Open architecture for multilingual web sites

M.T. Carrasco Benitez
Luxembourg, March 2012, draft version 0.19

1. Abstract

Multilingual Web Sites (MWS) refers to web sites that contain *multilingual parallel texts*; i.e., texts that are translations of each other. For example, most of the European Institutions sites are MWS, such as Europa [EU]. This document sketches a comprehensive **open architecture**. In particular, it takes into account:

- **Users** should expect the same multilingual behaviour when using different browsers and/or visiting different web sites.
- **Webmasters** should be capable of creating quickly high quality, low cost MWS.

2. Status

This document is a draft. The author **solicits** comments from the community at large. Some parts of this document are bullet lists that have to be written in prose. This is a general architecture document: a significant amount of work will be needed to write precise specifications. Also, this document might be considered a primer on MWS.

Multilingual web aspects must be positioned in the wider web context; e.g., language is just one of the dimensions in *transparent content negotiation* [TCN]. One should find particular solutions for the specific multilingual aspects that are not standardised, though taking into account the wider context; e.g., the interoperability among data in the web servers and *content management systems* (CMS) is a general issue and not particular to MWS.

There should be a general rolling architecture, but there should be also concrete immediate steps. The **next step** should be the one with the best *return on investment* (ROI); e.g., specifying and implementing the server side language as an URI: `http://example.com/var`.

2.1. Relevance

MWS are of great practical relevance as there are very important portals with many hits; also they are very complex and costly to create and maintain: translation is expensive. Facilitating and enjoying this common experience entails **standardisation**: current multilingual web sites are applications incompatible with each other. It is vital to agree on common behaviours for users: browser-side (*language button*) and server-side (*language page*).

2.2. Break nothing

Stating the obvious, adding these facilities1must not break any of the existing standards. Indeed, sometimes it just means implementing mechanisms already in the standards such as TCN; e.g., in the case of the language button it means that servers have to return the available languages in the HTTP header, and browsers must process this data and activate the language button, if necessary.

3. Interfaces

The users and webmasters issues map into two interfaces:

- **Interface browsers-servers (users)**: well established; indeed, the heart of the web with HTTP, URI and HTML. Very delicate as it touches the **grand public**; even a small additional complexity could be disastrous. So, the multilingual aspects have to enter the dance without causing any missteps.
- **Interface servers-content management (webmasters)**: lots of work to do. Indeed, it is more general than multilingual. One could have more complexity as it touches mostly professionals.

4. Interface browsers-servers (users)

4.1. Monolingual

From a users point of view, the most common usage is monolingual, thought a site might be multilingual; i.e., users might be interested in just one language of the several available at the server. The language selection is just a barrier to get the appropriate linguistic version. Some users might be really interested in several linguistic versions.

4.2. Transparent content negotiation (TCN)

One can get the *best language* with TCN. For this, browsers need to be properly set. TCN is not widely known and it might be confusing for naïve users; also, one might be using a browser that one cannot set, such in the case of kiosks. On the other hand, one should not unduly penalise users that want to use TCN. Hence a compromise should be found that combines TCN and other mechanisms such as direct language select perhaps using cookies. Proxy servers could create problems for TCN.

4.3. Language selection

One needs a direct language selecting mechanism if:

- TCN is not available.
- TCN is available, but the requested languages are unavailable.
- One wants to switch to another language.

4.4. Options

These are the options:

- Initial arrival
    - TCN
    - Select one language explicitly from a list of languages
        - Monolingual path
            - One server with all the languages
                - File extension
                - Language branch
            - A federation of servers, each with one language (Wikipedia)
        - Cookie
            - Alone or combined with TCN

- Switch among languages
    - TCN
    - Cookie
    - TCN + cookie
    - Links in pages
        - One per language
        - Menu

        - Select and enter
        - Select and automatically go with JavaScript
    - Change the language code in the URI

4.5. Browser and server side

- **Browser side**: The controls are in the browser.
- **Server side**: A page sent by the server.

4.6. Variant

4.6.1 Overview

This section addresses mainly the MWS aspects. It includes:

- Variants (languages, format, etc); immediate and deferred
- Metadata of the variant; e.g., author of the variant
- Metadata of the URI; e.g., editor for the whole URI set
- Metadata of the site; e.g., copyright for the whole site
- Client setting; e.g., language preferences
- HTTP response header
- Previous versions of the page
- Services; immediate, nearly immediate (e.g., PDF in Wikipedia) and deferred

4.6.2 Mechanisms for getting the variant

It is related to how one want to use the variant:

- **Browser side - headers**: Response header and document header. It is for populating browser facilities such as the *language button*. Only items in the HTTP header and appropriate formats (e.g., HTML) can be obtained; though TCN *features* are quite extensible
- **Server side - URI**: in the response body, in different formats. Usually an HTML page. Items not allowed by the HTTP protocol might be included.

