams(request.getParameter("jbdurl").trim());
//get the blurb site id from the request parameters
try {
    jbsid = Integer.parseInt(request.getParameter("jbsid").trim());
} catch(Exception e) { errFlag = true; }
if(!errFlag) errFlag = validateHost(jbsid,parseHostFromURL(documentURL));
if (errFlag)
    outputHTML = "document.write('<B>This web site is not authorized to view the content
requested.<\\>'); ";
else {
    jbact = request.getParameter("jbact");
    if(jbact != null) {
        if(jbact.equalsIgnoreCase("getpostdata")) {
            jbts = request.getParameter("jbts").trim();
            jpostbact = request.getParameter("jpostbact").trim();
            contId = session.getId() + "," + jbsid + "," + documentURL + "," + jpostbact + "," +
            jbts;
            //System.out.println(contId);
            cont = lookup(contId);
            if (cont != null)
                outputHTML = cont.detail;
            else
                outputHTML = convertTojs(doBlurb(request, response, documentURL, this.fPost, true));
            out.println(outputHTML);
            return;
        }
    }
    outputHTML = convertTojs(doBlurb(request, response, documentURL, this.fPost, true));
}
out.println(outputHTML);
return;

}

/**Process the HTTP Post request*/
public void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
    String outputHTML = "", referringURL = "", redirectURL = "", jpostbact = "", q = "?";
    String contId;
    boolean errFlag = false;
    int jbsid = -1;
    Content cont;
    HttpSession session = request.getSession (false);
    if (session == null) session = request.getSession(true);

    //get the referring URL from the request header and use as the forward link
    referringURL = removeAllParams(request.getHeader("REFERER"));
    if(referringURL.equals("error")) errFlag = true;

    //get the blurb site id and the form post action from the request parameters
    try {
        jbsid = Integer.parseInt(request.getParameter("jbsid").trim());
        jpostbact = request.getParameter("jbact").trim();
    } catch(Exception e) { errFlag = true; }

    if(errFlag == false) errFlag = validateHost(jbsid,parseHostFromURL(referringURL));
    if(!errFlag) {

        //process the posted form and convert to JS for output

```

```

        outputHTML = convertTojs(doBlurb(request, response, referringURL, this.fPost, true));

        //store the resulting JS in a Vector for later retrieval via browser JavaScript HTTP GET
        if(referringURL.indexOf("?) > -1)
            q = "&";
        lastModified = System.currentTimeMillis();
        redirectURL = referringURL + q + "jbsid=" + jbsid + "&jbact=getpostData&jpostbact=" +
        jpostbact + "&jbts=" + lastModified;

        contId = session.getId() + "," + jbsid + "," + referringURL + "," + jpostbact + "," +
        lastModified;
        //System.out.println(contId);
        cont = new Content(contId, outputHTML, lastModified);
        contents.addElement(cont);

    } else { //there has been an error or the referring URL is not allowed
        redirectURL = referringURL;
    }

    //redirect the browser to the referring page with the appropriate parameters needed to
    retrieve the JS from the Vector
    response.sendRedirect(redirectURL);
    return;
}

/**Clean up resources*/
public void destroy() {
    System.out.println("Destroy method running");
    sentinel.quit = true;
    contents = null;
}

public String convertTojs(String HTMLtext) {
    try {
        HTMLtext = new
RE("\"com.activedatax.products.Affiliate.Datamodules.RetrieveImage").substituteAll(HTMLtext,
"\\" + serverRoot + "com.activedatax.products.Affiliate.Datamodules.RetrieveImage") ;
        HTMLtext = new RE("[Ss][Rr][Cc] *=
*\\"com.activedatax.products.Affiliate.Datamodules.RetrieveImage").substituteAll(HTMLtext,"src=\""
+ serverRoot + "com.activedatax.products.Affiliate.Datamodules.RetrieveImage");
        HTMLtext = new RE("[Ss][Rr][Cc] *=
*\\"com.rnci.products.DataModules.RetrieveImage").substituteAll(HTMLtext,"src=\""
+ serverRoot + "com.rnci.products.DataModules.RetrieveImage");
        HTMLtext = new
RE("\\"com.rnci.products.DataModules.RetrieveAttachment").substituteAll(HTMLtext,"\""
+ serverRoot + "com.rnci.products.DataModules.RetrieveAttachment");
    }
    catch(Exception e) {
        System.err.println(e.getMessage());
        e.printStackTrace();
    }

    String scriptHTML = "";
    int startSearchAt = 0;
    int scriptLoc = indexOfIgnoreCase(HTMLtext,<SCRIPT",startSearchAt);
    if(scriptLoc > -1) { //Check for client scripts and separate them from regular HTML
        try {
            int endScriptLoc;
            int articleLength = HTMLtext.length();
            StringBuffer sbArticle = new StringBuffer(articleLength + 1);
            StringBuffer sbScripts = new StringBuffer();
            while(scriptLoc > -1) {
                endScriptLoc = indexOfIgnoreCase(HTMLtext,"</SCRIPT>",scriptLoc);
                sbArticle.append(HTMLtext.substring(startSearchAt,scriptLoc));

                sbScripts.append(HTMLtext.substring((indexOfIgnoreCase(HTMLtext,>",
scriptLoc)+1),endScriptLoc));
                sbScripts.append('\n');
                sbScripts.append('\r');
                //System.out.println("The HTML: " + HTMLtext.substring(startSearchAt,scriptLoc));
                //System.out.println("The Script: " +
HTMLtext.substring((indexOfIgnoreCase(HTMLtext,>",
scriptLoc)+1),endScriptLoc));

```

```

        startSearchAt = endScriptLoc + 9;
        scriptLoc = indexOfIgnoreCase(HTMLtext, "<SCRIPT", startSearchAt);
    }
    sbArticle.append(HTMLtext.substring(startSearchAt, articleLength - 1));
    //System.out.println(sbArticle.toString());
    //System.out.println(sbScripts.toString());
    scriptHTML = sbScripts.toString(); //contains just the scripts' code
    HTMLtext = sbArticle.toString(); //contains just the article HTML
}
catch(Exception e) {
    System.err.println(e.getMessage());
    e.printStackTrace();
}
}

HTMLtext = HTMLtext.replace('\r', ' ');
HTMLtext = HTMLtext.replace('\n', ' ');
try {
    HTMLtext = swapStrings(HTMLtext, "\\", "\\\"\\\"");
    HTMLtext = swapStrings(HTMLtext, "\'", "\\\'\\\"");
    HTMLtext = swapStrings(HTMLtext, "\"", "\\\"\\\"");
}
catch(Exception e) {
    System.err.println(e.getMessage());
    e.printStackTrace();
}

return scriptHTML + " document.write ('" + HTMLtext + "'); ";
}

public static String swapStrings(String TextIn, String OldString, String NewString) {
    int OldStringStartAt;
    int OldStringLength = OldString.length();
    int NewStringLength = NewString.length();
    String TempText = TextIn;
    OldStringStartAt = indexOfIgnoreCase(TempText, OldString, 0);
    while (OldStringStartAt != -1) {
        TempText = TempText.substring(0, OldStringStartAt) + NewString +
        TempText.substring(OldStringStartAt + OldStringLength);
        OldStringStartAt = indexOfIgnoreCase(TempText, OldString, OldStringStartAt +
        NewStringLength);
    }
    return TempText;
}

public static int indexOfIgnoreCase(String TempText, String subString, int StartAt) {
    return TempText.toLowerCase().indexOf( subString.toLowerCase(), StartAt);
}

private String removeAllParams(String URL) {
    String tempURL = URL;
    Util myUtil = new Util();
    try {
        myUtil.setOriginalURL(URL);
        myUtil.setParamToRemove("jbdurl");
        tempURL = myUtil.getRemovedParamURL();
        myUtil.setOriginalURL(tempURL);
        myUtil.setParamToRemove("jbbrand");
        tempURL = myUtil.getRemovedParamURL();
        myUtil.setOriginalURL(tempURL);
        myUtil.setParamToRemove("jbact");
        tempURL = myUtil.getRemovedParamURL();
        myUtil.setOriginalURL(tempURL);
        myUtil.setParamToRemove("jbsid");
        tempURL = myUtil.getRemovedParamURL();
        myUtil.setOriginalURL(tempURL);
        myUtil.setParamToRemove("jbgid");
        tempURL = myUtil.getRemovedParamURL();
    }
}

```

```

myUtil.setOriginalURL(tempURL);
myUtil.setParamToRemove("jbbid");
tempURL = myUtil.getRemovedParamURL();
myUtil.setOriginalURL(tempURL);
myUtil.setParamToRemove("jpostbact");
tempURL = myUtil.getRemovedParamURL();
myUtil.setOriginalURL(tempURL);
myUtil.setParamToRemove("jbts");
tempURL = myUtil.getRemovedParamURL();

} catch (MalformedURLException mue) {
    mue.printStackTrace();
    //put in garbage URL that will never match JW 12/4/00
    tempURL = "error";
}

//System.out.println(tempURL);
return tempURL;
}

private String parseHostFromURL(String targetURL) {
    StringTokenizer tokenizer = new StringTokenizer(targetURL, "/");
    int tokencount = tokenizer.countTokens();
    int i;
    String returnHost = "";
    if (tokencount > 1) {
        for (i = 1; i < 3; i++) returnHost = tokenizer.nextToken();
    } else {
        returnHost = "";
    }
    return returnHost;
}

private boolean validateHost(int jbsid, String targetHost) {
    //false: URL - Site match found
    //true: bad URL
    boolean errFlag = false;
    BlurbsiteDB bs = new BlurbsiteDB();
    Vector v = null;
    Connection conn = dataConnMan.getConnection(server, dbname);
    try {
        v = bs.getAll(conn, "blurb_site_id =" + jbsid + " and upper(url) like '%" +
        targetHost.toUpperCase() + "%'");
    } catch (Exception e) {
        BlurbCommon.handleException(e);
        errFlag = true;
    }
    dataConnMan.freeConnection(server, dbname, conn);
    if (v.size() < 1) errFlag = true;

    return errFlag;
}

Content lookup(String contentID) {
    Content cont;
    int index = -1;
    for (int i=0; i<contents.size(); ++i) {
        cont = (Content)contents.elementAt(i);
        if (contentID.equals(cont.contentID)) return (Content)contents.elementAt(i);
    }
    return null;
}

class Content {
    String contentID, detail;
    long timestamp;
    Content(String contentID, String detail, long timestamp) {
        this.contentID = contentID;
        this.detail = detail;
        this.timestamp = timestamp/60000; //converts milliseconds to minutes
}

```

```

        }

    }

private class Sentinel extends Thread {
    int itemsToKeep;
    int minutesToLive;
    boolean quit = false;

