Una guía al TMX

Josu Gómez, Grupo DEli de la Universidad de Deusto

De la Traducción Automática a la Traducción Asistida

Durante la última década del siglo XX comenzó a extenderse el sentimiento de que la traducción automática, disciplina que tantas esperanzas había concitado, no iba a llegar a ser un instrumento definitivo para solucionar los problemas de comunicación entre distintas lenguas. Las características del lenguaje natural llegaban en un determinado nivel de análisis a convertirse en obstáculos insalvables para el procesamiento automático; la ciencia de la ingeniería lingüística asumía que, si había que lograr traducciones que rozaran la aceptabilidad, la intervención del traductor humano iba a ser indispensable.

Como consecuencia de esta percepción, se acentuó la exploración en campos distintos de los más desarrollados hasta el momento. En concreto, el concepto de la traducción automática fue derivando, en muchos grupos de trabajo, hacia el de la traducción asistida (CAT, Computer Assisted Translation); ya no se pretendía que el ordenador hiciera el trabajo, sino simplemente que proveyera de las mayores facilidades posibles al traductor humano, que simplificara y a la vez fortaleciera su trabajo. De esta manera, la Machine Translation (MT) vio como sus siglas se invertían para dar lugar al concepto que más se ha extendido entre los traductores en los últimos tiempos, y que más esperanzas concita para un futuro cercano: las TM (Translation Memories), las Memorias de Traducción.

Memorias de traducción

La idea que subyace tras las Memorias de Traducción (desde ahora TM) es simple: se trata de permitir la posibilidad de reutilizar y aprovechar las traducciones ya efectuadas por uno mismo o por otra persona, para que sirvan de ayuda a la hora de acometer nuevos proyectos de traducción. Utilizando las TMs, cuando un traductor se enfrenta a una frase que ya ha traducido anteriormente, el programa que utiliza puede presentársela, para que no deba traducirla de nuevo, con riesgo de perder homogeneidad. A partir de aquí se abren posibilidades inmensas, que no detallaremos en este espacio.

Durante esa década fueron surgiendo un buen número de utilidades de gestión de TMs. IBM lideró durante un tiempo el mercado con su Translation Manager, pero este puesto fue rápidamente alcanzado por Trados, una empresa participada por Microsoft, gracias a la especial adecuación de su Translator's Workbench al programa Microsoft Word. Transit, de Star, es otra utilidad que se ha mantenido en una buena posición, detrás del TW, sobre todo por la potencia de su gestor de terminología; por último Deja Vú es un software español que está logrando una gran aceptación en el mercado por su gran servicio y su rebajado precio, desbancando en los últimos años a Transit del segundo puesto en las listas de preferencia de los traductores mundiales (si bien la hegemonía de Trados sigue sin ser discutida). En cualquier caso, éstas son unas pocas de las muchas utilidades de gestión de TMs que pueden encontrarse en el mercado.

La proliferación de herramientas dejó al descubierto un problema. La incompatibilidad entre ellas hacía que cuando un traductor comenzaba a recopilar una TM en un programa determinado, desde ese momento se hallaba circunscrito a ese programa, y no podía abandonarlo y comenzar a usar otro distinto sin perder toda la información que había recopilado en su TM. Esta situación era muy problemática para muchos traductores que, por exigencias de sus clientes, debían usar distintas herramientas, y que se veían imposibilitados de compartir su TM con todas ellas.

De manera que pronto se entendió la necesidad de que existiera un método de intercambio de memorias de traducción entre las distintas utilidades. Así nació en 1998 de la mano del consorcio LISA el Translation Memory eXchange, o TMX.

Translation Memory eXchange

TMX es un lenguaje que cumple las especificaciones XML, y cuyo propósito es proporcionar un estándar para el intercambio de las memorias de traducción. Cuando se esté trabajando con una utilidad y se desee pasar a trabajar con otra manteniendo la TM que se había ido recopilando, bastará con exportarla a formato TMX, e importarla en la nueva utilidad. Para ello es necesario que todas las utilidades soporten dicho formato: a 2001, puede decirse que ya hemos llegado a esta situación, puesto que actualmente las herramientas más importantes del mercado admiten la importación y exportación de memorias en TMX, si bien en distintos grados.

