Abstract: Template-based assembly of an electronic publication System provides a template specifying a markup for the electronic publication and having a plurality of content slots, content provider provides content, publisher formats content according to the template and comprising a language processor that automatically traverses the template and applies the content to the content slots, and electronic reading device displays formatted content. Method includes receiving template, traversing templates to identify content slot, retrieving content for insertion into identified content slot, applying markup to portion of content, formatting content by inserting portion of content into identified content slot, and rendering formatted content on an electronic reading device. Computer readable medium comprises instructions receive content for an electronic publication, receive templates, each template specifying a markup for the electronic publication and having a content slots, receive desired template selection from user, and render the content according to the selected template.
SYSTEM AND METHOD FOR TEMPLATE-BASED ASSEMBLY OF PUBLICATIONS

BACKGROUND INFORMATION

[0001] Printed and electronic publications are often associated with their unique and distinctive layouts. For example, the New York Times has a layout that is immediately recognizable ~ its articles are written in long, vertical columns and it uses distinct fonts for the name of the paper and the article headlines, bylines, and text. The USA Today newspaper has a different layout - - it has wide columns and uses a larger font. The layout of these, and other publications, can be thought of as their templates.

BRIEF DESCRIPTION OF THE DRAWINGS

[0002] The present invention, together with further objects and advantages, may best be understood by reference to the following description taken in conjunction with the accompanying drawings, in several figures of which like reference numerals identify like elements, and in which:

[0003] Fig. 1 is an illustration depicting a system for template-based assembly of publications according to one embodiment;

[0004] Fig. 2 is a schematic of an electronic reading device according to one embodiment;

[0005] Fig. 3 is an illustration depicting a system for template-based assembly of publications according to one embodiment;

[0006] Fig. 4 is a flowchart depicting a method for template-based assembly of publications according to one embodiment;

[0007] Figs. 5a-i depict example code for a template according to one embodiment;

[0008] Figs. 6a-b depict example code for a template according to one embodiment;
Figs. 7a-b depict example code for a template according to one embodiment;

Fig. 8 depicts a method for rendering an electronic publication according to one embodiment;

Fig. 9 depicts a screen shot of a display using one template according to one embodiment;

Fig. 10 depicts a screen shot of a display using one template according to one embodiment;

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

Several embodiments of the present invention and their advantages may be understood by referring to Figs. 1-10 wherein like reference numerals refer to like elements.

Referring to Fig. 1, a system for template-based assembly of publications is disclosed. System 100 includes content provider 110, template provider 120, and publisher 130. In one embodiment, content provider 110, template provider 120, and publisher 130 may be separate entities. In another embodiment, some, or all, of content provider 110, template provider 120, and publisher 130 may be part of the same entity, or may be the same entity.

Content provider 110 provides content that may be rendered on an electronic reading device. Examples of content provider 110 include newspaper publishers, periodical publishers, book publishers, Internet authors, schools, government agencies, etc.

In one embodiment, content provider 110 may be the author of the content. In another embodiment, content provider 110 may be a distributor of content.

In one embodiment, the template may specify at least one markup to be applied to the various portions of the content.
In one embodiment, advertisement provider 150 may be provided. Advertisement provider 150 may be, for example, a business entity that provides advertisements to publisher 130. Advertisement provider 150 may also be an individual. In another embodiment, advertisement provider 150 may be an aggregator of advertisements.

Template provider 120 may provide templates that may be applied to content. In one embodiment, a template may be applied to the content as it is being rendered on an electronic reading device. In another embodiment, a template may be applied to the content before it is provided to an electronic reading device. In yet another embodiment, a template may be applied to content as or just before it is printed.

In one embodiment, content provider 110 may provide a template with its content. In another embodiment, content provider 110 may also author templates for the content provided by other content providers 110.

In one embodiment, the template may have a plurality of slots for receiving content. For example, the template may include certain areas for receiving certain content, including text, photos, etc. In one embodiment, the number, size, location, etc. of the slots may depend on the layout of the template.

In one embodiment, the template may have a plurality of advertisement slots for receiving advertisements. In one embodiment, the number of advertisement slots may vary with the cost of the template. For example, a free template may have more advertisement slots than a template that a customer purchases.

In one embodiment, the content, the advertisements, and the template may be transmitted independently from content provider 110, template provider 120 and advertisement provider 150 to a publisher 130. In one embodiment, multiple publishers 150 may be provided. Final layout and assembly may take
place at a plurality of locations, including, for example, at content provider 110, at a centralized distribution server, on a client device in the field, etc.

[0024] In one embodiment, the advertisements may be streamed to publisher 130. In another embodiment, the advertisements may be targeted. For example, different advertisements may be provided for different electronic reading devices. In another embodiment, different advertisements may be provided based on location, demographics, etc. of the customer. In another embodiment, the advertisements may change as the publication is rendered. Other types of advertisements and/or techniques for displaying the advertisements may be used as necessary and/or desired.

[0025] Publisher 130 may apply the template provided by template provider 120 to the content provided by content provider 110 and the advertisements provided by advertisement provider 150. In one embodiment, publisher 130 may provide the content with the template applied to be rendered in an electronic reading device.

[0026] Publisher 130 may be able to proof and modify the look of an electronic publication without having to explicitly place each article. The content slot for each article may be programmatically determined based on the characteristics of the article. In one embodiment, characteristics of the article may include the article word count, length of the article title, length of byline, size of font, number of pictures or graphics, size of pictures or graphics, etc.

[0027] In one embodiment, the placement of articles, pictures, and advertisements may be heuristically determined.

[0028] In one embodiment, publisher 130 may include database 135 and language processor 145. Database 135 may be any suitable database. In one embodiment, database 135 may be a relational database. In another embodiment, database 135 may comprise a plurality of databases. In yet another embodiment, database 135 may be located remotely from publisher 130.