    Sentinel(int itemsToKeep, int minutesToLive) {
        System.out.println("Sentinel Constructor ");
        this.itemsToKeep = itemsToKeep;
        this.minutesToLive = minutesToLive;
        setPriority(Thread.MIN_PRIORITY);
        start();
    }

    public void run() {
        int itemsToRemove;
        while (!quit) { // this keeps cleanup alive until servlet is destroyed
            freshen();
            if (contents.size() > itemsToKeep) trim();
            //System.out.println(contents.size());
            //System.gc();
            try {
                Thread.sleep(5000);
            }
            catch (InterruptedException ex) {
                System.err.println(ex.getMessage());
                ex.printStackTrace();
            }
            //System.out.println("Sentinel ran at:" + System.currentTimeMillis());
        }
    }

    private void trim() {
        int itemsToRemove = contents.size() - itemsToKeep;
        int trimmed = 0;
        for (int i=0; i<itemsToRemove; ++i) {
            contents.removeElementAt(i);
            //System.out.println("Removed: " + contents.elementAt(i).toString());
            Thread.yield();
            trimmed++;
        }
        if (trimmed > 10) contents.trimToSize();
    }

    private void freshen() {
        long currentTime = System.currentTimeMillis()/60000;
        Content cont;
        for (int i=0; i<contents.size(); ++i) {
            cont = (Content) contents.elementAt(i);
            if (cont.timestamp + minutesToLive < currentTime) {
                contents.removeElementAt(i);
                //System.out.println("Expired: " + cont.contentID);
                Thread.yield();
            }
        }
    }
}
}

```

85
CLAIMS

1. A method of obtaining selected content for a web page, wherein the selected content itself is not initially part of the web page, the web page including script associated with the selected content, the method comprising:
 - (a) a web browser requesting a web page that includes script associated with the selected content; and
 - (b) the web browser interpreting the script and formatting a request for obtaining the selected content from a remote site, the request including a uniform resource identifier (URI) of the web page and a unique identifier of the selected content.
2. A method of claim 1 further comprising:
 - (c) a remote site receiving the request and authenticating whether the URI is authorized to receive the selected content, and, if so, the remote site locating the selected content, and sending the selected content to the web browser; and
 - (d) the web browser assembling the initially requested web page using the selected content obtained from the remote site.
3. The method of claim 2 wherein the remote site is a web server, and the selected content is stored in a content repository connected to the web server.
4. The method of claim 2 wherein the selected content includes two or more different selected content, each selected content being used for different parts of the web page, wherein each selected content has its own script for implementing steps (b)-(d).
5. The method of claim 2 wherein in step (c), if the URI is not authorized to receive the selected content, the remote site sends a signal to the web browser that the selected content is not available, and the web browser assembles the web page without the selected content.
6. The method of claim 1 wherein the URI is a uniform resource locator (URL).
7. The method of claim 1 wherein the selected content is only a portion of the web page.
8. The method of claim 1 wherein the selected content is a digital asset.
9. The method of claim 1 wherein the selected content is an executable file.

10. The method of claim 1 wherein the script includes a subscriber identifier and a content identifier, and step (b) further comprises using the subscriber identifier and the content identifier to create the unique identifier of the selected content.

11. The method of claim 1 wherein the web page is constructed using HTML, and the script is embedded therein.

12. The method of claim 1 wherein the script is JavaScript.

13. A method of syndicating digital assets comprising:

(a) constructing a web page; and

(b) inserting into the web page script associated with at least one digital asset that is desired to be part of a fully rendered web page, wherein the script, when executed by a browser, requests the content of the digital asset from a remote site, the request including a uniform resource identifier (URI) of the web page and a unique identifier of the selected content.

14. A method of claim 13 wherein the script is JavaScript.

15. A method of claim 13 wherein the selected content is an executable file.

16. A method of claim 13 wherein the script includes a subscriber identifier and a content identifier, which, together, create the unique identifier of the selected content.

17. An article of manufacture for syndicating digital assets, the article of manufacture comprising a computer-readable medium holding computer-executable instructions for performing a method comprising:

(a) constructing a web page; and

(b) inserting into the web page script associated with at least one digital asset that is desired to be part of a fully rendered web page, wherein the script, when executed by a browser, requests the content of the digital asset from a remote site, the request including a uniform resource identifier (URI) of the web page and a unique identifier of the selected content.

18. The article of manufacture of claim 17 wherein the script is JavaScript.