El propósito de este trabajo es proporcionar una guía somera al formato TMX, proveyendo de una descripción de sus especificaciones tal y como pueden localizarse en http://lisa.org/tmx, y aclarando algunos de los puntos que quizás puedan quedar más oscuros en su página oficial. Así pues, nuestro objetivo es más divulgativo que científico. No pretendemos realizar profundas disquisiciones sobre cada uno de los elementos o atributos: nos limitaremos a señalar la forma en que se relacionan unos con otros y con la propia memoria de traducción, y daremos una guía a la generación de nuevas TMs etiquetadas en este formato y preparadas ya para ser importadas a cualquier utilidad. Dejamos la discusión de las cuestiones más técnicas a otros artículos.

La descripción del formato que vamos a efectuar está basada en la versión 1.3 de TMX, datada en el 29 de agosto de 2001.

TMX: características generales

Hemos dicho ya que TMX cumple con las especificaciones de XML. Por ello, utiliza los estándares ISO para fechas, códigos de idiomas y códigos de país. Todos sus elementos se escriben en minúsculas, para evitar disfunciones a causa de la distinción que hace XML entre mayúsculas y minúsculas. El formato en que se escriben debe ser Unicode: UCS-2, UTF-8 o ISO-646.

Para los ficheros de 7 bites, los caracteres no ASCII se deben representar en hexadecimal, como á para "á". Se admiten, sin embargo, unas pocas entidades de caracteres: **<** para "<", **>** para ">", **&** para "&", o **"** para las comillas dobles.

Todo esto debe quedar especificado de alguna manera en el fichero, lo que se hace con la Declaración XML, que es **<?xml version="1.0" ?>**, seguida de una declaración DOCTYPE, que será **<!DOCTYPE tmx SYSTEM "tmx12.dtd">**, si es que se desea que el documento se valide contra el DTD propio de TMX. Estas dos declaraciones deberán aparecer en las dos primeras líneas de cualquier fichero TMX.

Una memoria de traducción siempre deberá comenzar con una etiqueta que lo marque como tmx, y que además advierta de la versión del lenguaje que se está utilizando; así, normalmente se abrirá la TM con **<tmx version="1.3">**, y se cerrará con **</tmx>**.

Como suele ocurrir, el <xml> se dividirá en dos únicas partes: un **<header>** y un **<body>**. Ambas etiquetas podrán tener atributos, y ambas contendrán otros elementos dentro de ellas. Vamos a comenzar explicando la información que debemos o que podremos encontrar dentro de la cabecera, y pasaremos posteriormente a explicar los elementos propios del cuerpo.

Header: una cabecera para saber quién somos

<header> es una etiqueta muy exigente: precisa de la existencia de varios atributos. Nos proporcionará información sobre el fichero TMX y su forma de creación. Los atributos obligatorios de **<header>** son los siguientes:

