# Mass-Customizing Electronic Journals

*M. Carmen Fernández Panadero*
mcfp@it.uc3m.es www.it.uc3m.es/~per
*Vicente Luque Centeno*
vlc@it.uc3m.es www.it.uc3m.es/~per
*Carlos Delgado Kloos*
cdk@it.uc3m.es www.it.uc3m.es/~per
*Andrés Marín López*
amarin@it.uc3m.es www.it.uc3m.es/~per
*Carlos García Rubio*
cgr@it.uc3m.es www.it.uc3m.es/~per
*Luis Sánchez Fernández*
luis@it.uc3m.es www.it.uc3m.es/~per
Universidad Carlos III de Madrid Área Ingeniería Telemática, Dept. Tecnologías de las Comunicaciones Avda de la Universidad, 30 Leganés Madrid Spain E-28911
*Tony Hernández*
Universidad Carlos III de Madrid. Dept. Biblioteconomía y Documentación. C/ Madrid 126, Getafe, Madrid Spain E-28903 tony@bib.uc3m.es www.bib.uc3m.es/~tony/

## Abstract
The evolution of the WWW has opened the way to putting information at the fingertips of the whole world with very little effort. As the amount of information available grows, there is an ever increasing demand for personalized information. In this paper, we present some ideas that we are developing in the project "El Periotrónico", where we take a new approach to electronic newspapers. We are taking advantage of new Web technologies such as XML, XSL or DOM to personalize both a newspaper's content and interface layout according to users' preferences.
Keywords: XML; XSL; WWW; e-journalism; personalization;

## Introduction
The current Web chaos can be considered a consequence of HTML. HTML allows separating the content, presentation through CSS (Cascade Style Sheets), and behavior (Java and JavaScript files), but is not rich enough to describe the logical structure of the document and does not take full advantage of processing capabilities on the client side. New Web technologies like XML (Extensible Markup Language), XSL (Extensible Stylesheet Language), XLL (XML Linking Language - Xlink, a work in progress at the Web Consortium to provide greater linking capabilities to structured documents), DOM (Document Object Model, a platform- and language-neutral interface to access and update dynamically a document's content, structure and style), or Java, a general-purpose programming language that supports programming for the Internet in the form of platform-independent Java "applets" are appearing to solve some of these problems.

Since the World Wide Web Consortium approved the XML standard with the consensus of companies like IBM, SUN, Microsoft or Netscape) several scientific communities has developed this kind of metalanguage: MathML, for the description of mathematical documents, CML (Chemical Markup Language), for chemical documents, SMIL (Synchronized Multimedia Integration Language) for synchronization of multimedia content on the web or one of the most

used, the CDF (Channel Definition Format) for the channel definition which allow information delivery to the user under the push paradigm. The XML standard (eXtensible Markup Language) was designed to make easier to use SGML (Standard Generalized Markup Language) on the Web and it's called "extensible" because it is not a fixed format like HTML so it permit us to design our own markup language.

One of the main decisions in our electronic newspaper project design was to use XML to define our own markup language, Journalism Markup Language, (JML) to properly tag the journal content, its logical structure, and its metadata. This allows news articles to be self-describing which in turn allows more precise search criteria to be applied.
Likewise, we are defining a Journalism Preferences Markup Language (JPML) based on XML to specify the user's interest. The reader of a newspaper indicates his preferred topics. These are saved in a JPML document in order the system to show him/her only those pieces of JML news that match his/her preferences.

The introduction of metadata into the news media affects the way the news are created, selected and retrieved. Journalists are no longer constrained by the physical amount of space in the printed newspaper so they can make references to other news articles (yesterday's news or complementary sources) and have new ways to present information (multimedia content). For journalists this implies a need to indicate the importance level of every news element predicated on the reader's characteristics. New tags and attributes are needed for highlighting text in a personalized manner, defining target readers and indicating the expected level of importance assumed by the journalist.

Since journalists have to include JML tags in their articles, a JML editor should be provided for them. New IBM tools that deal with XML include a program that generates an XML editor adapted to a user-defined DTD (Document Type Definition). The automatically generated JML editor can be extended with Java Database Connectivity (JDBC) routines that insert the JML documents into a SQL (an industry-standard language for creating, updating and querying relational database management systems) database. The editor can also manage images for illustration and advertising.