19. The article of manufacture of claim 17 wherein the selected content is an executable file.

20. The article of manufacture of claim 17 wherein the script includes a subscriber identifier and a content identifier, which, together, create the unique identifier of the selected content.

21. An apparatus for syndicating digital assets comprising:

(a) means for constructing a web page; and

(b) means for inserting into the web page script associated with at least one digital asset that is desired to be part of a fully rendered web page, wherein the script, when executed by a browser, requests the content of the digital asset from a remote site, the request including a uniform resource identifier (URI) of the web page and a unique identifier of the selected content.

22. The apparatus of claim 21 wherein the script is JavaScript.

23. The apparatus of claim 21 wherein the selected content is an executable file.

24. The apparatus of claim 21 wherein the script includes a subscriber identifier and a content identifier, which, together, create the unique identifier of the selected content.

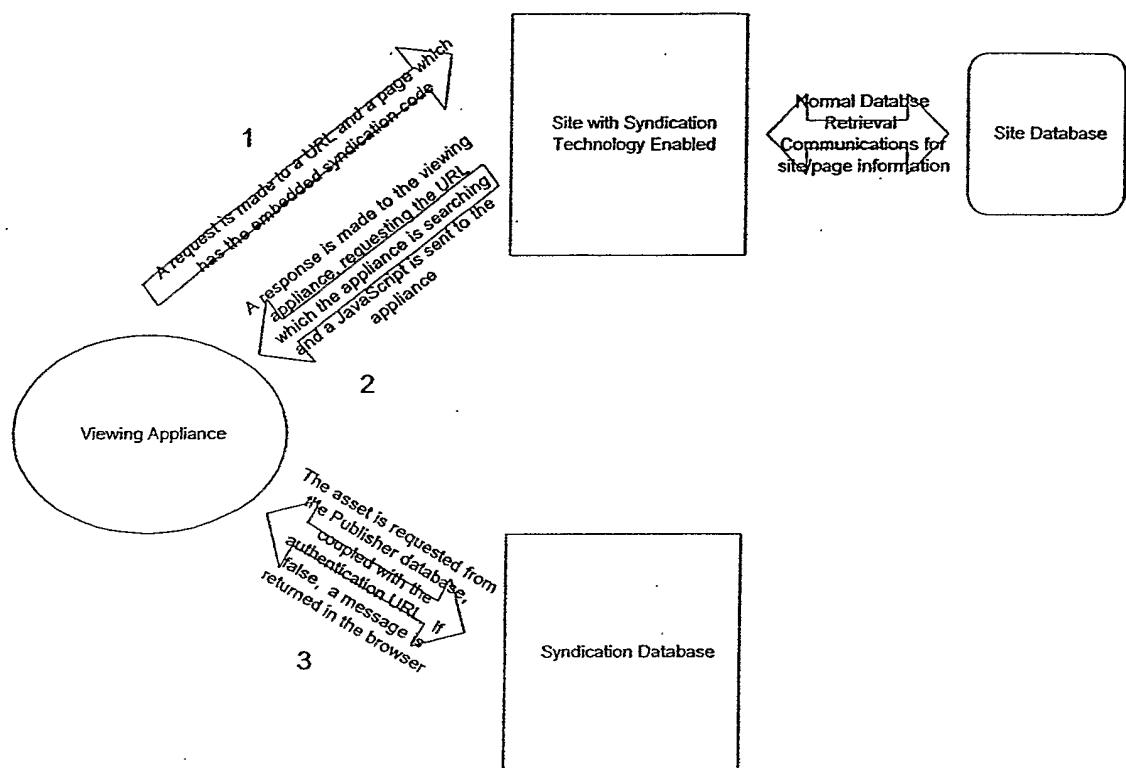


Figure 1

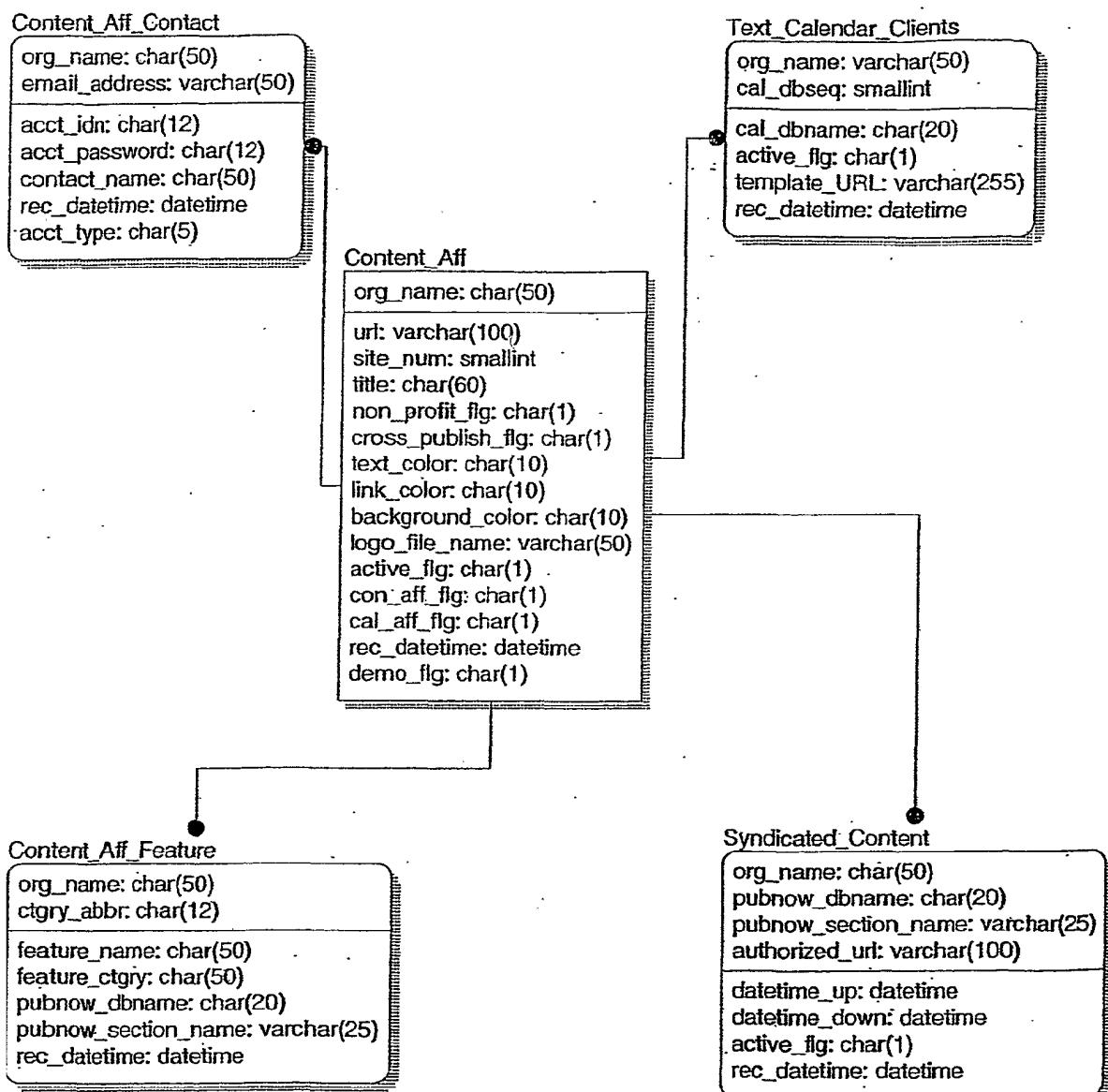


Figure 2

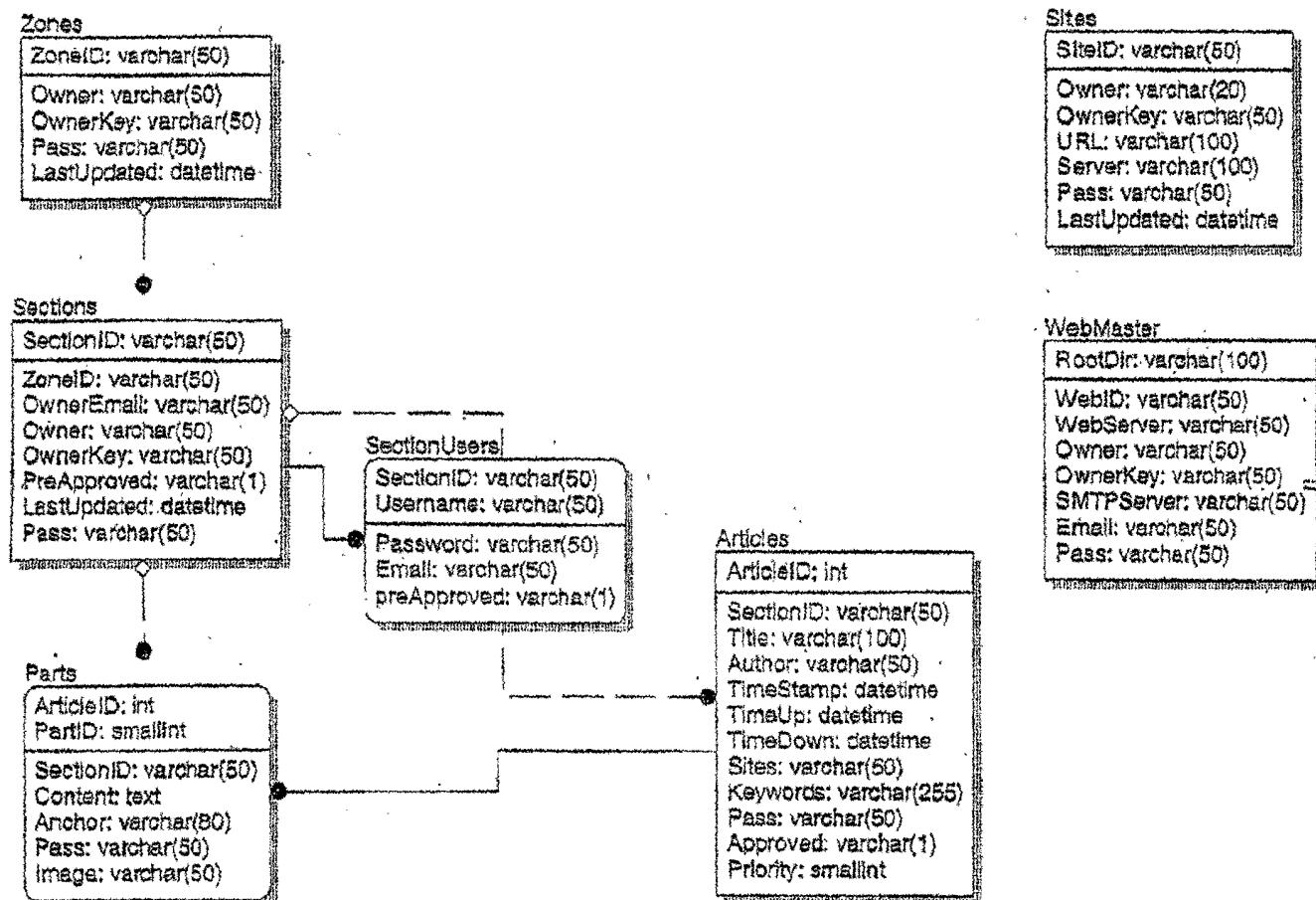


Figure 3

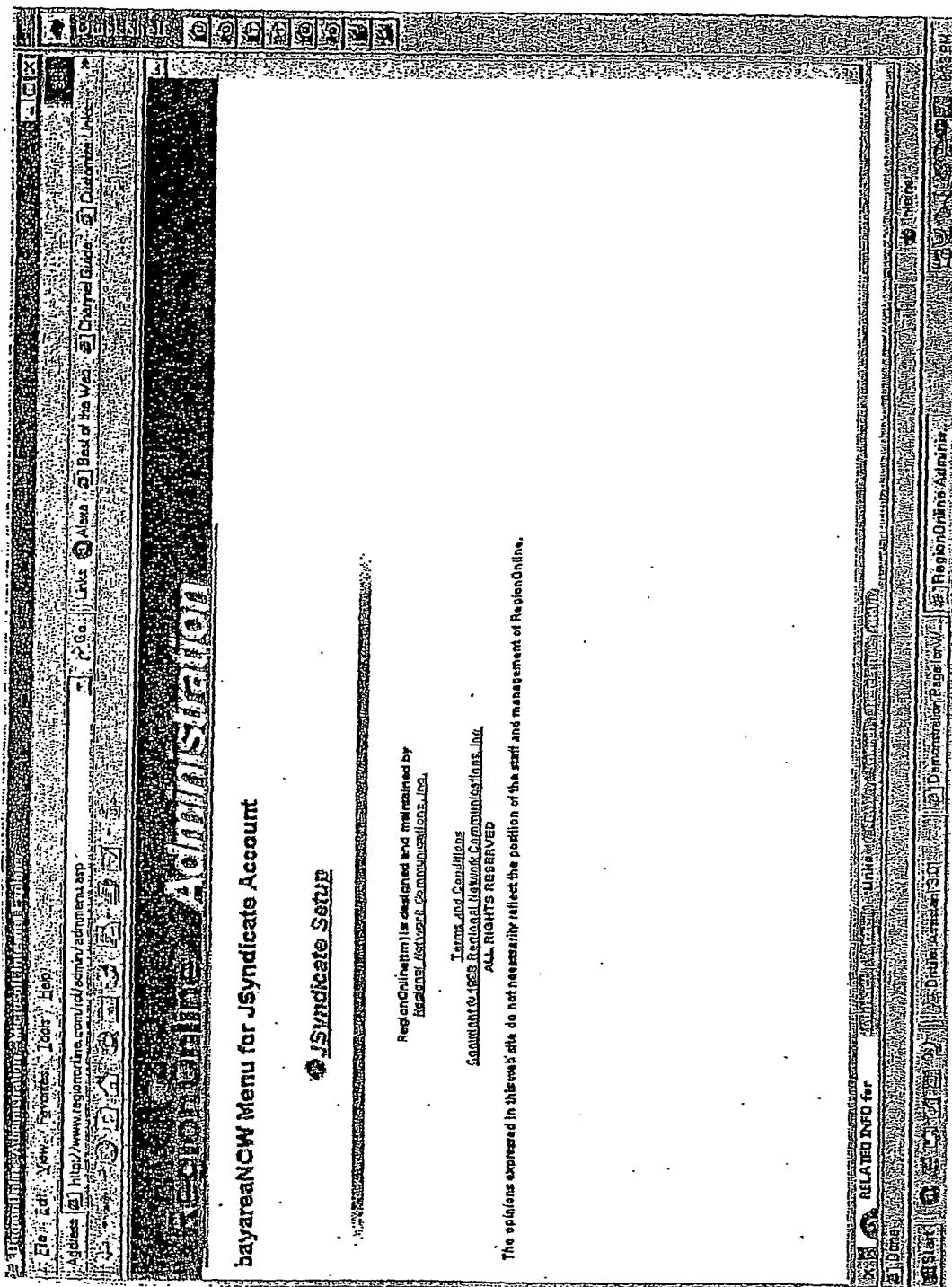


Figure 4

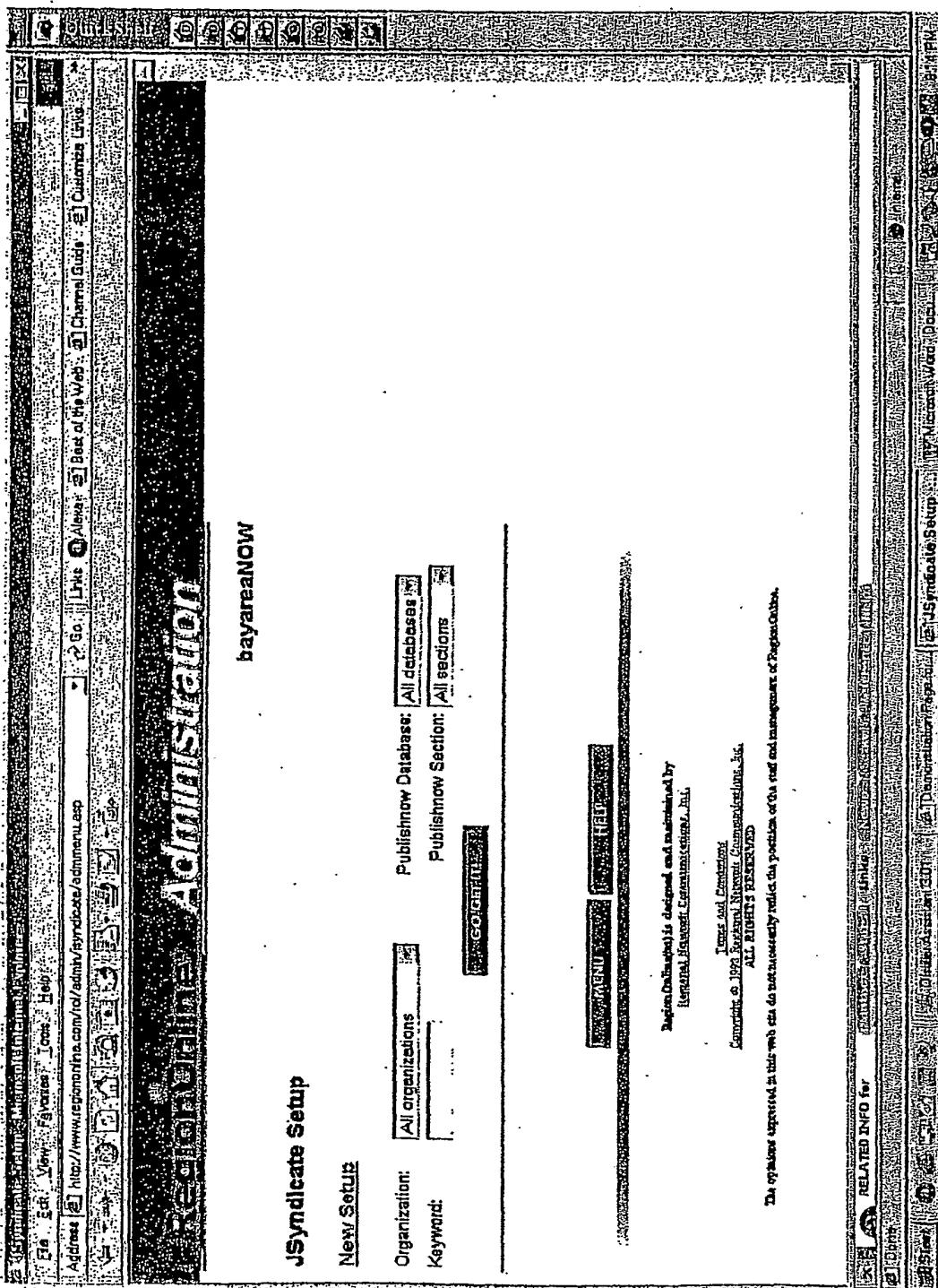


Figure 5

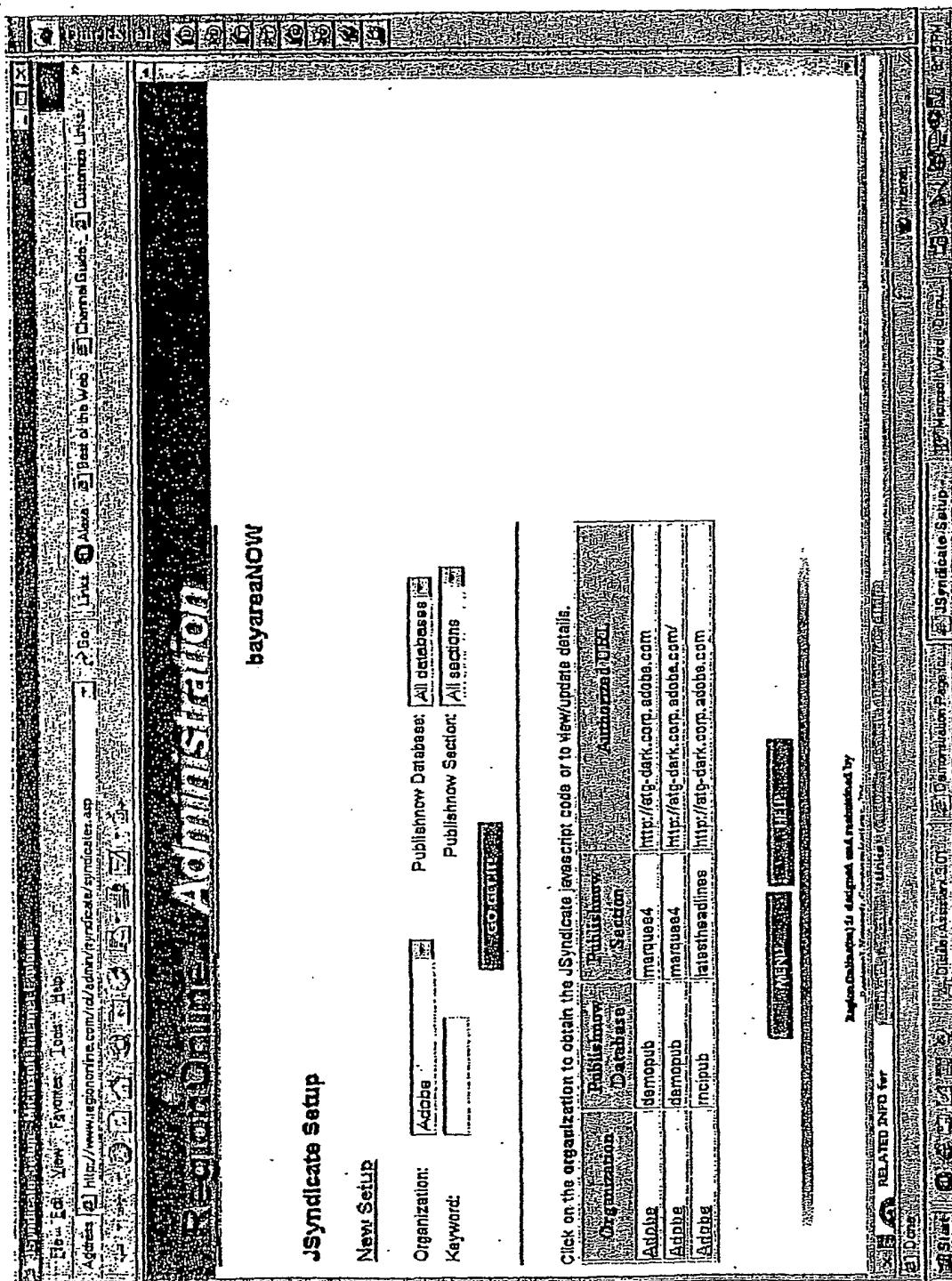


Figure 6

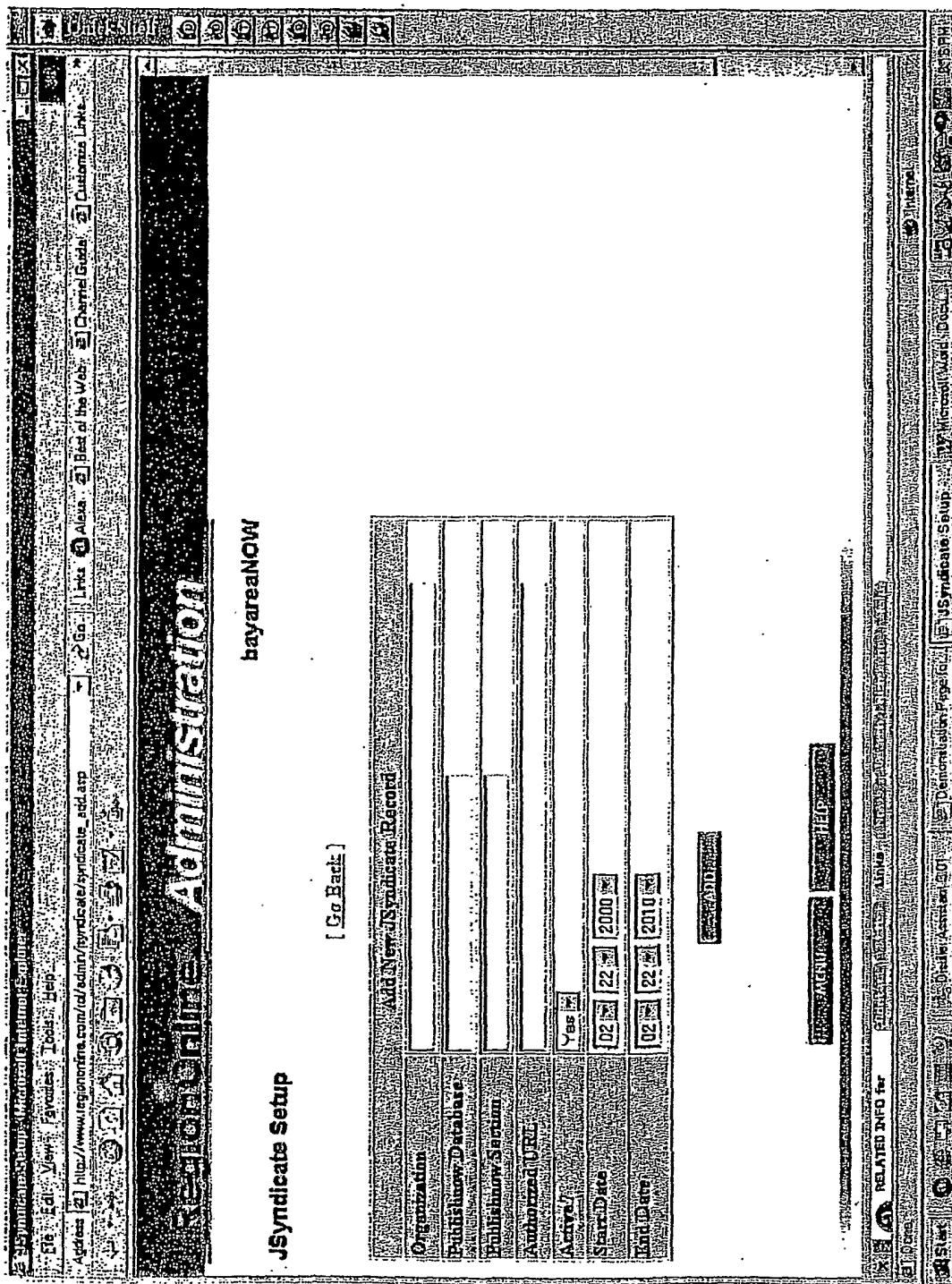


Figure 7

		Home About Us Services Tools Help	
<p>Address <input checked="" type="checkbox"/> http://www.infomatrix.com/r/admin/syndicate/add.asp</p> <p>Go Links <input checked="" type="checkbox"/> Alexa <input checked="" type="checkbox"/> Best of the Web <input checked="" type="checkbox"/> Business Links</p>			
<h1>Infomatrix Syndication</h1>			