- **creationtool:** El programa con el que se ha creado el fichero TMX. Es obligatorio, porque éste no es un formato entendido para ser creado manualmente, sino que por lo general será generado por distintas utilidades. Dicha utilidad debe especificarse con este atributo.
- **creationtoolversion:** De la misma manera, ha de advertirse de la versión concreta del programa con el que se ha creado el fichero TMX.
- **segtype:** Este atributo especifica el tipo de segmentación que se va a aplicar a las unidades de traducción. Si vamos a fraccionar nuestro texto frase a frase, este atributo tendrá el valor **phrase**. Pero podemos decidir que separaremos el texto por párrafos, por bloques o por oraciones; en esos casos el atributo tendrá el valor **paragraph**, **block** o **sentence**. Veremos esto más adelante.
- **o-tmf:** Esto significa "Original Translation Memory Format", o sea, "Formato de la Memoria de Traducción Original" a partir de la cual hemos creado el TMX. Las especificaciones no aclaran qué hacer en caso de haber creado el TMX partiendo de corpus paralelos, y no de una TM anterior: en este caso, nuestra opción es dejarlo en blanco.
- **adminlang:** Se advierte aquí del código de la lengua en que estarán redactados las notas (elementos <note>) o del valor del elemento <prop> ; más adelante veremos ambas. Podemos decir, en pocas palabras, que se trata de la lengua de trabajo del programador. Deberá ser una de las lenguas especificadas por un atributo **xml:lang**, como pronto veremos.
- **srclang:** Aquí debemos especificar el código de la lengua de origen. En toda memoria de traducción existe una lengua de origen, y una o varias de destino. Aquí debe especificarse la lengua de origen, aunque cabe la posibilidad de dar a este atributo el valor "*all*" (sin las comillas, pero con los asteriscos) para señalar que se puede usar cualquier combinación de idiomas.
- **datatype:** Por último, aquí se especifica el tipo de datos que se contendrán en un elemento. Por defecto su valor será "unknown" (desconocido), pero otros valores recomendados pueden ser "rtf", "transit", "plaintext", "xml", "html", "winres" (recursos Windows), y varios otros que pueden consultarse en http://www.lisa.org/tmx/tmxnotes.htm.

Éstos son los atributos que debe contener la cabecera <header>. Pero existen otros atributos que son opcionales, y que proporcionan informaciones adicionales. Son los siguientes:

- **o-encoding:** Todos los ficheros TMX están en Unicode, como ha quedado dicho; pero a veces es interesante saber qué código había sido usado para codificar el texto antes de que fuera convertido a Unicode. Este atributo especifica el código original (de ahí viene la "o-") en que estaba el texto, para que, si alguna utilidad debe convertirlo de Unicode a otro formato, sepa en qué formato es preferible reescribirlo. Se propone que su valor sea uno de los identificadores de IANA, en http://www.iana.org/assignments/character-sets.
- **creationdate:** Marca la fecha de creación del fichero TMX. Su formato debe ser el siguiente: AAAAMMDD**T**hhmmss**Z**, donde AAAA es el año en cuatro cifras, MM el mes en dos cifras, DD el día en dos cifras, hh la hora, mm los minutos y ss los segundos, todo en dos cifras (esto es, añadiendo un "0" a la izquierda si es preciso). **T** es la propia letra T, que sirve para marcar el comienzo de la parte "tiempo" y el final de la parte "fecha"; y **Z** es la propia letra Z, y advierte de que el tiempo se da en UTC (Tiempo Universal Coordinado, o sea, la hora en Greenwich: dos horas menos que en España en invierno, y una hora menos en verano). Un ejemplo puede ser "20010504T164220Z".
- **creationid:** Aquí se marca el usuario que ha creado el fichero TMX.
- **changedate:** Se especifica en este atributo la fecha de la última modificación del fichero. El valor debe estar en el mismo formato que **creationdate**.
- **changeid:** Debe marcarse aquí el nombre de la persona que ha realizado el último cambio en el fichero

De manera que una cabecera que contara con todos los atributos obligatorios podría ser la siguiente:

<header creationtool="corp2tmx.pl" creationtoolversion="beta" datatype="plaintext" segtype="sentence" adminlang="EU" srclang="ES" o-tmf="">

Elementos de cabecera: la información general

Las cabeceras pueden estar vacías, y no contener ningún elemento dentro de **<header>** (a excepción de los atributos obligatorios, desde luego). Pero también pueden contener alguno de estos tres tipos de elementos: **<note>**, **<prop>** y **<ude>**.

<note> se usa para los comentarios. Un <note> dentro del <head> implica que el comentario se refiere a todo el fichero TMX. Puede incluir el atributo **o-encoding**, del que ya hemos hablado, y que marca el formato de texto original con que fue escrito, o el atributo **xml:lang**, del que hablaremos enseguida, y que señala el lenguaje en que está escrita la nota. No contiene ningún otro elemento dentro de él.