[0029] Database 135 may receive and store content and templates.
In another embodiment, advertisement provider 150 may be provided as a database to publisher 130.

In another embodiment, database 135 may be memory 210 of electronic reading device 200.

Language processor 140 may be a software application executed by a processor. In one embodiment, language processor 140 may operate on an extensible markup language (XML) dialect. In another embodiment, language processor 140 may operate on JSON. Other languages and/or syntaxes may be used as necessary and/or desired.

For example, some or all of the following elements may be part of the XML dialect:

- `<Select Trigger="field" Identifier="field">` - The data in Select nodes is assumed to be a SQL select statement. A query is made using the select statement and the rows returned are used as data to substitute in to any other nodes processed at the level of the `<Select>` tag or above. To request the substitution of a variable, the user specifies "@variable@". This will cause the current value of "variable" to be searched for in all active queries and substituted in.

- `Trigger="field"` - The Trigger property identifies a field in the database that, when referenced, triggers the saving of the current value of the field identified by the Identifier property in a list of context identifiers that are passed to the Ad Server when an Ad request is made. This allows identifiers, such as ArticleIDs, to be collected for articles referenced from within a given template. In one embodiment, an Identifier property may be provided if a Trigger property is specified. This may be specified as: Trigger="Text".

- `Identifier="field"` - The Identifier property identifies a field in the database that may be saved whenever the field identified by the Trigger property is reference in the list of identifiers that gets passed to
the Ad Server. This allows identifiers such as ArticleIDs to be collected for all articles actually referenced from within a given template. Note that a Trigger property may be provided if an Identifier property is specified. This may be specified as: Identifier" ArticleID".

[0037] <Execute> ~ Execute a SQL statement.
[0038] <ExecuteIfCreated> ~ Execute a SQL statement if the output book was successfully created.
[0039] <Head> ~ A Head node may be output to a separate stream from other tags and is placed before all the body data that is output. A Head section is only output once.
[0040] <BookID, Title, Author, Category, Encrypt> ~ These sub-elements may be stored as variables that can be referred to by the engine when creating books.
[0041] <Img> - The Img element may provide the same options as the XHTML <img> element. In addition, a ScaleToFit="Y" attribute may be specified. When this attribute is specified, the actual size of the graphics may be determined and the width and height attribute values may be replaced with a width and height that will preserve the aspect ratio of the inserted image within the originally-specified width and height rectangle. In one embodiment, the width and height should be specified in pixels.
[0042] <Ad Width="x" Height="y" other_img_properties /> -- The Ad element specifies a location at which to insert an advertisement from the Ad Server. NewsManager may generate a request to the Ad Server for each Ad element encountered. Each Ad Server request may include the context in which the ad will be used. The Ad Server may return an appropriate advertisement. News Manager may retrieve the supplied advertisement graphic and store it along with the rest of the publication's information on the News Manager server.
[0043] Width="x" ~ Specifies the width of the area that the advertisement will be inserted into. The value may be specified using any dimensional units (mm, px, pt, in, etc.).

[0044] Height="y" ~ Specifies the height of the area that the advertisement will be inserted into. The value may be specified using any dimensional units (mm, px, pt, in, etc.).

[0045] In addition to the Width and Height properties, other properties may be specified. This allows such function as ScaleToFit to be used in conjunction with Ad elements.

[0046] <NewAdContext /> ~ The NewAdContext element may cause pending Ad elements to be processed by issuing requests to the Ad server for each of them. The context information passed to the Ad Server will include all article identifiers that have been accumulated since the previous NewAdContent element or the beginning of the edition, whichever is most recent. This gives publication designers a way of identifying which articles should be considered by the Ad Server when determining which Ad to provide. The context identifier list is cleared whenever a NewAdContent element is processed.

[0047] <If> ~ The If element may allow one to conditionally process the contained elements. Each attribute name may be assumed to be a variable name as if it were enclosed in "@" characters. The variable may be looked up and compared to the data (with variable substitution) specified as the attribute value. If there is equality for all attributes/values specified, then the element may be processed, otherwise the element will be ignored.

[0048] It is also possible to specify the special attributes "And" and/or "Or" to change the processing from requiring all comparisons to be equal to only requiring a single comparison to be equal.
<IfOption> ~ The IfOption element functions in exactly the same manner as the <If> element except that the values that it queries may be preceded by "__" in the hashtable for there to be a match. This separates them from the rest of the variables and field values. This allows the user accessible options to be identified in templates.

A number of special variables can be referenced within values to provide specialized control:

- Previous — The previous row value of the specified column.
- PreviousIgnoreBlanks ~ The previous row value of the specified column ignoring any rows that have the specified column as blank or null.
- UniqueID ~ The numeric ID that is unique for each thread in the system. It is constructed by taking the Process ID and Thread ID and combining them.
- FirstRow ~ Comparison to the variable @FirstRow@ will be true if the current row number specified in the retrieved data is 0.

A "!" character may be specified as the first character of an attribute value to negate the comparison. For example, CustomerID="!3" would be true if the CustomerID field in the database was not 3.

<Include Src="file" /> ~ The Include element replaces itself with the file referenced by its Src attribute.

<Set variable- "text" /> ~ Set the internal variable to the value specified in the text

<Unset Variables- "regular expression" /> -- Deletes all variables that match the regular expression provided. This allows easy reversion to one or more values that have previously been set using the <Set> command.

<Evaluate variable l="variable2" /> ~ Sets variable l to the value of the variable2 that is looked up.
The Filter element allows modification of a data field prior to its output into the data stream. The processing depends on the attributes specified.