<p>bayareaNOW</p>			
<h2>JSyndicate Setup</h2>			
<p>[Go Back]</p>			
<p>Organization</p> <p>infomatrix.com</p>	<p>Address</p> <p>http://www.infomatrix.com/r/admin/syndicate/add.asp</p>	<p>Attorney at law</p> <p>contentpub</p>	<p>Latest news</p> <p>www.your law site.com</p>
<p>Publication Date</p> <p>infomatrix.com</p>	<p>Yes</p> <p><input checked="" type="checkbox"/></p>	<p>Submit</p> <p>Submit</p>	<p>Cancel</p> <p>Cancel</p>
<p>Syndicate URL</p> <p>infomatrix.com</p>	<p>Date</p> <p>02/22/2001</p>	<p>Date</p> <p>02/22/2001</p>	<p>Date</p> <p>02/22/2001</p>
<p>RELATED INFO for</p> <p>Infomatrix.com</p>			

Figure 8

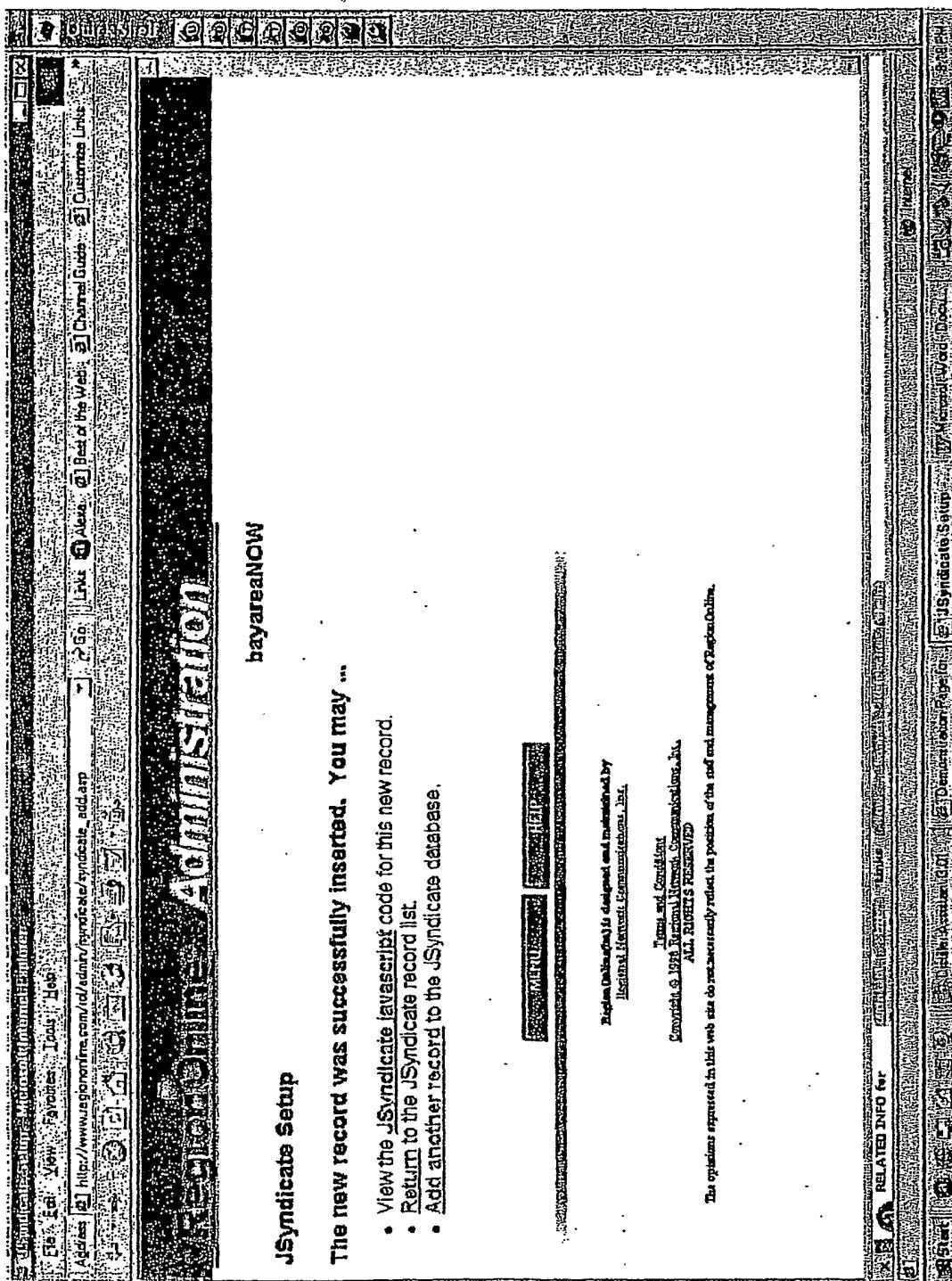


Figure 9

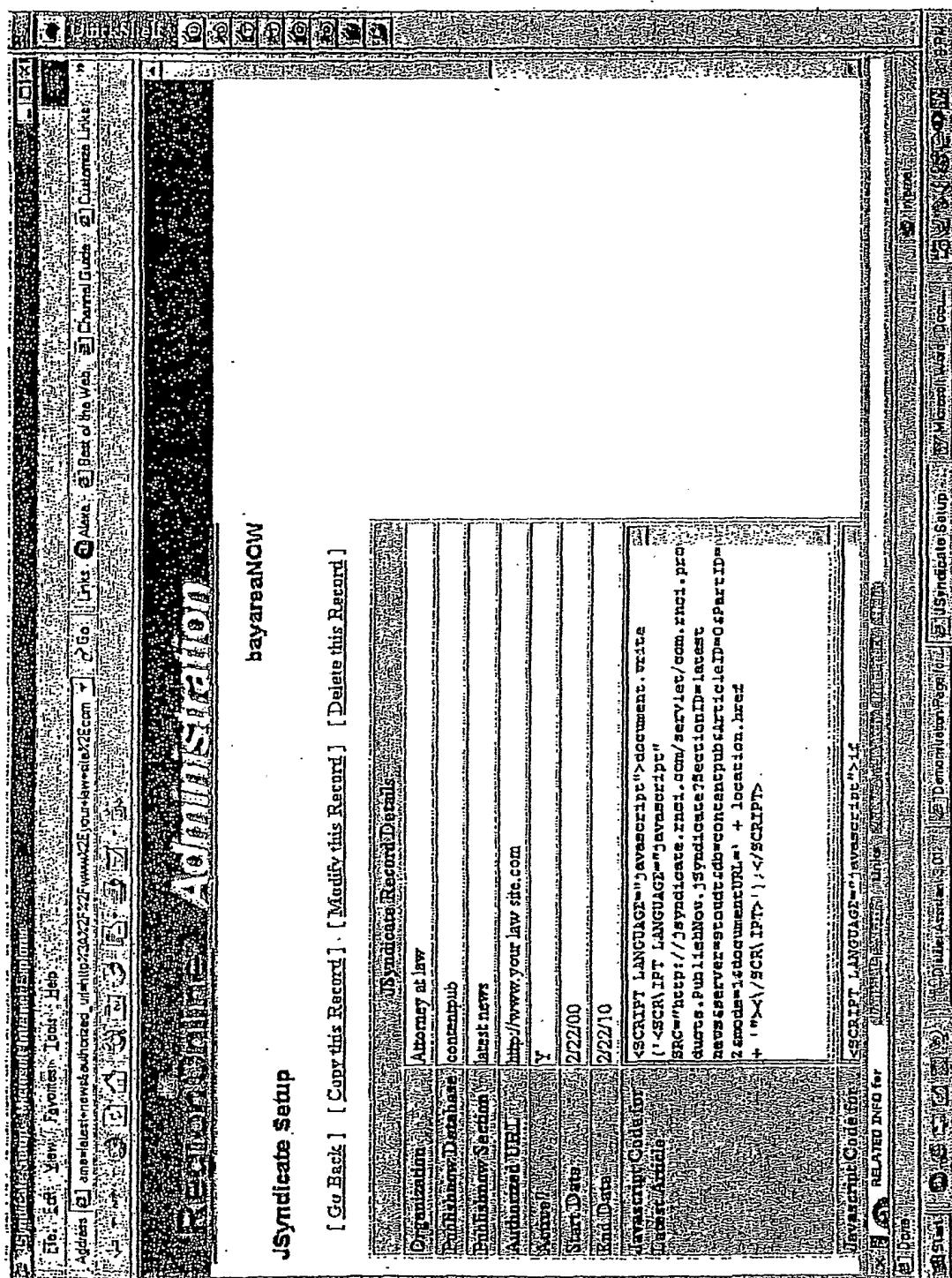


Figure 10A

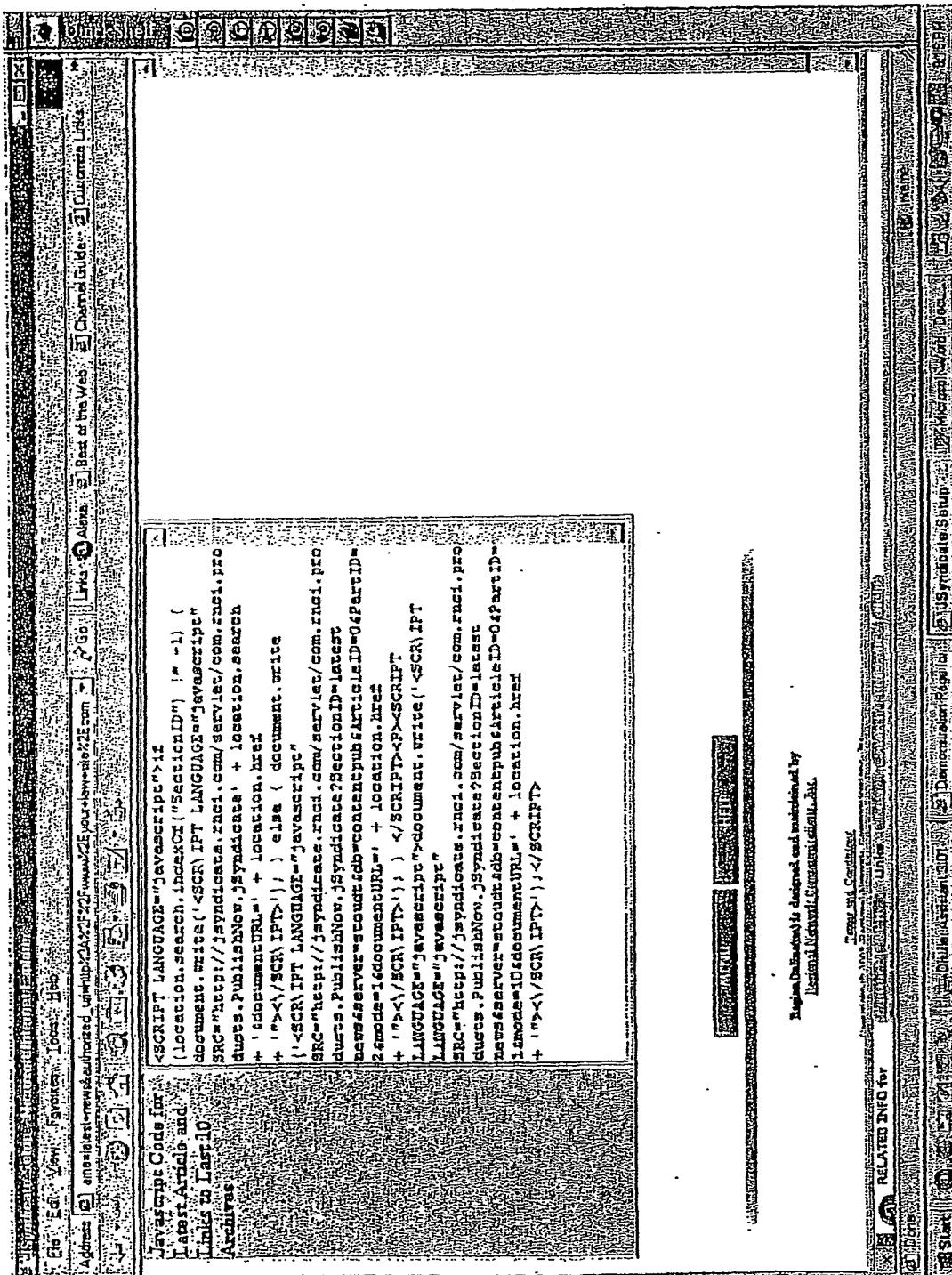


Figure 10B

12/21

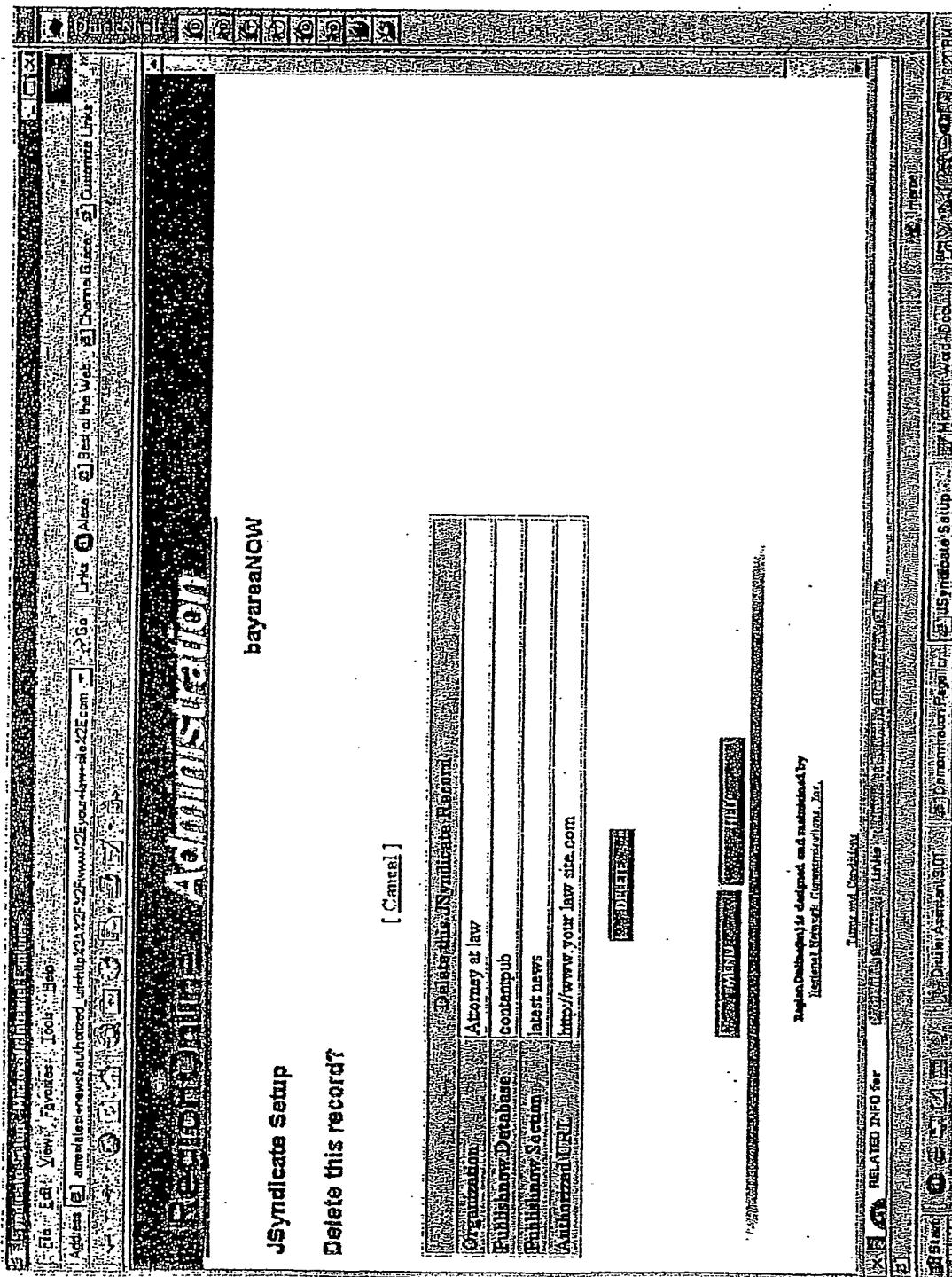


Figure 11

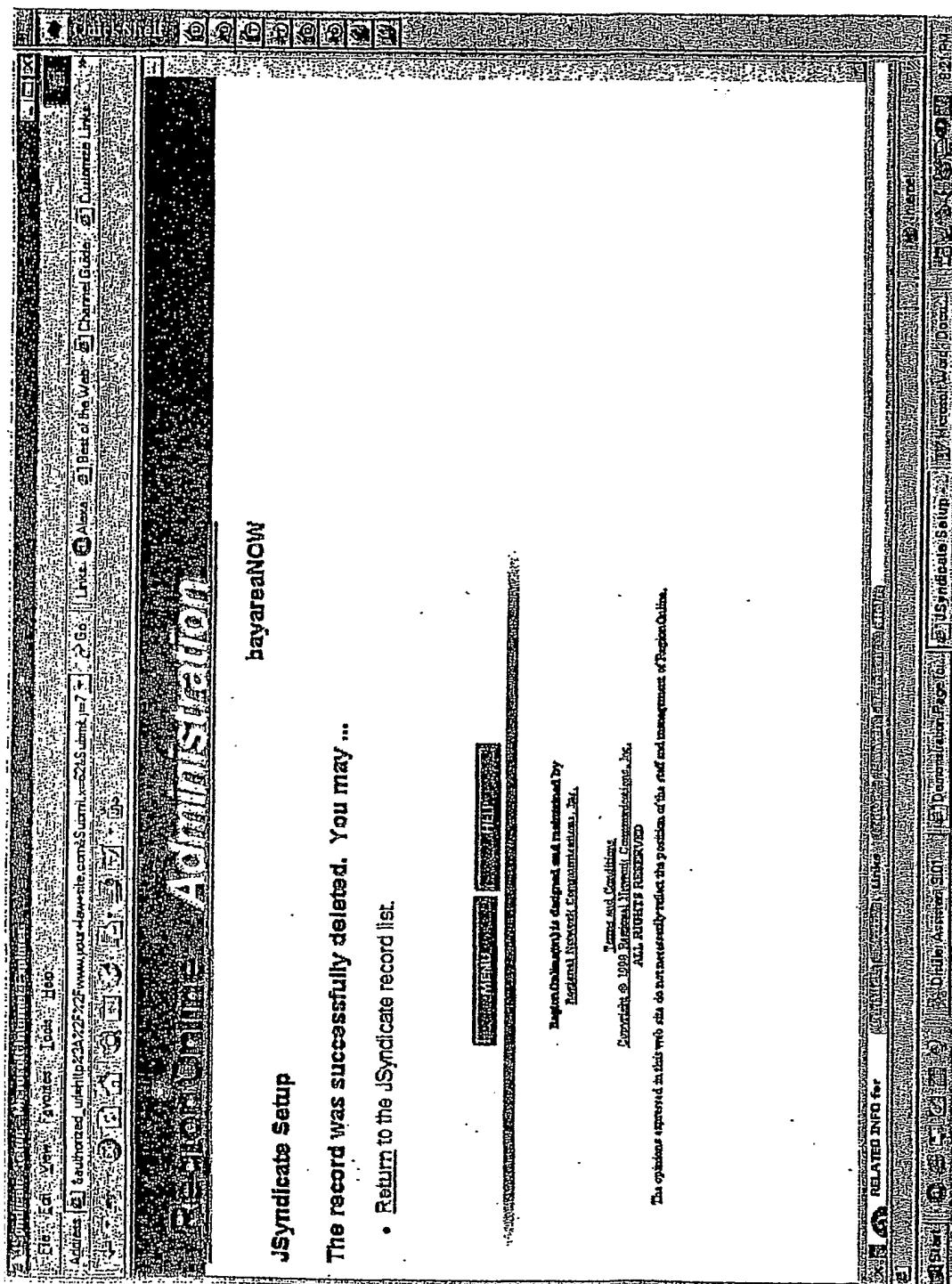


Figure 12

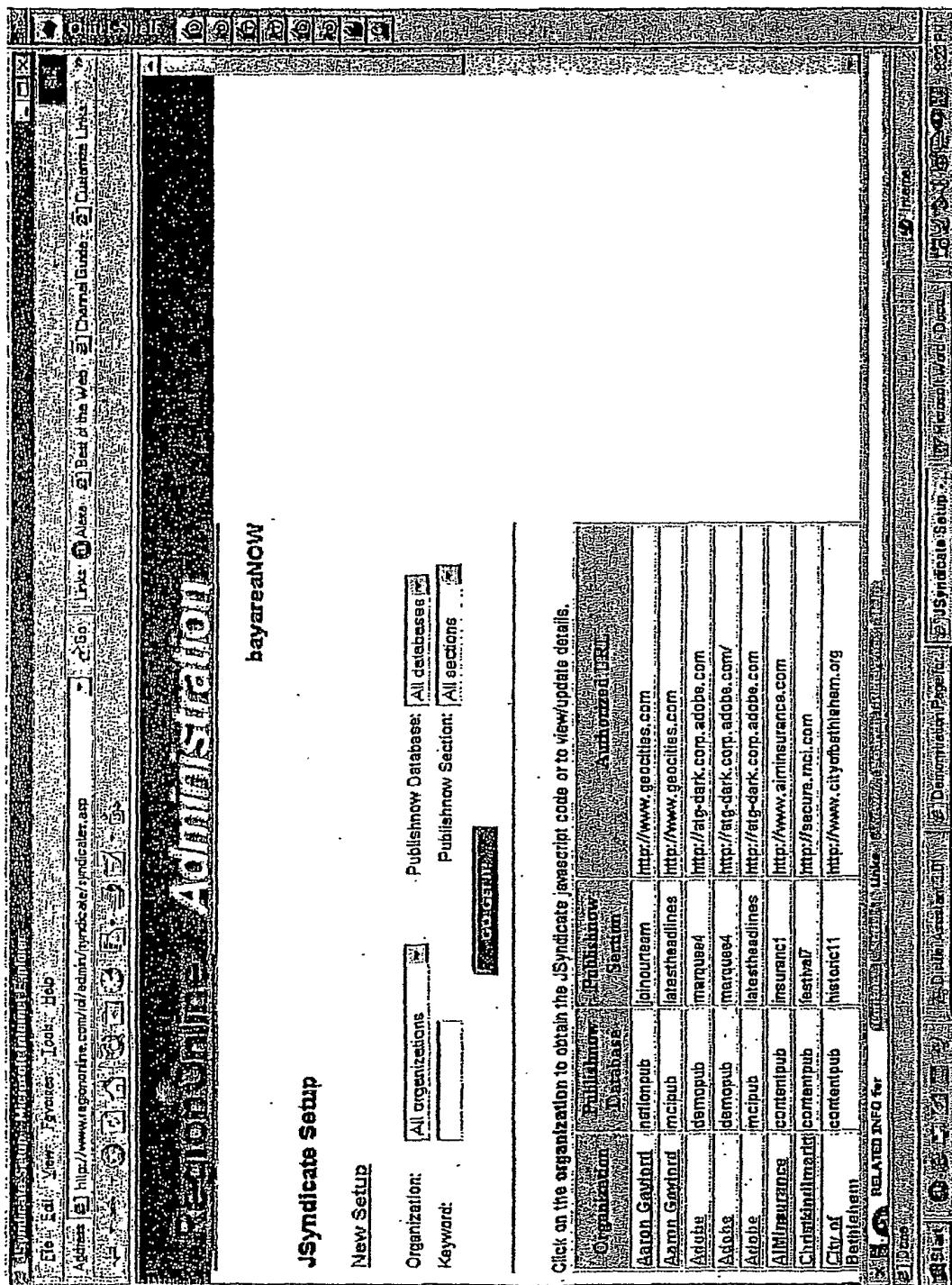


Figure 13A

Java Sources List		
Morris Black	contentpub	intenior1
Neighborhoods	contentpub	community
Online	contentpub	community0
Neighborhoods	contentpub	lateheadlines
Online	mcipub	lateheadlines
Neescape Test	mcipub	lateheadlines
OPN	contentpub	newspaper9
RNCI	inteniorpub	productraq
RNC	inteniorpub	joinourteam
RNCI	publishnow	NYdemo
RNC	doubleclick01	lateheadlines
RNC	mcipub	lateheadlines
RNCI	mcipub	lateheadlines
Summit	contentpub	commerc2
Test	demopub	married
Test	lehighupub	sectiontest2
Test	lehighupub	sectiontest2
Test	lehighupub	sectiontest2
Test	inteniorpub	joinourteam
Test	mcipub	lateheadlines
XingNet	mcipub	lateheadlines

Figure 13B

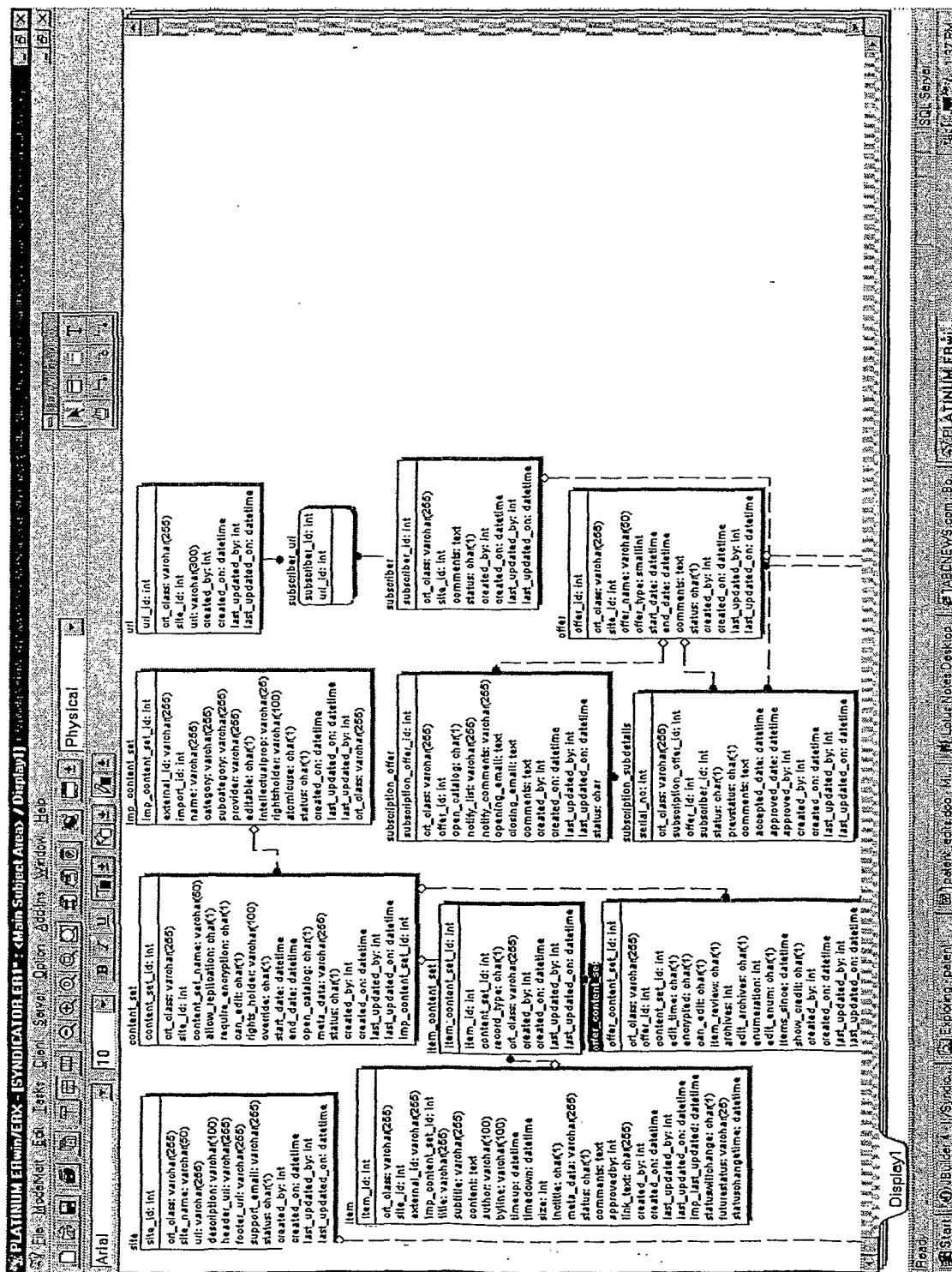


Figure 14

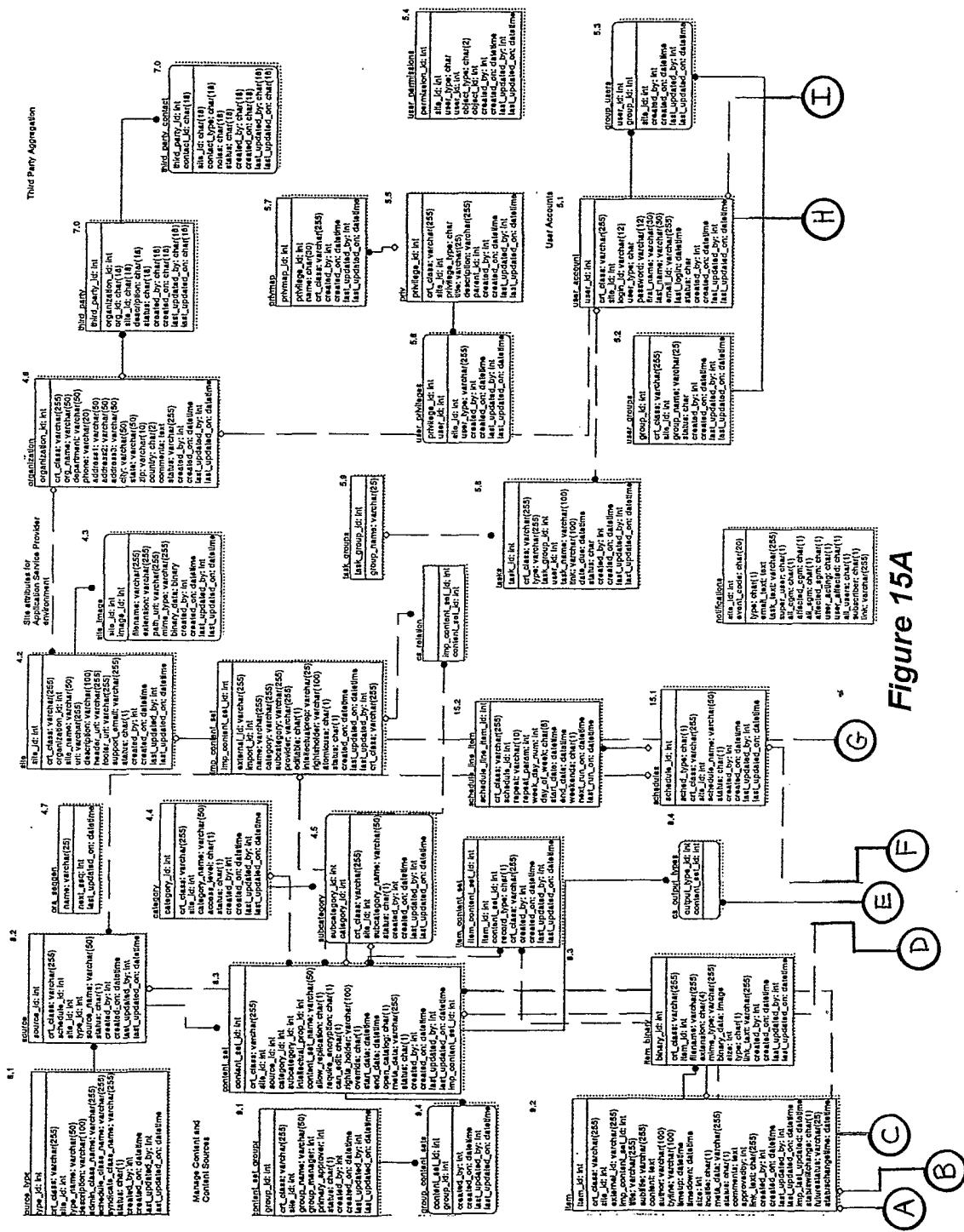


Figure 15A

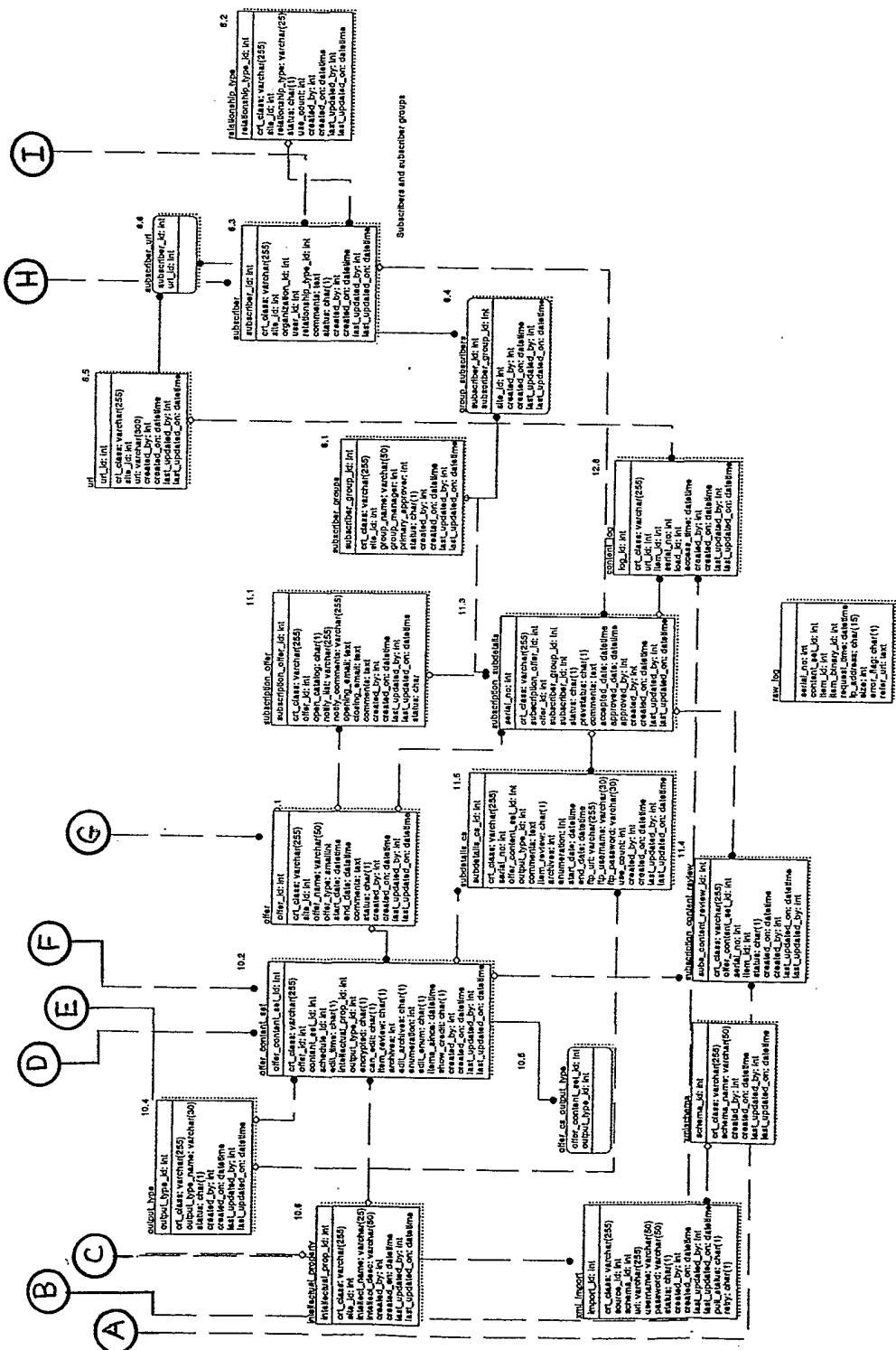


Figure 15B

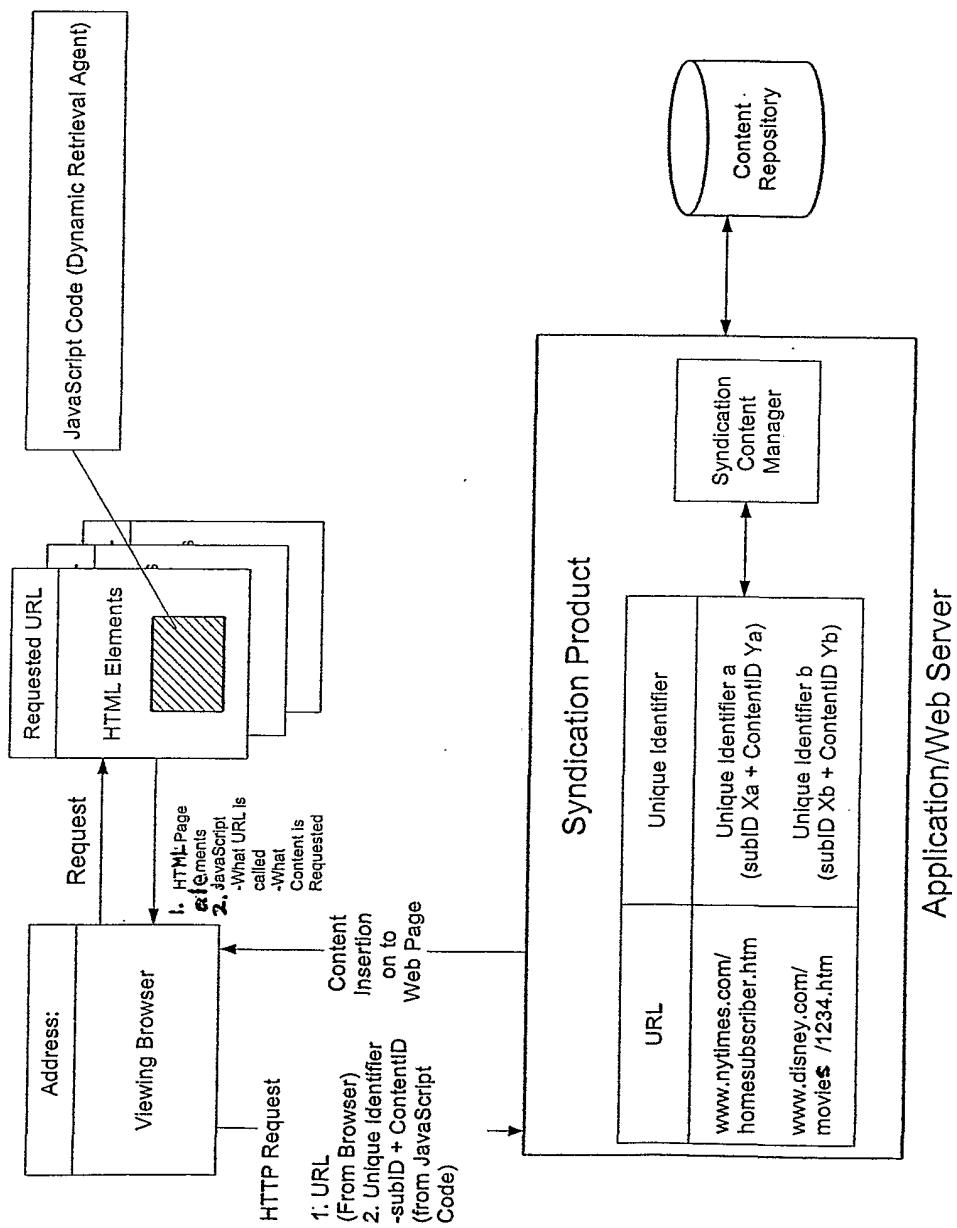


Figure 16

Subscriber ADP Example from Syndicator 3.0