The next sections describe the benefits of using a XML technology like XSL in the journal generation process, some journal personalization details used in our project and a description of JML and JPML as proposed XML applications for news markup and personalization markup. Finally, some details about the mixed evolution of Web Technology and Digital TV and some conclusions and future work are presented.

## Architecture and system components

The system is composed by five main components: journalist's editor, the database, the publisher daemon, the digital news archive, and the digital bookbinder which includes a personalization agent.

### Journal generation - Journalist's editor

We see an electronic journal as an information system composed by multiple pieces of boxes containing news in form of text, still image, video, audio or both of them. These boxes maintain different relations at different levels between them. For example, all boxes may be published by the same media, or may be published the same day or may share de same topic. Our unit of analysis will be the content of this boxes.

A great part of the news published in a newspaper or via online news delivery system, each box, has a macrostructure easy to identify and therefore easy to automate, such as the publication date of the article, the author, the section/s where is allocated or the extension (in bytes) a clue for the depth of the news article. However the content of every box has a microstructure not so easy to automate: for example, to establish the relationships between boxes that share the same topic in the same or different day of publication or to distinguish the importance level between boxes, something relatively easy on the paper form but not in electronic form.

We assume that people, overwhelmed by information streams, will prefer information companies able to narrowcast according to user preferences. This narrowcasting services will send users mainly the information they marked as susceptible of interest but also a stream of news selected by the editor of the service. So, journalists will maintain old roles: to select (information gatekeeper) and harvest the news (researching, summarizing and writing) and offer it in a proper manner, but also new tasks like offering to their readers the sources where they got the data (hyperlinks, sources where to get a deep knowledge of the news read) and marks (JML) that help users (through software agents) to identify a real topic from a simple match of words. The journalist's editor we develop is designed to be a Java applet / application whose main purpose is to edit the news articles that comprise the journal by using JML tags. This tool validates the document conforming to the DTD grammar and inserts the news into the database categorizing them by the metadata described in their headers and in a near future also in the body of the news articles. The journalist's editor can also manage images for illustration and advertising. This application will function with the editorial system of newspaper companies.

### The database

The database is the main repository of published documents. Media files like text, audio, video or pictures that comprise the news document are stored in the database. Only standard SQL queries for relational databases are composed by our system, so that this component can be easily replaced by a large amount of products from different vendors. Currently, we are using a freeware database called mysql in a networked Linux platform.

### Publisher Daemon

The publisher daemon consists of an application that periodically dumps the most recent content of the database to files ready to be delivered by the Web server. Though our final prototype will send JML news to the client's browser just the way they were written by the journalists, our current prototype implementation dumps that content in HTML format, providing every news metadata stored in JavaScript variables. These JavaScript variables are used in the process of a simple personalization executed in the client by the personalization agent.

Digital News Archive

News articles no longer published are removed from the server. However, the database stores every news document published by the journal. Since news articles can be extracted by combining some matching criteria, it is possible to provide researchers with a tool for extracting interconnected information by just allowing the readers to directly access the database. The larger content stored in the database, the more valuable this service will be considered by the user. The digital news archive is to be implemented as a Common Gateway Interface (CGI) program. A CGI is just a technique which allows users to run programs on a WWW server. However, the execution of this CGI program (implemented in Java by using the JDBC package) implies an allocation of resources that may considerably reduce the number of concurrent accesses to the server.

### Digital Bookbinder - Personalization agent

The digital bookbinder is the agent that builds the personalized newspaper on-demand. Since

the journal's edition is divided in different sections, the digital bookbinder is required to compose the personalized newspaper by including only the sections that the reader has previously subscribed to or has explicitly required and a minimum of the editor's choice. Unlike printed journals, which include a compendium of all of their sections gathered consecutive in a single edition which weighs some hundreds of grams, the bookbinder provides the reader with a collection of sections which are known to be of the interest of the reader filtering out sections the reader never reads.

Sections included by the bookbinder are merely an index of classified links and headlines that can be expanded and collapsed by using cross browser dynamic HTML routines. Each reference is self-described in JavaScript variables so that the personalization agent executed in the client (written in JavaScript) can be applied to highlight headlines that are detected as important for the reader. Currently, the bookbinder is being implemented as a CGI program that merely includes the files needed in the client. As new HTML 4.0 compliant browsers arrive, the dynamic inclusion of sections stored in different files will be performed on the client side and the digital bookbinder will be integrated into the personalization agent executed in the client's browser. Dynamic selective inclusion of documents can be easily implemented in a JavaScript routine by using the HTML OBJECT tag, which allows embedding different HTML documents in the same main document. However, since most browsers don't support this tag yet, the bookbinder agent has to be executed in the server side at present.