Current attribute options include:

- **Data=** Specifies the field \( x \) in the database to use for text processing.
- **Start=** Ignore the first \( x \) characters of the data supplied.
- **Length=** Truncate the supplied data to \( x \) characters.
- **ReplaceLinefeed=** Replace all linefeed characters with the string \( x \).
- **TranslateFrom=** Search for characters matching the characters in the specified string and return the corresponding characters in the **TranslateTo** string.
- **TranslateTo=** Search for characters matching the characters in the specified string and return the corresponding characters in the **TranslateTo** string.

In addition to all of the selected fields and the following special variables are available for reference using the **@variable@** syntax in the template file:

- **Date** - The date that is being built for. This may be in the format "01/01/2006"
- **DD** -- The day of the month in the form "01"
- **D** ~ The day of the month in the form " 1"
- **MMMM** ~ The month in the form "January"
- **MMM** -- The month in the form "Jan"
- **MM** ~ The month in the form "01" for January
- **Jvl** ~ The month in the form "1" for January
- **YYYY** -- The year in the form "2006"
The template may be used to determine where to flow the different portions of content (an article, for example), and what XHTML to include for
the various portions of that content (e.g., title, by-line, pictures, etc.). The output is an EPUB document.

[0096] Referring to Fig. 2, a block diagram of an electronic reading device is provided. In general, electronic reading device 200 may include microprocessor 205, memory 210, software and applications 215, display and display driver 220, user interface 225, power supply 230, data interface 235, and audio 240. Other features may be provided as necessary and/or desired.

[0097] Processor 205 provides the processing power for electronic reading device 200. Processor 205 may be any suitable processor or integrated circuit, including microprocessors, programmed microprocessors, micro-controllers, peripheral integrated circuit elements, CSICs (Customer Specific Integrated Circuit) or ASICs (Application Specific Integrated Circuit), logic circuits, digital signal processors, programmable logic devices such as FPGAs, PLDs, PLAs or PALs, or any other device or arrangement of devices that is capable of performing the function of a microprocessor.

[0098] Memory 210 may be any suitable memory, and may be used to store software and applications 215. Memory 210 may be volatile or non-volatile as necessary and/or desired. Memory 210 may include static RAM, dynamic RAM, flash memory, magnetic memory, etc.

[0099] In one embodiment, memory 210 may store data files, including electronic content files, style sheets, etc. In one embodiment, software and applications 215 render an image on the display of electronic reading device 200.

[0100] Display and drivers 220 are provided for displaying characters, graphics, videos, pictures, etc. for the user. The display may be any suitable display, including flexible displays, touch-sensitive displays, etc.

[0101] User interface 225 may be provided for the user to interact with electronic reading device 200. Any suitable input mechanism may be provided. In one embodiment, buttons may be provided. In another embodiment, as
discussed above, a touch-sensitive screen may be provided. In still another embodiment, a microphone may be provided to detect speech. In another embodiment, a camera may be provided. Other inputs may be provided as necessary and/or desired, depending on the application.

In one embodiment, the user may use user interface 225 to zoom on the content.

Electronic reading device 200 may be powered by power supply 230, such as a battery, AC power, DC power through an AC adapter, etc.

Electronic reading device 200 may also be provided with data interface 235. In one embodiment, data interface may be any suitable wired or wireless communication method, including cellular communications, a USB interface, unshielded twisted pair interface, radio frequency (RF), infrared (IR), Bluetooth, near field communication, WiFi (e.g., any suitable IEEE 802.11 protocol), etc.

In one embodiment, electronic reading device 200 may be provided with audio capability 240.

In one embodiment, electronic reading device 200 may include language processor 245. Language processor 245 may be similar to language processor 140, discussed above. Language processor 245 may combine the raw content data, including, for example, raw articles, images, titles, etc., with a template to render the content for the user. Language processor 245 will be described in greater detail below.

Examples of electronic reading devices include electronic book readers, electronic newspaper readers, electronic paper readers, etc. An example of an electronic display device is disclosed in U.S. Patent Appl'n Ser. No. 12/481,677, entitled "Flexible Electronic Device And Method Of Manufacture," the disclosure of which is incorporated by reference in its entirety. Other examples of electronic reading devices include the Amazon Kindle, the Sony Reader, and the Plastic Logic Reader.
Other examples of electronic reading devices include computers (desktop, notebook, laptop, tablet, etc.), personal digital assistants (PDAs), music players, telephones, etc. Any electronic reading device that is capable of rendering electronic content may be used as necessary and/or desired. For example, any electronic device having a screen or a display may be used.

Referring to Fig. 3, a system for template-based assembly of publications is disclosed. System 300 may include content provider 110, template provider 120, and advertisement provider 150, all of which have been described above. System 300 further includes electronic reading device 200.

Referring to Fig. 4, a method for preparing electronic publications automatically is disclosed. In step 410, a template is received. In one embodiment, the template specifies the markup to be applied to the content. In one embodiment, the template may use the XML dialect described above.

In one embodiment, a user may also select from a plurality of templates. For example, a user may like the traditional layout of the New York Times newspaper, and is comfortable with that format. Thus, the user may apply a template for the New York Times to another publication.

Referring to Figs. 5a-5b, an example template similar to that of the New York Times is provided. This template refers to two files ~ NYTimes.inc and NYTimesCSS.inc. These include files, examples of which are provided as Figs. 6a-b and 7a-b, respectively.

According to another embodiment, templates for other orientations, such as a horizontal layout and a vertical layout, may be provided. Templates for other newspapers, for example, the Washington Post, the LA Times, USA Today, the Wall Street Journal, etc. may be provided as necessary and/or desired.

In one embodiment, the templates may be stored in the device memory for the electronic reading device.
Referring again to Fig. 4, in step 410, content is received. Any sort of content may be provided. In one embodiment, content for an electronic publication may be provided. In another embodiment, content for an electronic book may be provided. In yet another embodiment, digital photographs may be provided. Any content that may be displayed for a user on an electronic display device may be used.