```
<SCRIPT LANGUAGE="javasCript">
if (location.search.indexOf("j3serial") != -1) {
document.write('<><SCR\IPT LANGUAGE="javasCript"'
SRC='http://localhost:8080/servlet/com.activedatax.products.syndicator.export.ADPExport' +
location.search + '&j3durl=' + escape(location.href) + '<><\SCR\IPT>');
} else {
document.write('<><SCR\IPT LANGUAGE="javasCript"'
SRC='http://localhost:8080/servlet/com.activedatax.products.syndicator.export.ADPExport?j3serial='
19000025&j3sub=190000003&j3ocset=19000023&3190000017&j3itemid=0&j3ful1num=0&3A0%3A0&j3a
rcnum=10%3A10%3A10&j3newwindow=false&j3showdates=false&j3durl=' + escape(location.href) +
'&jbrand=' + escape(Math.random()) + '<><\SCR\IPT>');
}
</SCRIPT>
```

Figure 17

Admin ADP For Randomizer/Blurb Machine

```
<SCRIPT LANGUAGE="javasCript">
if (location.search.indexOf('jbact') != -1) {
document.write('<><SCR\IPT LANGUAGE="javasCript"'
SRC='http://publisher3.activedatax.com/servlet/com.activedatax.products.Blurb.JBlurbAdmin' +
location.search + '&jbdurl=' + escape(location.href) + '&jbrand=' + escape(Math.random() +
'<><\SCR\IPT>');
} else {
document.write('<><SCR\IPT LANGUAGE="javasCript"'
SRC='http://publisher3.activedatax.com/servlet/com.activedatax.products.Blurb.JBlurbAdmin?jbssid=9
31717&jbdurl=' + escape(location.href) + '&jbrand=' + escape(Math.random() + '<><\SCR\IPT>');
}
</SCRIPT>
```

Figure 18

Random Blurb Output ADP