The bookbinder also allows a range of dates to be specified, so that sections published within that period are included in the personalized edition. This way, a reader can easily ask for the newspaper of every weekday except weekend, the newspaper of the last N days, or just for the time during which he/she has been on holidays.

### Personalization Agent

Personalization applies to contents but also to style and layout. We have implemented a personalization agent written in JavaScript that performs contents customization of the news at the client side. Readers can subscribe to different sections. Although the reader can specify his preferences statically in a form, Web technology allows dynamic personalization too. The object-oriented model of the XML documents and the DOM standard are a perfect material in which to structure information for further processing by languages like Java or JavaScript. The usage of these languages allows the document to interact with the reader. The system can automatically detect the behavior of the user and analyze it to dynamically modify the configuration parameters.

Our personalization agent performs contents customization for every headline at client side. Headlines are or are not highlighted according to the agent's criteria as configured by the reader's preferences. Reader's preferences can be classified in two different groups:

> Static preferences: this include the preferences explicitly specified by the reader in a form. Readers are allowed to enumerate a list of keywords, authors, information sources, places and sections that are of interest as well as some restrictions of these conditions. Static preferences are allowed to specify interests like headlines that refer to the euro but are not in the Finances section or headlines that refer to my local soccer team and do deal with European competitions or my favorite journalist articles that are referred to Ecology. Though the personalization agent is not fully implemented and currently only highlights headlines that include any element from the user's list of keywords as a substring, our future versions will deal with more complex conditions like the ones enumerated above. For that reason, we have also defined an XML application called

JPML (Journalism Preferences Markup Language) which will be used to describe the user's static preferences and will configure the behavior of the personalization agent. Examples of the previous complex preferences can be found in section JPML.
Dynamic preferences: this include the preferences that the personalization agent automatically detects by analyzing the behavior of the user. Since the list of user's preferred keywords evolves as the user's interests, a mechanism to dynamically modify the user's preferences is needed. This way, temporal or even unconscious preferences can be considered in the highlighting links process, releasing the user to regularly reprogram his static preference list. Aging algorithms can be applied so that temporal interests can be automatically removed from the list of dynamic preferences. Besides our current personalization agent performs a quite simple personalization based exclusively in static preferences, it is not considered to be difficult to include the treatment of dynamic preferences of the user as soon as every news article has a self described list of keywords. A simple JavaScript routine can add those keywords to the dynamic list. This routine is called every time the user activates events that manifest his/her interest on a news article

The personalization agent take both static and dynamic preferences into account on the process of selecting news and recommending about interesting articles. But the agent may also take in account other readers' preferred articles with the same interests of the user. This opens the door to virtual communities, or individuals with common interests.

As well as content's personalization user may to personalize the style and layout of the newspaper. Not only colors, fonts sizes or font families can be specified in the personalization process. Style personalization also applies to the positioning of elements across the window and the way headlines are highlighted. Personalized style sheets provide this functionality. The user can manage his own cascading style sheet file by simply filling in a form. Currently, user's style sheets are easily maintained by a CGI program that stores them in the server side. It is one of our short term aims to allow the user to store his own style sheet locally in his computer, so that the server doesn't need to store users' files. (It is not our aim to maintain thousands of user's files in the server side).

While XML defines the logical structure of the news, XSL allows specifying the formats for different news. One of the main advantages of using XSL instead of CSS is the possibility of specifying a transformation step before the formatting in order to achieve not only a different format but also a different physical structure. With the same XML document but different style sheets we can generate different versions of the newspaper with structure and formatting properties that match different information spaces (printed version, online-version in broadband networks, online-version on networks with smaller bandwidth, etc). In particular, we can convert XML documents to HTML, SMIL (SMIL (Synchronized Multimedia Integration Language), and maybe in the future to MHEG (Multimedia and Hypermedia information coding Expert Group), an ISO standard encoding for multimedia and hypermedia information, designed to facilitate use and interchange of information in varied domains such as games, electronic publishing and medical applications, for display via a set-top box on a TV set. After the transformation step, style sheet rules specify the format of the document.