In step 420, advertisements may be provided from an advertisement provider. In another embodiment, advertisements may be retrieved from a database. In another embodiment, advertisements may be provided and assembled by the client. In another embodiment, the advertisements may be streamed. Other ways of providing the advertisements may be used as necessary and/or desired.

In step 425, the language processor applies the template to the content. In one embodiment, the language processor retrieves raw data from the database according to the markup specified in the template.

In step 430, the publication is displayed according to the template.

Referring to Fig. 8, a method for rendering an electronic publication according to one embodiment is provided.

In step 810, a plurality of templates are received. In one embodiment, the templates may be received from a content provider. In another embodiment, the templates may be received from a database. In still another embodiment, the templates may be received from a storage medium.

In step 815, content is received. In one embodiment, content may be received from a content provider. In another embodiment, content may be received from a database. In still another embodiment, content may be received from a storage medium.

In step 820, advertisements may be provided from an advertisement provider. In another embodiment, advertisements may be retrieved from a database. In another embodiment, advertisements may be provided and
assembled by the client. In another embodiment, the advertisements may be streamed. Other ways of providing the advertisements may be used as necessary and/or desired.

[00123] In step 825, the user may select a template that it wishes to have applied to the content. In one embodiment, this may be done through any suitable input device on the electronic reading device.

[00124] In step 830, the template is applied to the content. In one embodiment, this may be performed as discussed above.

[00125] In step 835, the publication is rendered in accordance with the selected template.

[00126] Referring to Figs. 9-10, examples of templates applied to raw content are provided. In the example of Fig. 9, the template is a New York Times template in which a portion of an article is provided at the top, while portions of other articles are provided below. In the example of Fig. 10, an entire article is provided.

[00127] According to one embodiment, the layout method may provide resizable, resolution-independent publications. This provides a reading experience with user selectable font sizes.

[00128] According to one embodiment, the approach combines and leverages the power of database query languages, including, for example, SQL, and markup languages, for example, EPUB.

[00129] According to one embodiment, there may be cost savings realized by being able to automatically layout publications that have the appearance of hand designs.

[00130] According to one embodiment, the invention provides the ability to create attractive publications on a device using personalized information for each user while still maintaining the ability to control the layout of the information from a centralized server.
[00131] A system for template-based assembly of an electronic publication is disclosed. According to one embodiment, the system includes a template provider that provides a template, the template specifying a markup for the electronic publication and having a plurality of content slots; a content provider that provides content; a publisher that formats the content according to the template, the publisher comprising a language processor that automatically traverses the template and inserts the content into the plurality of content slots; and an electronic reading device that displays the formatted content.

[00132] A method for template-based automatic assembly of an electronic publication is disclosed. According to one embodiment, the method includes (1) receiving at least one template from a template provider; (2) traversing one of the at least one templates to identify a content slot; (3) retrieving content for insertion into the identified content slot; (4) applying a markup to at least one portion of the content; (5) formatting the content by inserting at least a portion into the content into the identified content slot; and (6) rendering the formatted content on an electronic reading device.

[00133] A computer readable medium is disclosed. According to one embodiment, the computer readable medium comprises instructions to perform the steps of: receive content for an electronic publication; receive a plurality of templates, each template specifying a markup for the electronic publication and having a plurality of content slots; receive a desired template selection from a user; and render the content according to the selected template.

[00134] The system of the invention or portions of the system of the invention may be in the form of a "processing machine," such as a general purpose computer, for example. As used herein, the term "processing machine" is to be understood to include at least one processor that uses at least one memory. The at least one memory stores a set of instructions. The instructions may be either permanently or temporarily stored in the memory or memories of the processing machine. The processor executes the instructions that are stored in the memory
or memories in order to process data. The set of instructions may include various instructions that perform a particular task or tasks, such as those tasks described above in the flowcharts. Such a set of instructions for performing a particular task may be characterized as a program, software program, or simply software.

[00135] As noted above, the processing machine executes the instructions that are stored in the memory or memories to process data. This processing of data may be in response to commands by a user or users of the processing machine, in response to previous processing, in response to a request by another processing machine and/or any other input, for example.

[00136] The processing machine used to implement the invention may utilize a suitable operating system. Thus, embodiments of the invention may include a processing machine running the Microsoft Windows™ Vista™ operating system, the Microsoft Windows™ XP™ operating system, the Microsoft Windows™ NT™ operating system, the Windows™ 2000 operating system, the Unix operating system, the Linux operating system, the Xenix operating system, the IBM AIX™ operating system, the Hewlett-Packard UX™ operating system, the Novell Netware™ operating system, the Sun Microsystems Solaris™ operating system, the OS/2™ operating system, the BeOS™ operating system, the Macintosh operating system, the Apache operating system, an OpenStep™ operating system or another operating system or platform.

[00137] As described above, a set of instructions may be used in the processing of the invention. The set of instructions may be in the form of a program or software. The software may be in the form of system software or application software, for example. The software might also be in the form of a collection of separate programs, a program module within a larger program, or a portion of a program module, for example. The software used might also include modular programming in the form of object oriented programming. The software tells the processing machine what to do with the data being processed.
Further, it is appreciated that the instructions or set of instructions used in the implementation and operation of the invention may be in a suitable form such that the processing machine may read the instructions. For example, the instructions that form a program may be in the form of a suitable programming language, which is converted to machine language or object code to allow the processor or processors to read the instructions. That is, written lines of programming code or source code, in a particular programming language, are converted to machine language using a compiler, assembler or interpreter. The machine language is binary coded machine instructions that are specific to a particular type of processing machine, i.e., to a particular type of computer, for example. The computer understands the machine language.