<prop> marca, cuando está dentro del **<header>**, las propiedades del fichero TMX que no sean estándar, y que dependan de la herramienta que se haya utilizado. Estas propiedades no están definidas por el estándar TMX, y por ello puede tratarse de cualquier tipo de datos que nuestra herramienta desee procesar. Debe llevar el atributo **type**, que marca el tipo de datos que el **<prop>** contendrá. Puede utilizarse, por ejemplo, para introducir una serie de instrucciones que deberá tener en cuenta la herramienta que se utiliza.

Además del **o-encoding**, ya explicado, **<prop>** también puede llevar el atributo **xml:lang**. Este atributo especifica el lenguaje en que están escritos los datos, tanto de un comentario (**<note>**) como de un **<prop>**). Por defecto, el valor de este atributo es el mismo que el de **adminlang**. Su valor debe ser un código de los identificadores de lenguaje de ISO, de dos o tres letras (como aparecen en http://www.unicode.org/unicode/onlinedat/languages.html) o los identificadores de país (http://www.unicode.org/unicode/onlinedat/countries.html); para el español habría que definir **xml:lang="es"**, y para el euskera, **xml:lang = "eu"**.

Por último, las cabeceras pueden llevar también el elemento **<ude>**, User-Defined Encoding, o "Codificación Definida por el Usuario". Este elemento marca un conjunto de caracteres definidos por el usuario, que el texto necesitará para su correcta interpretación. Debe llevar un atributo **name**, con el nombre de ese subconjunto de caracteres, y debe contener uno o varios elementos **<map/>**.

Fijémonos un poco en estos elementos **<map/>**. Se trata de elementos vacíos, que transmiten información por medio de sus atributos. En este caso, cada **<map/>** marca un carácter definido por el usuario, cuya información necesitará el programa. Debe llevar el atributo **unicode**, que será el valor Unicode en hexadecimal del caracter; y al menos uno de estos tres: **ent** (que da el nombre de la entidad del carácter), **subst** (que informa de una secuencia de caracteres alternativa que se puede usar para ese carácter, en ASCII), o **code** (que especifica el "point-code", esto es, el "punto de codificación", en hexadecimal). Si el **<map/>** lleva un atributo code, el elemento **<ude>** deberá llevar un atributo base, que especificará el set de códigos en los que se basa la codificación definida por el usuario.

En resumen: la cabecera (**<header>**) nos da información sobre el conjunto del fichero TMX. Esta información puede ser de varios tipos:

- Contextual, como la herramienta con que ha sido creado el fichero (atributos obligatorios **creationtool** y **creationtoolversion**), su fecha y autor de creación (atributos opcionales **creationdate** y **creationid**) o de la última modificación (atributos **changedate** y **changeid**), y la lengua que se ha usado para las anotaciones (atributo obligatorio **adminlang**).
- De formato, sea el formato original en que fue escrito el texto (atributo obligatorio **o-tmf**) o la codificación original que se usó (atributo opcional **o-encoding**), la lengua original en que fue redactado (atributo obligatorio **srclang**), el tipo de datos que contiene (atributo obligatorio **datatype**) o el tipo de segmentación que se ha escogido a la hora de dividir el texto en unidades de traducción (atributo obligatorio **segtype**).
- Comentarios que se refieren a todo el fichero (elemento opcional **<note>**, con sus atributos **xml:lang** para marcar el idioma del comentario, y **o-encoding** para marcar la codificación original en que se escribió el comentario).
- Propiedades no estándar del fichero, que pueden ser utilizadas por nuestra herramienta (elemento opcional **<prop>**, con sus atributos **type**, que marca el tipo de datos que contiene el prop, **o-encoding** y **xml:lang**).
- Información sobre los caracteres no estándar del fichero TMX (elemento opcional **<ude>**, con un atributo **name** que le da nombre, y uno o varios elementos **<map\>**, uno por cada carácter definido por el usuario).