The result for the end-user is a stream of data received periodically in the manner, and in a near future via the medium, chosen by the user containing different sections (expanded or collapsed) with links and/or full text articles (highlighted or not) to be reviewed by the user with the style marked in the preferences through the JPML (see below).

## Journalism Markup Language (JML)

Some associations and consortiums like Newspaper Association of America (NAA) with a recent standard for online classified ads or the News Industry Text Format (NITF) or like the ICE Authoring Group which includes Adobe Systems Inc., Sun Microsystems, Microsoft, Vignette, CNET, Tribune Media Services, CNN, Pointcast, Reuters NewMedia or ZDNet are developing standards based on XML. The ICE standard (Information and Content Exchange) was created "to significantly reduce the cost of online business by providing the standard to build Internet value or trading networks, such as syndicated-publishing networks, Web super-stores and online-reseller channels".

The purpose of JML, as markup language based on XML, is to properly tag the journal's contents and its metadata so that three different aims can be achieved.

- News articles may be "self described'' in order to be properly handled in the personalization process.
- The news archive can be accessed by combining matching criteria in order to produce refined results, not every news article that just contains the searching term somewhere in its text.
- Different style rules can be applied to the same document, so that the same document can be viewed with a different layout in a personalized manner.

In JML every box will have head and body elements. Head elements include among others: title, author, place, date, section, journal, keywords. Body elements include about 25 HTML tags reducing the number of attributes and changing some requirements (for example, <img> is at block level instead of inline and if it's in the box it has to be complemented with an image title. Another elements are: names (of people, organizations and products) and attributes like role or alias; locations, numbers or cites among others. Ideally much of the tags could be inserted semi-automatically because the editor is able to mark the elements existing in a kind of dictionary files.

The text below shows a reduced version of the JML's DTD grammar and below that we show a small example of a news article tagged in JML.

```
<!ELEMENT JML (JML_AUTHOR, JML_PLACE?, JML_DATE?,
                JML_TITLE, JML_ABSTRACT?, JML_BODY)>
<!ELEMENT JML_AUTHOR EMPTY>
<!ATTLIST JML_AUTHOR value CDATA #IMPLIED>

<!ELEMENT JML_PLACE EMPTY>
<!ATTLIST JML_PLACE value CDATA #IMPLIED>

<!ELEMENT JML_DATE EMPTY>
<!ATTLIST JML_DATE value CDATA #IMPLIED>

<!ELEMENT JML_TITLE (#PCDATA)>

<!ELEMENT JML_ABSTRACT (#PCDATA)>

<!ELEMENT JML_BODY (#PCDATA|P)*>
```

```
<!ELEMENT P (#PCDATA|B|I)*>
<!ATTLIST P importance_level CDATA #IMPLIED>

<!ELEMENT B (#PCDATA)*>
<!ELEMENT I (#PCDATA)*>
```
JML DTD grammar

```
<?xml version=”1.0"?>
<!DOCTYPE JML SYSTEM “jml.dtd”>
<JML>
  <JML_AUTHOR value=”Maruja Torres”/>
  <JML_PLACE value=”Madrid”/>
  <JML_DATE value=”09-06-1998"/>
  <JML_TITLE>This is the title</JML_TITLE>
  <JML_ABSTRACT>This is the abstract</JML_ABSTRACT>
  <JML_BODY>
      <P importance_level=”general”>
      This <B>is</B> the body</P>
  </JML_BODY>
</JML>
```
Example of JML document

### Journalism Personalization Markup Language (JPML)

JPML has been defined to specify user’s interests. Preferences determine the way headlines are shown (highlighted, collapsed, inline, linked, ...). However, the reader can also perform explicit requests that don’t match the preferences. Text below shows a simple example of a reader’s preferences and figure below that specifies the DTD grammar for this markup language.

```
<?xml version=”1.0"?>
<!DOCTYPE JPML SYSTEM “jpml.dtd”>
<JPML>
<RULE>
      <ATOM key=”keyword” value=”euro”/>
      <ATOM key=”section” value=”finances” negated=”true”/>
</RULE>
<RULE>
      <ATOM key=”keyword” value=”Real Madrid”/>
      <ATOM key=”keyword” value=”Champions League”/>
</RULE>
<RULE>
      <ATOM key=”author” value=”Clark Kent”/>
      <ATOM key=”keyword” value=”Ecology”/>
</RULE>
</JPML>
```
JPML example

```
<!ELEMENT JPML (RULE)*>

<!ENTITY % match “(starts_with|ends_with|substring|fullword|is_equal_to)” >
```

```
<!ELEMENT RULE (ATOM)*>
<!ATTLIST RULE
        enabled (true|false) “true”
        description CDATA #IMPLIED
        action CDATA #IMPLIED
>

<!ELEMENT ATOM EMPTY>
<!ATTLIST ATOM
        key CDATA #REQUIRED
        value CDATA #REQUIRED
        ignorecase (true|false) “true”
        ignoreaccents (true|false) “true”
        negated (true|false) “false”
        matching %match; “substring”
>
```
JPML DTD