```
<SCRIPT LANGUAGE="javasCript">
document.write('<><SCR\IPT LANGUAGE="javasCript"'
SRC='http://publisher3.activedatax.com/servlet/com.activedatax.products.Blurb.JBlurb?jbssid=931717
&jbgid=136593&jbdurl=' + escape(location.href) + '&jbrand=' + escape(Math.random() +
'<><\SCR\IPT>');
</SCRIPT>
```

Figure 19

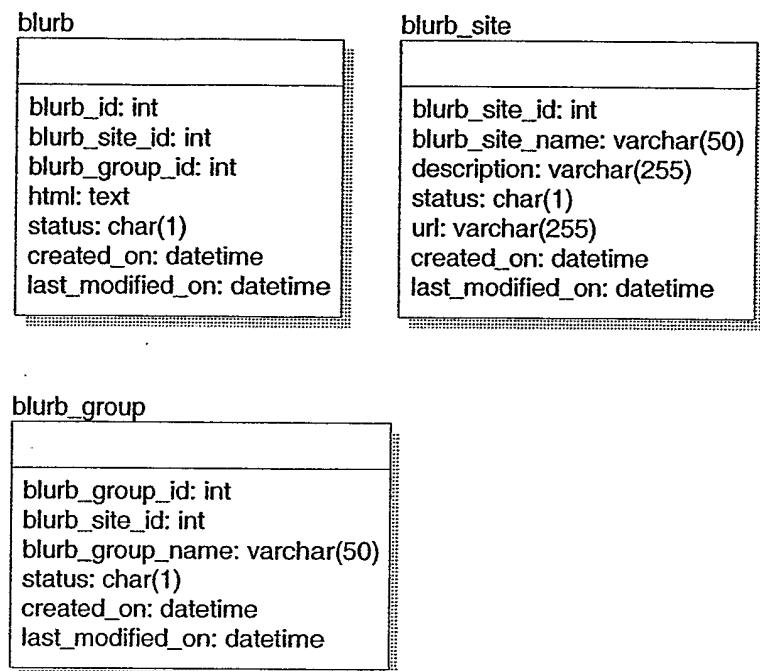


Figure 20

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US01/24675

A. CLASSIFICATION OF SUBJECT MATTER

IPC(7) :G06F 15/16, 15/173, 18/00, 18/14

US CL :709/200-203, 217-219, 225-229; 707/8, 10, 500-501, 513

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 709/200-203, 217-219, 225-229; 707/8, 10, 500-501, 513

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
NONE

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

WEST-->Search Terms: web page, script, web browser, formatting, remote site, URI, URL, web server, authenticating, HTML, JavaScript

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y, P	US 6,112,242 A (JOIS et al) 29 August 2000, abstract; figures 3-6; column 3 line 47 to column 4 line 41; column 5 line 5 to column 6 line 45; and column 8 lines 18-41 and 51-56.	1-24
Y, P	US 6,185,567 B1 (RATNARAJ et al) 06 February 2001, abstract; figures 1-4; column 7 line 24 to column 10 line 13; and column 12 lines 12-59.	1-24
A, P	US 6,157,933 A (CELI, JR. et al) 05 December 2000.	1-24
A	US 5,898,835 A (TRUONG) 27 April 1999.	1-24
A	US 5,835,712 A (DUFRESNE) 10 November 1998.	1-24

Further documents are listed in the continuation of Box C.

See patent family annex.

* Special categories of cited documents:	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A"	document defining the general state of the art which is not considered to be of particular relevance	
"E"	earlier document published on or after the international filing date	"X"
"L"	document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"Y"
"O"	document referring to an oral disclosure, use, exhibition or other means	
"P"	document published prior to the international filing date but later than the priority date claimed	"G"
		document member of the same patent family

Date of the actual completion of the international search
31 OCTOBER 2001

Date of mailing of the international search report

21 NOV 2001

Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230	Authorized officer AN, MENG-AI Telephone No. (703) 305-9678
---	---