Some items might be not immediately available. For example, the German language variant might be available in about one hour.

4.6.3 Variant URI

It returns the variants for a particular URI. It has three dimensions:

- **Items**: item set; e.g., only the languages.
- **Format**: the format of the returned data.
- **Parameter**: how to pass the parameter to the server

The *variant selector* is the part of the variant URI used to indicate the combination of the above dimensions. For example, `http://example.com/var` returns all allowed items by the server as a microformat XHTML; the string `var` is the *variant selector*.

Examples:

- **Microformat XHTML**: Typically included pages as a link. In addition to human-machine (switching among languages), it can also be used like a web service for machine-machine. The selector is `var/xhtml`; abbreviated to `var`.
- **Languages**: Only the languages in microformat XHTML and a few other essential items. The selector is `var/xhtml/lang`; abbreviated to `var-lang`.
- **XML**: The selector is `var/xml`.
- **Plain text**: The selector is `var/text`.

- **PDF**: The selector is `var/pdf`.

The parameter could be:
- **Local URI**: `foo`
- **Full URI**: http://example.org/foo

The parameter can be sent to the server in the following ways:
- **Referer**: the parameter is in the `Referer` header field; e.g., `http://example.com/var`. The **same** URI is used for obtaining the var of all URIs as the browser graciously provides the parameter. This mechanism **greatly simplifies** building MWS as having a different URI for each page is much harder.
- **GET method**: the parameter is the query part; e.g., `http://example.com/var?foo`
- **POST method**: the parameter is in the body; e.g., `http://example.com/var` and the body contains the string `foo`.
- **Dedicated server**: the parameter is in the path; e.g., `http://meta.example.com/foo`

Typically, the variant URI would be included in all the pages of a server. As a scenario, a user would click the var URI to obtain the variants; and select the appropriate navigation facility, such as another language.

If the server does not know any of the languages requested and there is no default, it should present the appropriate language neutral variants to choose one language. It should be easy to have the variants in many languages.

Reservation of magic strings: the path and third level domain starting with “`var"`. For example, `http://example.com/var` or `http://var.example.com`.

Next step: implement `http://example.com/var` returning languages in XHTML.

4.7. Browser side: variant button

Browsers should contain a *variant button* (in the row with File, Edit, etc) for using the variants accessible through this mechanism. The variant button could become just a *language button* (i.e., dedicated just to the language) by changing the browser preference panel; e.g., the button could be labeled as *variant* or *language*.

The variant button should become enabled when relevant; e.g., when other languages are available. A more economical approach from the server point of view would be to send the appropriate variant content only on request (when the user presses the variant button) or just send the *variant*. Again, these options could be chosen through the preference panel.

There would be a long transition period and a server side mechanism would be needed in the meantime.

The browser might inform the server that it has a variant or language button, perhaps in the `User-Agent` or `Expect` headers. The server might act accordingly.

4.8. Server side: variant page (XHTML)

The variants would look like this small example:

```
URI                        : http://example.com/foo
This page available in     : English, Spanish, French
Copyright                  : Somebody
Site copyright             : http://example.com/copyright
Your preferred languages   : English, Spanish {add the cookie one}
```

Look at the XHTML example at [MR].

The page should be as a XHTML microformat. In particular, the languages should be links to the relevant variants for easy switching among language variants. Other dimensions could also be included; e.g., formats.

The language of the variant should be per preferred language (TCN or cookie). Also, there should be a mechanism to change the language of the variant.

5. Server-content management (webmasters)

5.1. Webmaster

Webmaster refers to all the aspect of the construction of MWS: author, translator, etc. The objective is the creation of high quality low cost MWS. Many existing applications have some multilingual facilities and (stating the obvious) one should harvest the best techniques around.

Servers should expect the same API. The first API could be just a multilingual data structure. The absence of this data structure means that each application has to craft this facility; having the same data structure means that servers (or other programs) would know how to process this data structure directly. It is a case of production of multilingual parallel texts: the cycle *Authorship, Translation and Publication chain* (ATP-chain).

The API between the CMS and the server is a general issue and not specific to MWS.

5.2. Multilingual data structure

- All in one server
    - File extensions; e.g., `foo.en.html`, `foo.es.html`
    - One directory per language; e.g., `en/foo.html`, `es/foo.html`
- Federation of servers, each server with one language; e.g., Wikipedia

In the case of federated servers, one server could be dedicated to variants. Hence, a system is required to collect the data from the different servers, either on the fly or before hand.