The meaning of condition attributes is described below:

**key:** defines the name of the metadata field to which the matching criteria is applied. Possible values for this attribute are title, section, author, keywords, date, source, ....

**value:** defines some value specified by the user that can be compared against the value of the key metadata.

**ignorecase:** defines whether values have to be folded to uppercase before compared, overriding the need of exact match.

**ignoreaccents:** defines whether orthographic accents should be considered.

**negated:** reverses the condition.

**matching:** defines the criteria which is to be applied between the value and the key’s value. Though the less restrictive “substring’’ criteria is applied as the default, other criteria can be specified, like ``fullword’’ for matching whole words, ``starts_with’’ or ``ends_with’’, which require that the specified value can be found at the beginning or the end of the metadata field. This allows people to search and/or highlight headlines whose author is Clark Kent, whose title starts with Clinton or whose keywords contain Iraq.
Besides that, rules also define the following attributes:

**enabled:** for enabling/disabling the rule.

**description:** a short user’s description for that rule.

**action:** the action to be performed when the rule is activated. Possible values for this attribute are highlight, iconify, hide, open in full window.

## Future work: integration of Web Technology in Digital Television

WWW evolves too fast. Though JML, our XML language for journalism, is currently only present at our server side, it seems to be clear that XML browsers for Internet will appear in a

few months. Then will be the moment to use a style sheet so that JML can be directly visualized in the client's browser instead of being transformed into HTML at the server side. Backward compatibility will be achieved by maintaning a HTML version for older browsers, but, for these readers, personalization services will loose the benefits of XML.

The recent DOM standardization is also a very important milestone that will allow software agents implemented in JavaScript to run without platform details in any browser, with much more flexibility and portability than current Dynamic HTML. Our efforts are now moving to convert the prototype in a real application but taking in account the tendencies that drive to an integration of Web and digital TV.

There is currently a big activity around the integration of Web based technology in digital television. This integration offers advantages both to Internet content providers and digital television companies. Digital television offers the possibility of integrating audio video and data in real time and processing capabilities at the customer location by means of the set-top-boxes. This opens the possibility to offer to the customers new services, including interactive television. These new services could be based on Web technology. The Internet content providers can access to a big amount of potential customers. Many of these potential customers are not using Internet and therefore, cannot be addressed by this means.

Examples of the applications that could be offered include access to Internet via digital television, or using Web technology for annotating broadcast television. In the first case, part of the bandwidth provided is shared between all the customers to access the Internet. In the second case, Web technology gives the enhanced television contents a certain degree of interactivity. Among the different initiatives that are currently being carried out with respect to "television Web" we could cite the following. MHEG is a family ISO standards that deal with the coding of hypermedia contents. It includes the definition of multimedia objects, a declarative language for presenting multimedia contents, and a scripting language for data processing in MHEG applications.

ATVEF is an industry group that includes many companies interested in interactive television. Among these companies we can cite CNN, Disney, Intel, Microsoft, etc. Inside ATVEF it is being developed a specification that supports the presentation of so-called "HTML-enhanced" television contents. It is composed of announcements of the programming, triggers that define the actions to take and the location of the contents and the multimedia contents.

Finally, the World Wide Web Consortium has created a "Television and the Web" Interest Group. Inside this interest group, several activities around the integration of the Web and digital television are performed.