Any suitable programming language may be used in accordance with the various embodiments of the invention. Illustratively, the programming language used may include assembly language, Ada, APL, Basic, C, C++, COBOL, dBase, Forth, Fortran, Java, Modula-2, Pascal, Prolog, REXX, Visual Basic, and/or JavaScript, for example. Further, it is not necessary that a single type of instruction or single programming language be utilized in conjunction with the operation of the system and method of the invention. Rather, any number of different programming languages may be utilized as is necessary and/or desirable.

Also, the instructions and/or data used in the practice of the invention may utilize any compression or encryption technique or algorithm, as may be desired. An encryption module might be used to encrypt data. Further, files or other data may be decrypted using a suitable decryption module, for example.

In the system and method of the invention, a variety of "user interfaces" may be utilized to allow a user to interface with the processing machine or machines that are used to implement the invention. As used herein, a user interface includes any hardware, software, or combination of hardware and software used by the processing machine that allows a user to interact with
the processing machine. A user interface may be in the form of a dialogue screen for example. A user interface may also include any of a mouse, touch screen, light pen, keyboard, voice reader, voice recognizer, dialogue screen, menu box, list, checkbox, toggle switch, a pushbutton or any other device that allows a user to receive information regarding the operation of the processing machine as it processes a set of instructions and/or provides the processing machine with information. Accordingly, the user interface is any device that provides communication between a user and a processing machine. The information provided by the user to the processing machine through the user interface may be in the form of a command, a selection of data, or some other input, for example.

[00142] It will be readily understood by those persons skilled in the art that the present invention is susceptible to broad utility and application. Many embodiments and adaptations of the present invention other than those herein described, as well as many variations, modifications and equivalent arrangements, will be apparent from or reasonably suggested by the present invention and foregoing description thereof, without departing from the substance or scope of the invention.

[00143] In the preceding specification, various preferred embodiments have been described with references to the accompanying drawings. It will, however, be evident that various modifications and changes may be made thereto, and additional embodiments may be implemented, without departing from the broader scope of invention as set forth in the claims that follow. The specification and drawings are accordingly to be regarded in an illustrative rather than restrictive sense.
WE CLAIM:

1. A system for template-based assembly of an electronic publication, comprising:
   a template provider that provides a template, the template specifying a markup for the electronic publication and having a plurality of content slots;
   a content provider that provides content;
   a publisher that formats the content according to the template, the publisher comprising a language processor that automatically traverses the template and applies the content to the at plurality of content slots; and
   an electronic reading device that displays the formatted content.

2. The system of claim 1, further comprising:
   an advertisement provider that provides at least one advertisement;
   wherein the template further comprises advertisement slots, and the language processor applies the advertisements to the advertisement slots of the template.

3. The system of claim 1, wherein the language processor uses an XML dialect.

4. The system of claim 1, wherein the publisher is part of the electronic device.

5. The system of claim 1, wherein the content provider and the template provider are the same entity.

6. The system of claim 1, wherein the content comprises at least one newspaper article.
7. The system of claim 2, wherein the at least one advertisement is streamed to the publisher.

8. A method for template-based automatic assembly of an electronic publication, comprising:
   receiving at least one template from a template provider;
   traversing one of the at least one templates to identify a content slot;
   retrieving content for insertion into the identified content slot;
   applying a markup to at least one portion of the content;
   formatting the content by inserting at least a portion of the content into the identified content slot; and
   rendering the formatted content on an electronic reading device.

9. The method of claim 8, further comprising:
   traversing the template to identify an advertisement slot;
   receiving an advertisement for insertion into the identified advertisement slot; and
   wherein the step of formatting the content by inserting at least a portion of the content into the identified content slot further comprises:
   inserting the advertisement into the identified advertisement slot.

11. The method of claim 8, wherein a plurality of templates are received, and the method further comprises:
   receiving a selection of a desired template of the plurality of templates from a user.

12. The method of claim 8, wherein the content comprises at least one newspaper article.
13. The method of claim 9, wherein the advertisement is streamed.

14. The method of claim 9, wherein the advertisement is targeted to a characteristic of a user.

15. The method of claim 9, wherein the advertisement comprises a plurality of advertisements, and the advertisements change as the formatted content is rendered.

16. A computer readable medium comprising instructions to perform the steps of:
   receive content for an electronic publication;
   receive a plurality of templates, each template specifying a markup for the electronic publication and having a plurality of content slots;
   receive a desired template selection from a user; and
   render the content according to the selected template.

17. The computer readable medium of claim 16, wherein the step of render the content according to the selected template comprises:
   traversing the desired template to identify the content slots;
   retrieving content for insertion into the identified content slots; and
   formatting the content by inserting at least a portion of the content into the identified content slot;

18. The computer readable medium of claim 17, wherein the step of render the content according to the selected template comprises:
   traversing the desired template to identify an advertisement slot;
   receiving an advertisement for insertion into the identified advertisement slot; and
inserting the advertisement into the identified advertisement slot.