Vamos a analizar ahora, a modo de prueba, una cabecera creada por el programa Transit, tras haber exportado a TMX una memoria de traducción:

```
<header
       creationtool="Transit"
       creationtoolversion="3.0"
       datatype="Transit"
       segtype="block"
       adminlang="en"
       srclang="en-gb"
       o-tmf="Transit"
       creationdate="20010507T083458Z"
       creationid="XTRA-Bi"
       o-encoding="Unicode"
      >
<prop type="Project">Traduccion de
prueba</prop>
</header>
```

Vemos en ella que el TMX ha sido generado por el programa Transit 3.0, en un tipo de datos propio de Transit, que el texto está segmentado en bloques, que la lengua de los comentarios es el inglés, que el lenguaje de original es el inglés de Gran Bretaña, que el formato original es el propio de Transit, que fue creado el 7 de mayo del 2001 a las nueve menos veinticinco de Greenwich por el cliente "XTRA-Bi", que originalmente estaba codificado en Unicode, y que existe una propiedad no estándar llamada "Project" que tiene como valor "Traduccion de prueba".

Como se ve, se ha incluido toda la información posible excepto "changeid" y "changedate", que no eran relevantes al ser un fichero de nueva creación y los comentarios "note", y no se ha querido definir ningún subconjunto de caracteres nuevo por medio del "ude".

Tras haber observado el funcionamiento de la cabecera y la información que nos ofrece, es hora de entrar al meollo del fichero: cómo se organiza la Memoria de Traducción en sí misma.

TUs: se construye con unidades

Además del <header>, el fichero TMX consta de un **<body>** en el que estará incluída toda la información de la Memoria de Traducción. Este body contendrá únicamente elementos **<tu>**.

TU son las siglas de "Translation Unit", o "Unidad de Traducción". Si recordamos el funcionamiento de las TMs, tenemos que los programas que las utilicen deben advertir al traductor sobre cuándo han encontrado en la traducción actual algo que ya ha sido traducido anteriormente. Este "algo" puede ser una frase, una oración, un bloque o un párrafo, dependiendo de que nos interese que el programa traduzca de frase en frase, de oración en oración, de bloque en bloque o de párrafo en párrafo. Esto habrá sido definido, si se recuerda, en el atributo **segtype**.

En tal caso, y poniendo como ejemplo que hayamos dado a **segtype** el valor "phrase", tendremos que la memoria de traducción constará de un gran número de frases, cada una de las cuales tendrá asociada su traducción o traducciones a uno o varios idiomas. Una Unidad de Traducción, una TU, es el conjunto de una frase y su traducción o traducciones. La frase original y cada una de sus traducciones estarán contenidas en elementos **<tuv>**, esto es, Translation Unit Variants, o Variantes de la Unidad de Traducción. Estas Variantes deberán especificar, como es lógico, el idioma al que pertenecen, por medio de un atributo **xml:lang**. Y el texto en concreto de esa versión estará dentro de un elemento **<seg>**.

Para utilizar un ejemplo que sonará conocido, una TU típica podría ser:

```
<tu>
<tuv xml:lang="es">
    <seg>Hola,
    mundo.</seg>                    </tuv>
<tuv xml:lang="en">
    <seg>Hello,
    world.</seg>                    </tuv>
<tuv xml:lang="eu">
    <seg>Kaixo,
    mundua.</seg>                   </tuv>
</tu>
```

De esta manera, cuando el traductor reciba la frase "Hola, mundo.", sabrá que no hace falta traducirla de nuevo; o si lo prefiera, podrá incluso pedir a su utilidad que la traduzca automáticamente, para adelantar trabajo.

Tanto los TUs como los TUVs tienen un buen número de atributos opcionales, que dan información sobre esa frase (o párrafo, o lo que sea) en concreto. Algunos de estos atributos ya los conocemos, como **o-encoding**, **datatype**, **creationtool**, **creationversion**, **creationdate**, **creationid**, **changedate**, **changeid**, **segtype**, **o-tmf** y **srclang**. Esta información puede ser relevante en caso de que nuestra Memoria de Traducción haya sido realizada en distintos momentos y por distintos programas, y hayamos ido acumulando en ella el resultado de trabajos diversos.