{TODO} Evolution to this approach: old links.

5.3. Generating language in parallel

For the purpose of this section, the files in a server are classified as per these two scenarios:

- **Navigation**: HTML mainly used for navigation with a few paragraphs.
- **Big**: PDF with a few dozen pages.

These are two extremes. The point is that there is no intention of addressing the general problem of generating complex thousand of pages document on high quality typography.

The general approach should be to **generate all the linguistic versions in parallel**.

So, an approach could be based on the following two components:

- Skeleton
- Language table: a list of key/values, where the values are language segments (typically a sentence)

The intention is to replace each key in the skeleton by its value. In the case of HTML and XML, the keys could be entities.

Example:

- Input
    - Language table
    - Skeleton
- Output pages
    - 100.en.html
    - 100.es.html

Input language table

Keys	Values	
Lang	en	es
Hello	Hello world	Hola mundo

Input skeleton

```
<html lang="&lang;">
  <body>
   <p>&hello;</p>
 </body>
</html>
```

Output file 100.en.html

```
<html lang="en">
  <body>
   <p>Hello word</p>
 </body>
</html>
```

Output file 100.es.html

```
<html lang="es">
  <body>
   <p>Hola mundo</p>
 </body>
</html>
```

5.4. Skeleton

This construction is format (MIME type) dependent; e.g., it can be done in HTML and XML, but it might not be done in other formats.

5.5. Language table

The language table is an abstract construction: a list of key/value pairs, where the key is a unique identifier and the value a language segment. The language table can be implemented in at least the following ways:

- Text files: one file per language; e.g., the line `k1` in `mytext.es.txt`
- URI: e.g., http://example.com/es/k1
- Database: e.g., SQLite

5.5.1 Key

The language key is a unique identifier in each processing of the skeleton and language table, though it would be better to be unique at least across a system that could include a set of sites.

The language keys in a skeleton could be abbreviated; e.g., `http://es.example.com/k1` could be abbreviated to just `k1`.

The HTML generator program must know how to compose the full key; e.g., a parameter or a meta declaration in the skeleton.

5.5.2 Value

A language value is whatever a language key points to; typically a sentence. But it could be in any format; e.g., a sound file.

In this context, a sentence does not have any grammatical connotation: one can think of it as a string.

5.6. Author, Translation, Publishing chain (ATP-chain) in MWS

- The author produces the skeleton and the source language values
- The translator produces the other language values
- A program generates the HTML pages in all the required languages

5.7. Generating techniques

It could be:

- Internal to the server
- Dynamically; i.e., when the pages are requested
- Generate the first time requested and keep until stale, for example because one of the entities has changed
- Separated program
- Batch; i.e., all in one go

5.8. Multilingual Web Content Management System

- Describe a scenario
- Assisted authorship
- Compulsory summary style
- Revision system
- Modify source text following the output of revision system
- At the very least, automatic summary
- Interface to Computer-Aided Translation (CAT).
- Minimise translations
- Translation on demand
- Translation after voting
- Vote weighting
- Translation could be human, machine, etc
- Generalise for translation request management
- Type of pages
    - Navigational: could include a short text
    - Documental: significant text
- Automatic typography
    - SVG

    - Formatting objects

6. Language neutral URIs

- No extensions
- Short
- Flat, no hierarchy
- Base 36
- Separation URI from storage

7. Background

- Minimal primer
- Gory details in primary sources
- Wider context OAMPT

7.1. Transparent Content Negotiation (TCN)

One URI can have several variants. For example, `http://example.com/foo` could have:

- English in HTML
- English in PDF
- Spanish in HTML
- Spanish in PDF

Extensions address one particular variant

Informally:

- Variant list: list of available variants
- Language variant list: list of available linguistic versions; a subset of a variant list

Some of the HTTP header fields involved are:

- Accept-Language
- Content-Language
- Alternate
- Referer

In addition to *language* and *format* (MIME type), there are other dimensions. For details (and strict definitions) have a look to the RFC Transparent Content Negotiation in HTTP [TCN].

Often, servers do not return the variant list. For example, Apache seems only to return the variant list with *406 Not Acceptable*. One can make Apache to always return the variant list (in Alternate) by changing only one line in the source code and recompiling it (thanks to K. Holtman for pointing out the line). But the requirement is for parametrisation servers to return the variant list or subsets. For example:

- VariantList All
- VariantList Language
- VariantList Language MediaType

Note that these parameters do not exist in Apache. It is just an example and proposal.