19. The computer readable medium of claim 16, wherein the content is a newspaper article.
Fig. 2
Fig. 3
Fig. 4

START

Receive template (410)

Receive content from publisher (415)

Receive advertisements (420)

Language processor applies template (425)

Render publication (430)

STOP
<xml version="1.0" encoding="utf-8" ?>
<Example>
<Select>
    SELECT PublicationID, Name, Title PublicationTitle, Directory FROM Publication
    WHERE Name = 'NYTimes'
</Select>
<ExecuteIfCreated>
    IF (SELECT COUNT (*) FROM Edition WHERE PublicationID = 
    @PublicationID AND PublicationDate = @Date AND 0 BEGIN
    INSERT INTO Edition (PublicationID, PublicationDate, BookID, 
    Title, Author, Category, [Key], [Size], URL) values (@PublicationID,
    @Date, @BookID, @Title, @Author, @Category,
    @Key, @Size, @YYYY@@MM@@DD@@BookID@@imp')
    END
    ELSE BEGIN
    UPDATE Edition SET BookID = @BookID, Title = @Title,
    Author = @Author, Category = @Category, [key] = @Key, [size] = @Size,
    url = @YYYY@@MM@@DD@@BookID@@imp WHERE PublicationID = @PublicationID AND PublicationDate = @Date
    END
</ExecuteIfCreated>

<BookID>@ContentFeedID@@YYYY@@MM@@DD@@</BookID>
<Title>@PublicationTitle@@Title>
<Author>@DayOfWeek@@MMM@@DD@@YYYY@@Author>
<Category>Newspapers@@Category>
<HTMLDirectory>@ContentBase@@Directory@@YYYY@@MM@@DD@@htm
<HTMLDirectory>
<IMPDirectory>@OutputBase@@Directory@@IMPDirectory>
<FileName>@YYYY@@MM@@DD@@FileName>
<TargetDevice>7<TargetDevice>
<Encrypt></Encrypt>
<Set From="" />
<Set To="" />
<Set TextFrom="" />
<Set TextTo="" />