Sólo hay dos atributos que no conocemos aún. **Usagecount** nos informa de cuántas veces se ha accedido a esa TU o TUV en concreto por parte del programa gestor de TMs. De esta manera podemos saber qué partes de la Memoria de Traducción son útiles, y cuáles quizás no merezca la pena conservar. En la misma línea, **lastusagedata** nos transmite la fecha de la última vez que el programa accedió a esa TU o TUV, en el mismo formato que se usa en "creationdate". Son atributos opcionales también.

Por último, cada TU podrá tener un identificador, llamado **tuid**, cuyo valor podrá ser cualquiera que el usuario prefiera (como <**tu tuid="frase1"**>, o simplemente <**tu tuid="12"**>, o lo que más convenga).

Para organizar la traducción: elementos de un TU

Cada uno de los TU puede contener varios elementos accesorios. Uno de ellos es el **<note>**, que permite introducir comentarios, que se referirán únicamente al TU (mientras que si este **<note>** se colocaba en la cabecera, se entendía que los comentarios se referían al fichero entero).

Otro es el **<prop>**. Colocado dentro de un **<tu>**, este elemento funcionará de forma algo distinta a como funcionaba al colocarlo dentro de la **<header>**. Aquí, pese a que seguirá proporcionando información de "propiedades no estándar", su función principal será la de agrupar a distintos TUs en un grupo lógico. Incluyendo un **<prop type="proyecto">**25-A-30**</prop>** en cada una de las TUs referentes a ese proyecto se recordará que pertenecen al grupo "25-A-30". Dado que TMX no tiene en cuenta la noción de "orden" entre sus unidades de traducción, ésta será la única forma de reorganizar las frases una vez creada la TM.

En el nivel de los TUVs también pueden aparecer tanto los "note" como los "prop", que tendrán la misma función. Sin embargo, un TUV contendrá también un elemento **<seg>** que, como hemos dicho, tendrá la versión en cada idioma del segmento a traducir.

Pero no será necesario que este texto esté únicamente en ASCII. Para permitir una flexibilidad mayor, y que no se pierda la información de los códigos nativos, dentro de **<seg>** podrá haber otros elementos que nos permitan recuperar la información del formato original, si es que nos interesa.

Cómo recuperar el formato: Etiquetas

Los distintos procesadores de texto existentes utilizan "etiquetas" (o "tags") para señalar la información de formato. Estas etiquetas suelen constar de un "Comienzo de Etiqueta" y un "Fin de Etiqueta", y se supone que se debe aplicar su formato a todo el texto que está entre ambos. Para mantener esta información, TMX tiene dos elementos:

> **<bpt>** (Begin Paired Tag, o Comienzo de Etiqueta Emparejada) y
> **<ept>** (End Paired Tag, o Fin de Etiqueta Emparejada).

Cada uno de ellos marca una secuencia de códigos nativos que se usan, en el texto original, para definir un comienzo o fin de etiquetas de formato. Pongamos un ejemplo: en una línea típica de HTML, la información de "negrita" se marca con las etiquetas "" y ""; la negrita se aplica al texto que está entre ambas. En TMX, "" iría a un elemento **<bpt>**, y "</bpt>" a un elemento **<ept>**.

> HTML: "Esto es una prueba de una frase en negrita, sin más"
> TMX: "Esto es una prueba de una frase en **<bpt>****</bpt>** negrita **<ept>**<\b>**</ept>**, sin más"

Si esta frase estuviera en RTF v.1, tendríamos "una frase en {\b negrita}, sin más", y al pasarlo a TMX, se convertiría en "una frase en **<bpt>**{\b**<\bpt>** negrita **<ept>**}**</ept>** , sin más".

Podemos dar más información a los elementos **<bpt>** y **<ept>**. Por ejemplo, podemos decirle el tipo de etiqueta que es, por medio del elemento **type** (como **<bpt type="bold">**). Además de "bold" se recomiendan otros como "italic", "ulined" (subrayado), "scaps" (small caps, versalita), "link" (texto con enlace) y alguno más.