We are witnessing an explosion of multimedia and interactive services. This is causing a whole plethora of standards to come up, or existing standards to try to adapt to the new media. HTML belongs to the former case, and JPEG and MPEG are examples of the latter. Given this scenario, the need to standardize a higher level interface than the current standards naturally arises. MHEG, which is an acronym for Multimedia and Hypermedia coding information Experts Group, is a set of standards under development by ISO that address the specification of platform independent applications consisting of multimedia objects. Specifically, it focuses on:
Synchronization in space and time of these multimedia objects

User interaction via links and user interface elements such as menus, buttons and text entry fields

An MHEG application consists mainly of declarative code that describes the objects that make it up. The application code is stored in servers that handle it to requesting clients. Nor the models nor the applications that are likely to make use of MHEG objects are defined by the standards. Possible scenarios include periodic broadcasting of Near Video on Demand or demand downloading of an electronic education application. The encoding of multimedia content is not part of the standards either; it is assumed that existing standards such as MPEG or AVI will be used.
On the client side, an MHEG engine parses the declarative code, produces the required on-screen presentation and handles all user interaction. This engine should be supported on machines with minimal resources, such as set top boxes. This is the reason why cpu-hungry tasks such as 3D imaging have been left out of the initial standard. The low resources constraint implies that MHEG is not restricted to Web browsers, but instead intended to serve as a basic form of encoding multimedia/hypermedia presentations to be transferred between pairs of heterogeneous machines, one acting as the server and the other(s) acting as the client(s).

MHEG shares with HTML the declarative approach, but while HTML is inherently a document description language, MHEG takes on the job of describing multimedia/hypermedia applications. Similar standardization efforts been done by other groups. This is the case of SMIL, which is an application of XML targeted at the synchronization of video and audio. MHEG should eventually emerge as the leading technology in the field of multimedia presentations.
MHEG presence is imminent in the field of interactive TV, as it has been adopted by DAVIC. DAVIC is an international industry consortium whose purpose is to establish a common field of standards and protocols for the emerging digital interactive television.

The MHEG standard specifies the following notations to represent application components:
ASN.1 - this notation was the first one to be developed. Although application components are unambiguously expressed in ASN.1, it is not considered friendly enough to be read by humans so the following alternate notation was developed.

Textual Notation - this notation was developed to overcome the problems with ASN.1. It doesn't add new features, there is a one to one mapping between both ASN.1 and textual notations.
XML - currently under development, this notation is targeted to the Web world. It is thought to attract a wider user community due to the growing acceptance of XML. An earlier effort to define an SGML based encoding was cancelled due to lack of resources.

## Acknowledgements

The work reported in this paper has been partially funded by the project **TEL97-0788** of the Spanish **CICYT**. We wish to acknowledge fruitful discussions with our colleagues Peter T. Breuer, Pilar Diezhandino, Natividad Martínez, Tomás Nogales, A. Rodríguez de las Heras and David Rodríguez of the **Universidad Carlos III de Madrid**. Useful assistance has been provided by **El País Digital** and **Fundesco**.

## References

Advanced Television Enhancement Forum (ATVEF). http://www.atvef.com

Bowser, Andrew. Into the Digital Future, TechNews 4 (2): March/April 1998. http://www.naa.org/technews/tn980304/p8digfut.htm

El Digital de Telepolis http://www.telepolis.es

Feola, Chris. Cool as ICE. TechNews 5 (1): January/February 1999.

ISO/IEC 13522-5: Information technology - Coding of multimedia and hypermedia information - Part 5: Support for base-level interactive applications

Krishna Bharat, Tomonari Kamba, Michael Albers, Personalized, interactive news on the Web, Multimedia Systems 6: 349-358 (1998)

MHEG Centre. http://www.mhegcentre.com

MHEG Complements Interactive TV. Valerie Thompson. http://www.byte.com/art/9702/sec17/art6.htm

Singer, Jane B. Journal of Computer Mediated Communication 4 (1) September 1998. http://www.ascusc.org/jcmc/vol4/issue1/singer.html

Tim Bray, Jean Paoli, and C. M. Sperberg-McQueen (eds): XML: Extensible Markup Language (XML) 1.0, W3C Recommendation, 10-Feb-1998, http://www.w3.org/TR/REC-xml

Titulares.com http://www.titulares.com

Van Dijk, Teun A. “La Noticia Como Discurso.” Barcelona: Paidós, 1990.

Watters, C.R., Shepherd, M.A., & Burkowski, F.J. Electronic News Delivery Project. Journal of the American Society for Information Science 49 (2): 134-150, 1998

World Wide Web Consortium. Television and the Web. http://www.w3.org/TV/