TO FIG. 5a-2

FIG. 5a-1

SUBSTITUTE SHEET (RULE 26)
FROM FIG. 5a-1

```html
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<head>
  <meta http-equiv="content-type" content="text/html; charset=utf-8" />
  <title>@PublicationTitle@</title>
</head>

<!-- include CSS style file -->
<include src="NYTimesCSS.inc" />
```

```html
<! [CDATA[ ]] >

<! [CDATA [body id="top"] ] >

<!-- Begin - build query filter -->
<Set SelectionFilter="" />
<if _PageSelection="1">`

FIG. 5a-2
<If CategorySelection="" />
  <Set SelectionFilter="@SelectionFilter@ AND (Category = '[@_CategorySelection@]') />
  <!-- if category included as param then the category is passed as this -->
  </If>
  <!-- End - build query filter -->
</ForEachRow>
<!-- Begin for each category -->
<Select>
  SELECT ROW_NUMBER() OVER (ORDER BY PageOrder) AS theRowNumber,
  MAX(PageOrder) as PageOrder,
  ISNULL(MAX(PageType), CAST(PageOrder AS NVARCHAR)) AS PageStyle,
  MAX(Category) as Category,
  cbo.fn_FormatTime(dateadd('nh', -3, getdate()), 3) as theTime
FROM Article
WHERE (PublicationDate = '@_Date@') AND (PublicationID = @PublicationID@) @SelectionFilter@
GROUP BY PageOrder
ORDER BY PageOrder
</Select>

TO FIG. 5b-2

FIG. 5b-1
<Set SelectionFilter="@SelectionFilter@ AND (PageOrder = @PageSelection@)"/>

<!-- if category included as param then the category is passed as this -->

<!-- End - build query filter -->

<ForEachRow><!-- Begin for each category -->

<Select>
  SELECT ROW_NUMBER() OVER (ORDER BY PageOrder) AS 'theRowNumber',
  MAX(PageOrder) as PageOrder,
  ISNULL(MAX(PageType), CAST(PageOrder AS NVARCHAR)) AS PageStyle,
  MAX(Category) as Category,
  cbo.fn_FormatTime (dateadd (hh, -3, getdate ()) ) as 'theTime',
  left (DATENAME ( dw , '@_Date@'), 3) as 'theDOW'
FROM Article
WHERE (PublicationDate = '@_Date@') AND (PublicationID = @PublicationID@) @SelectionFilter@ GROUP BY PageOrder
ORDER BY PageOrder
</Select>

TO FIG. 5c-2

FIG. 5c-1
FROM FIG. 5c-1

<FrontPage>
   <!-- begin - NYT selection front page/homepage header -->
   <![CDATA[
   <div class="page-head" style="">
      <div style="line-height: 8.5mm; font-family: Sans Serif;">"><#160;></div>
      <div class="page-head-text">
         <div style="position: absolute; top: 0mm; left: 2mm; width: 42mm;">"<img alt="chronLogo"
            src="./ImagesLogo/nytlogo379x64.gif" style="width: 153px; height: 23px;" /></div>
      </div>
      <div style="position: absolute; top: 6.5mm; left: 2mm; width: 38mm; font-size: 5.5pt;">">@_DayOfWeek@. @_MMM@ @. @_YYYY@</div>
      <div style="position: absolute; top: 4.5mm; left: 75mm; width: 40mm; font-size: 10.5pt; text-align: middle; text-transform: capitalize;">">@Category@</div>
      <div style="position: absolute; top: 5.5mm; right: 3mm; width: 30mm; text-align: right; font-size: 6.5pt;">"<a href="@Category@" navlabel="@Category@">Tap for Section</a></div>
   </div>
   <!-- end - NYT section front page/homepage header -->
   <!-- include info settings file -->
   <Include Src="NYTimes.inc"/>
   ]]>\n</FrontPage>

FIG. 5c-2
<ForEachRow>

<!-- Begin - Top story image -->

<Select>
  SELECT TOP 1 articleID as ImageArticleID,
  FROM Image
  WHERE ArticleID IN
  (select articleID from @Directory@ TEMP
  AND Path IS NOT NULL
  order by width desc
</Select>

<If Path="" !>

<![CDATA[
  <div class="top_story">
    <div class="top_story_photo">
      <img alt="image Path" src="@Path@" style="width: 325px; height: 216px; scaletofit:"Y" />
      <a href="#article_@imageArticleID@">@Credit@</div>
    <div class="top_story_photo_credit">@Credit@</div>
    <div class="top_story_photo_caption" style="width: 325px; @Caption@</div>
  </div>
]]>

<!-- End - Top story image -->

TO FIG. 5d-2

FIG. 5d-1
FROM FIG. 5d-1

<! [CDATA [
  <div class="top_story_articleclip">
    <![CDATA[
      <!-- Begin - Top story title -->
      <ForEachRow>
        <Select>
          SELECT ROW_NUMBER () OVER (ORDER BY Priority) AS 'RowNumber', Path, Orientation, TextTitleText
          FROM Title
          WHERE ArticleID = @ImageArticleID@
        </Select>
        </ForEachRow>
        <!-- End - Top story title -->
        <div class="top_story_headline" translatefrom="@From@" translateto="@To@" translate=""/>
        <![CDATA[<div>]]></div>
      </ForEachRow>
      <!-- End - Top story byline and text -->
      <ForEachRow>
        <Select>
          SELECT ArticleNumber, ArticleID, Text, Summary, Byline
          FROM @Directory@.TEMP
          where ArticleID =
        </Select>
      </ForEachRow>
    ]]>]]>
  </div>
]>

FIG. 5d-2
FIG. 5e-1
FROM FIG. 5e-1

Summary, priority

FROM @Directory@_TEMP
ORDER BY ArticleNumber

</Select>

<![CDATA[
<div id="@priority@" class="sec_story">
  <a href="#article_@TEMPArticleID@" />
</div>
]]>

<!-- Begin - Section front page story title -->
<ForEachRow>
  <Select>
    SELECT ROW_NUMBER() OVER (ORDER BY Priority)
    FROM Title
    WHERE ArticleID = @TEMPArticleID@
    ORDER BY Priority
  </Select>
<![CDATA[<div
    class="sec_article_headline">]]>
  <Filter Data="TitleText"
    TranslateFrom="@From@" TranslateTo="@To@"/>
<![CDATA[<div>]]>

FIG. 5e-2
<ForEachRow>
<!-- End - Section front page top story title -->

<![CDATA[<div class="sec_article_byline">]]>
<Filter Data="Byline" TranslateFrom="@From@" TranslateTo="@To@" />
<![CDATA[<div>]]>

<!-- Begin - Section front page Top story image -->

<![CDATA[<div class="sec_article_photo_container">]]>
<ForEachRow>
  <Select>
    SELECT TOP 1 Path as ImagePath,
    Caption, Credit, width, height
    FROM Image
    WHERE ArticleID = @TEMPArticleID@
  </Select>
<![CDATA[<img alt="image. @imagePath@" src="@ImagePath@" class="sec_article_photo" style="width: 50px; height: 50px;" />]]>
</ForEachRow>
<!-- End - Section front page Top story image -->

<!-- Begin - Section front page Summary -->

<![CDATA[<span>]]>
<Filter Data="Summary" TranslateFrom="@From@" TranslateTo="@To@" />

TO FIG. 5f-2

FIG. 5f-1
<IF RowNumber="2">
  <![CDATA[<br />]]>
  <![CDATA[<span class="full_article_subhead" id="article_idArticleID@">]]> <Filter Data="TitleText" TranslateFrom="@From@" TranslateTo="@To@"/>
</IF>
</FOREachRow>
</FOREachRow>
</FOREachRow>

<![CDATA[<div class="full_article_columns body_indent">
  <![CDATA[<span class="full_article_text"-->]]>
</div>]]>

<FOREachRow>
  <![CDATA[Begin for each image -->]]>
  <SELECT>
    SELECT Top 1 Path, Caption, Photographer, Width as ThImageWidth, Height as ThImageHeight, Credit
    FROM Image
    WHERE ArticleID = @ArticleID@ AND Path !=
  </SELECT>

TO FIG. 5h-2

FIG. 5h-1
FROM FIG. 5h-1

ORDER BY Priority
</Select>

<![CDATA[
  <div class="full_article_photo">
    <![CDATA[
      <img alt="image.@Path@" src="@Path@" style="width: 310px; height: 206px;" scalable="Y" />
    ]]> 
    <div class="full_article_photo_credit">@Credit@</div>