Podemos decirle también cómo tienen que emparejarse, porque muchas veces ocurre que, al encontrar un cierre de etiqueta, no está claro a qué etiqueta cierra. Para ello se usa el atributo identificador **i**, como en **<bpt i="1">**. De esta manera, una línea en HTML como:

> HTML: Hay negrita, negrita e italica, e italica sola

en la que se cierra antes la etiqueta de negrita sin haber cerrado la de itálica, sería en TMX:

> TMX: Hay **<bpt i="1">****</bpt>** negrita, **<bpt i="2">****</bpt>** negrita e italica, **<ept i="1">****</ept>** e italica sola **<ept i="2">****</ept>**

Y a veces es interesante también poder emparejar las distintas etiquetas entre las distintas TUVs. A veces, en una frase en castellano puede haber un segmento en negrita y otro en itálica, que, a causa de la diferente sintáxis de los dos idiomas, en su versión en euskera estén situados en distinto orden, primero en itálica y después en negrita. Pero nos puede interesar que el programa reconozca sus verdaderas equivalencias: para ello existe el atributo **x**.

Los atributos **x** e **i** son muy similares, pero mientras que **i** empareja etiquetas dentro del mismo segmento, **x** las empareja dentro de las distintas TUVs de una TU. Un ejemplo típico podría ser:

TMX:

<tu>
<tuv>
<seg>
<bpt x="1">**</bpt>** Creo **<ept>****</ept>** que **<bpt x="2">****</bpt>**
hoy **<ept x="2">****</ept>** va a llover
</seg>
</tuv>
<tuv>
<seg>
<bpt x="2">**</bpt>** Gaur **<ept x="2">****</ept>** euria egingo duela
<bpt x="1">**</bpt>** uste dut **<ept x="2">****</ept>**
</seg>
</tuv>
</tu>

Si el atributo **x** no estuviera presente, el programa podría emparejar la etiqueta itálica de la primera TUV con la negrita de la segunda. De esta manera el emparejamiento se produce correctamente.

A veces surgen problemas con las etiquetas, dado que, por ejemplo, a causa de la segmentación, es posible que una etiqueta se abra en un segmento pero se cierre en un segmento posterior. Por ejemplo, puede haber una etiqueta de "abrir negrita" en una línea, pero no cerrarse hasta varias líneas después. Estas etiquetas que aparecen aisladas en un segmento se marcan con el elemento **<it>**, que quiere decir "isolated tag" o "etiqueta aislada". Funcionan igual que las **<btp>** o **<ept>**, sólo que necesitan de un atributo más, **pos**, que diga si es una etiqueta de comienzo (**pos="begin"**) o de fin (**pos="end"**).

Otra posible situación es la aparición de elementos que van solos, como por ejemplo las notas a pie de página, las imágenes... No se trata de etiquetas emparejadas que han quedado aisladas, sino de etiquetas que por su propia naturaleza van solas. Para este tipo de objetos se utiliza el elemento **<ph>** que significa "place holder". Puede llevar el atributo **type** (para el que se recomiendan valores como "image" para imagen, "fnote" para nota a pie de página, "enote" para nota al final, "alt" para texto alternativo, y otros), el atributo **x**, que sirve para emparejarlo con otros de distintas TUVs dentro de su misma TU, como ya hemos visto, y un atributo nuevo, **assoc**, que nos informará de si este elemento debe ser asociado al texto precedente (en cuyo caso su valor será "p", de "previous"), al texto siguiente (en cuyo caso su valor será "f", de "following") o a ambos (en cuyo caso su valor será "b", de "both").

Este elemento **<ph>** contiene un elemento **<sub>** que tiene bastante importancia. **<sub>** puede aparecer en cualquier elemento de etiqueta de los que ya hemos visto, y marcará el texto que se encuentra dentro de las etiquetas. En un caso como una <ph type ="fnote">, una nota a pie de página, este elemento <sub> será donde se inserte el texto de la nota.

Para dar un ejemplo, tomamos el que aparece en la documentación:

RTF:

Los elefantes
{\cs16\super \chftn
{\footnote \pard\plain \s15\widctlpar \f4\fs20
{\cs16\super \chftn }
Animal de la familia de los paquidermos.
}
}
son grandes.