7.2. Translation

The greatest cost with MWS is translating:
- Original pages
- Corrections

The public expect web sites to be up to date; errors are expected to be corrected immediately. This is very different from paper publications where the public expects errors to be corrected in the next edition.

Hence, often one has to translate many *linguistic segments* (abbreviated to *segment*); a costly business as there is a fix overhead for each translation request, independently of the size. Indeed, most translation services are geared to the translations of full documents.

7.3. Authorship, Translation and Publishing chain (ATP-chain)

ATP-chain is the cycle for multilingual publishing. Traditionally it was a one-way path:
- Authorship: The author writes the source material
- Translation: The translator(s) translate(s) into the target language(s)
- Publishing: The typographer composes the publication

In some cases, this chain could be two ways; e.g., the translator could send back the source material to the author with change requests to facilitate the translation. Also, one has marking from the beginning to automate the whole process.

7.4. Multilingual parallel text

Multilingual parallel texts are translations of each other. For example, the Treaty of Rome in 23 languages.

7.4.1 Source and target languages

The most common case is that the author writes in one source language and it is translated to other target languages. But it is not rare to have multilingual sources; e.g., a document with three chapters each written in a different language. Indeed, in the case of MWS it is quite common.

For a legal point of view, one can have multilingual parallel texts where all the linguistics versions are considered source languages.

8. Legal and miscellaneous

8.1. Disclaimer

This document represents only the views of the author and not necessarily the views of any other parties. In particular, it does not necessarily represent the opinion of the European Commission, his present employer.

8.2. Comments

To send comments to the author and follow-ups see:

```
http://dragoman.org/mws
```

9. References

[DC] Dublin Core Metadata Element Set, Version 1.1
http://dublincore.org/documents/dces

[EU] Europa

http://europa.eu

[ISO-639-1] Codes for the Representation of Names of Languages -- Part 1: Alpha-2 code
http://www.loc.gov/standards/iso639-2/php/code_list.php

[HTML] HTML 4.01 Specification, A. Le Hors et al.
http://www.w3.org/TR/html401

[HTTP] Hypertext Transfer Protocol -- HTTP/1.1., R. Fielding et al.
http://ietf.org/rfc/rfc2616

[IPR] Intellectual Property Rights in IETF Technology
http://www.ietf.org/rfc/rfc3979.txt

[LISA] Localization Industry Standards Association
http://www.lisa.org

[MED] Multilingual Electronic Dossier, M.T. Carrasco Benitez.
http://dragoman.org/med

[MF] Microformats
http://microformats.org

[MWP] Panel, The Multilingual Web: anything missing ?, 9th International WWW Conference
http://www9.org/w9-panels.html

[MWS] Multilingual Web Site BOF, 15th International WWW Conference
http://www2006.org/wiki/w/Multilingual_Web_Site_BOF

[OASIS] Organization for the Advancement of Structured Information Standards
http://www.oasis-open.org

[PARTEXT] Generation of Multilingual Parallel Texts with XML, M.T. Carrasco Benitez
http://www.dragoman.org/partext

[PM] Multilingual Web Sites, WWW2009.org panel, M.T. Carrasco Benitez
http://MultilingualWebSites.org

[UNICODE] The Unicode Consortium
http://unicode.org

[URI] Uniform Resource Identifier (URI): Generic Syntax, T. Berners-Lee et al.
http://ietf.org/rfc/rfc3986

[UTF8] UTF-8, a transformation format of ISO 10646, F. Yergeau
http://www.ietf.org/rfc/rfc2279.txt

[W3C] World Wide Web Consortium
http://w3c.org

[WIMS] Web Internationalization & Multilinguism Symposium
http://www.w3.org/International/Sevilla-96

[WINTER] Winter (Web Internationalization and Multilinguism), Lapsed Internet-draft, M.T. Carrasco Benitez.
http://www.w3.org/International/tomas.carrasco.benitez.html

[WWW2008] 17th International World Wide Web Conference
http://www2008.org

[XDOSSIER] Xdossier, M.T. Carrasco Benitez, Internet-Draft, (work in progress).
http://www.dragoman.org/xdossier

[XHTML] XHTML™ 1.0 The Extensible HyperText Markup Language (Second Edition), S. Pemberton et al.
http://www.w3.org/TR/xhtml1

[XEP-LT] XEP-0171: Language Translation, B. Fletcher et al.
http://www.xmpp.org/extensions/xep-0171.html

[XLIFF] XML Localization Interchange File Forma
http://docs.oasis-open.org/xliff/xliff-core/xliff-core.html

[XML] Extensible Markup Language (XML) 1.1 (Second Edition), T. Bray et al.
http://www.w3.org/TR/xml11