  </div>
  <![CDATA[
    <div class="full_article_photo_caption" style="width: 310px;">
      @$Caption@$</div>
  ]]> 
</div>

</ForEachRow> <!-- End for each image -->

<![CDATA[
  <![CDATA[<div class="full_article_author">]]></CDATA>]]>
  <Filter Data="Byline" Start="0"
ReplaceLinefeed="" TranslateFrom="@TextFrom@" TranslateTo="@TextTo@" />
<![CDATA[</div>]]>
</If>

<![CDATA[<div class="full_article_author_credit">]]></CDATA>]]>
  <Filter Data="AuthorCredit" Start="0"
ReplaceLinefeed="" TranslateFrom="@TextFrom@" TranslateTo="@TextTo@" />
<![CDATA[</div>]]>

<![CDATA[<span class="full_article_text">]]></CDATA>]]>

FIG. 5h-2

SUBSTITUTE SHEET (RULE 26)
CONCLUSION

Another technical challenge addressed in the presented solution is how to efficiently filter and display a large dataset of articles within a limited space on a mobile device. The proposed solution utilizes a combination of XML for data representation and CSS for styling to achieve this goal. The filter mechanism is implemented using XML filters to dynamically select and display articles based on user preferences.

The integration of this solution with existing mobile applications will improve user experience by providing a more personalized and interactive browsing experience. This is particularly relevant in the context of news aggregation apps where users often seek articles that align with their specific interests.

FIG. 5i
<UserOptions PageStyles="1, 2"/>
<Set Orientation="horizontal"/>

<!-- Start with a clean slate each page -->
<Set NumberOfArticlesOnPage="100"/>
<Set FirstArticleInList="100"/>
<Set ArticlesInList="{'',''}"/>
<Unset Variables="ImageWidth, ""/>
<Unset Variables="ImageHeight, ""/>
<Unset Variables="ImageCutoff, ""/>
<Unset Variables="ArticleTitle, ""/>
<Unset Variables="ColStyle, ""/>
<Unset Variables="Style, ""/>

<If PageStyle="1"/>
    <UserOptions ArticleStyles1="1, 2, 3, 4, list"/>
</If>

<Set NumberOfArticlesOnPage="4"/>
<Set FirstArticleInList="5"/>

<Set ImageWidth1="150"/>
<Set ImageHeight1="150"/>

<Set ImageWidth2="100"/>
<Set ImageHeight2="100"/>

<Set ImageWidth3="100"/>
<Set ImageHeight3="100"/>

TO FIG. 6a-2

FIG. 6a-1
FROM FIG. 6a-1

<Set ImageWidth4="0" />
<Set ImageHeight4="0" />

<Set ImageWidth5="75" />
<Set ImageHeight5="75" />

<Set ImageWidth6="60" />
<Set ImageHeight6="60" />

<Set Ad="6" />
<Set AdWidth="45.5mm" />
<Set AdHeight="24mm" />

</If>
<If PageStyle="2">
<UserOptions ArticleStyles2="1, 2, 3, 4, 5, 6, list" />

<Set NumberOfArticlesOnPage="6" />
<Set FirstArticleInList="7" />

<Set ImageWidth1="100" />
<Set ImageHeight1="100" />
<Set ColStyle1="columns2" />

<Set ImageWidth2="100" />
<Set ImageHeight2="100" />

<Set ImageWidth3="100" />
<Set ImageHeight3="100" />

<Set ImageWidth4="0" />
<Set ImageHeight4="0" />

FIG. 6a-2
FIG. 6b
/* begin selectors for SECTION */

.sec_columns {column-count: 5; column-gap: 2.1mm; column-rule-style: solid; column-rule-width: 0.1mm; margin-top: 1mm;}

.sec_story {border-bottom: dotted black .1mm; padding-left: 0mm; padding-top: 1mm; padding-bottom: 0mm; padding-right: 0mm;}

.sec_article_headline {font-family: Serif; font-size: 8pt; font-weight: bold; margin-top: 0mm; margin-bottom: 0mm; margin-left: 0mm; margin-right: 3mm;}

.sec_article_byline {font-family: Sans Serif; font-size: 6pt; font-variant: small-caps; margin-left: 0mm; margin-top: 0mm; margin-bottom: 0mm;}

.sec_article_photo_container {font-family: Serif; font-size: 6.5pt; margin: 0mm;}

.sec_article_photo {margin-left: 0px; margin-right: 2px; margin-bottom: 0px; float: left;}

/* end selectors for SECTION */

/* begin selectors for FULLARTICLES */

.full_article_CatagoryTOC {position: relative; font-family: Serif; font-weight: bold; font-size: 10 pt; font-variant: small-caps; margin-left: 2mm; margin-top: 0mm; margin-bottom: 1mm; border-bottom: solid black 1mm;}

FIG. 7a-2
FIG. 7b
START

Receive template (810)

Receive content (815)

Receive advertisements (820)

Receive selection of template (825)

Apply template to content (830)

Render publication (835)

STOP

Fig. 8
Insurgent Attacks in Iraq

Iraq Attacks Lower, but Steady, New Figures Show

Controversial Canal

FIG. 9

SUBSTITUTE SHEET (RULE 26)
Iraq Attacks Lower, but Steady. New Figures Show

FIG. 10

SUBSTITUTE SHEET (RULE 26)
## A. CLASSIFICATION OF SUBJECT MATTER

<table>
<thead>
<tr>
<th>IPC(8)</th>
<th>USPC</th>
<th>Classification of Subject Matter</th>
</tr>
</thead>
<tbody>
<tr>
<td>G06F 17/00, G06F 17/20</td>
<td>715/235</td>
<td>According to International Patent Classification (IPC) or to both national classification and IPC</td>
</tr>
</tbody>
</table>

## B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
- USPC: 715/235

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
- USPC: 715/200,209,234,236,237,239,243,248
- 255 | 707/705,758,759,760,765,766, 999.1 | 358/1 | 1.1.13 (term limited; see below)

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
- PubWest(PGPB,USPT,EPAB,JPAB); Google Scholar; XML, template, ad, advertisement, ads, automatic, format, publication, newspaper, article, content, page, perform, render, display, present, create, generate, develop, assemble, online, Internet, web, electronic, content, container, computer, reading device

## C. DOCUMENTS CONSIDERED TO BE RELEVANT

<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>US 2003/0040926 A1 (MILTON) 27 February 2003 (27.02.2003) entire document, especially Abstracts; para [0004], [0006], [0008], [0019]-[0021], [0026], [0030]-[0033], [0036], [0037], [0040], [0043]</td>
<td>1 - 9, 11 - 19</td>
</tr>
</tbody>
</table>

Further documents are listed in the continuation of Box C.

**A** Special categories of cited documents:
- **A** document defining the general state of the art which is not considered to be of particular relevance
- **E** earlier application or patent but published on or after the international filing date
- **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)
- **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

**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
- **X** document of particular relevance: the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
- **Y** document of particular relevance: the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
- **&** document member of the same patent family

Date of the actual completion of the international search: 23 December 2010 (23.12.2010)

Name and mailing address of the ISA/US
- Mail Stop PCT, Attn: ISA/US, Commissioner for Patents
- P.O. Box 1450, Alexandria, Virginia 22313-1450
- Facsimile No. 571-273-3201

Date of mailing of the international search report: **11 JAN 2017**

Authorized officer: Lee W. Young
- PCT Helpdesk: 571-372-4300
- PCT O&S: 571-372-7774

Form PCT/ISA/210 (second sheet) (July 2009)