TMX:

Los elefantes
<ph type="fnote"> {\cs16\super \chftn
{\footnote \pard\plain \s15\widctlpar \f4\fs20
{\cs16\super \chftn }
_{**Animal de la familia de los paquidermos.**}
}
}
</ph>
son grandes.

Existe también un elemento **<ut>**, que significa "unknown tag", y se utiliza para las "etiquetas desconocidas". Sólo puede llevar el atributo x ya conocido.

Y por último hay que citar un elemento más, **<hi>**, de "highlight", "subrayar", cuya función es marcar porciones de segmentos para cualquier propósito definido por el usuario. Pueden llevar el atributo **x** o el **type**.

Con estos elementos, un archivo TMX puede mantener no sólo el texto de los archivos originales, sino también sus marcas de formato y la relación entre ellas.

Tres niveles de información, tres niveles de implantación

Cuando se habla de la implantación del formato TMX en las distintas herramientas de gestión de Memorias de Traducción, se habla de que dicho formato está implementado "en el nivel 1", "en el nivel 2" o "en el nivel 3". Las páginas web de las distintas utilidades no dejan claro en qué nivel implementan TMX, excepto algunas como Trados o SDLX, que reconocen una implementación en el nivel 2.

¿A qué se están refiriendo al hablar de estos "niveles"?

Pues, simplemente, al reconocimiento de los distintos códigos de formato y etiquetas.

- El Nivel 1 es aquel en el que no se toman en consideración los códigos de formato ni las etiquetas. Únicamente se reconoce el texto plano que aparece dentro de cada elemento **<seg>**, sin anotaciones de contenido. Es la opción menos eficiente, pero es la que da mejores resultados a la hora de buscar las equivalencias entre los textos, dado que la comparación textual no se ve incomodada por ningún código.
- El Nivel 2 es aquel en el que se toman en consideración las etiquetas en su formato TMX; esto es, la aparición de **<btp>** y **<etp>**, o de **<ph>**, pero no incluye los códigos nativos de dichas etiquetas.
- El Nivel 3 asume tanto las etiquetas TMX como el código nativo que aparece en cada elemento de formato. Con este nivel no se pierde nada de la información, y

se facilita la recuperación de los formatos originales.

Estos códigos trabajan junto a las declaraciones del formato original (el **o-encoding** de la cabecera y el **datatype** de la cabecera, TUs, TUVs y sub).

Resumen: lo que puede y no puede transmitirse
En resumen, lo que puede transmitirse en una memoria de traducción dentro de un <body> es lo siguiente:

- Unidades de Traducción (**<tu>**), con sus posibles anotaciones y propiedades específicas, y con Variantes de la Unidad de Traducción (**<tuv>**) para cada idioma. Tanto en los TUVs como en los TUs habrá información contextual sobre la frase; en el caso del TU también un identificador de unidad, un TUID.
- Dentro de cada TUV habrá un SEG, que contendrá el texto plano, y las etiquetas de formato. Éstas podrán ser etiquetas de comienzo y de fin de un formato (**<bpt>** y **<ept>**), emparejadas o aisladas (en este caso serán **<it>**, con un atributo que dirá si es de principio o de final); etiquetas que van solas (**<ph>**, asociadas al texto anterior, al posterior o a ambos); o etiquetas desconocidas. Estas etiquetas estarán relacionadas entre sí, cada comienzo con su fin, y entre las distintas TUVs, para asegurar la alineación correcta de las mismas. Además, podrán marcarse porciones de los segmentos, para propósitos ulteriores

Ésta es, básicamente, la estructura de los ficheros TMX, y ésta es la información que se guarda a la hora de intercambiar Memorias de Traducción (TMs). Hemos presentado aquí, básicamente, una traducción y exposición de dos documentos básicos del TMX: "TMX Format: Specifications", http://lisa.org/tmx/tmx.htm, y "TMX Format: Implementation Notes", http://lisa.org/tmx/tmxnotes.htm . En cualquier caso, siempre puede acudirse a estas dos fuentes para aclarar cualquier cosa que hayamos podido explicar defectuosamente, y para datos técnicos que hayamos podido pasar por alto.