Recomendaciones para desarrollar software internacionalizado

Recommendations for developing internationalized software

Ricardo Payán Goméz, Ing., Julián Barbosa, Ing., & Miguel E. Torres Moreno, M.Sc.
Pontificia Universidad Javeriana, Sede Bogotá, Colombia
rpayan6@gmail.com, jbarbosam983@hotmail.com, metorres@javeriana.edu.co

Recibido para revisión 01 de octubre de 2010, aceptado 28 de junio de 2011, versión final 30 de junio de 2011

***Resumen*— El presente artículo tiene como objetivo exponer una serie de recomendaciones que deberían ser tomadas en cuenta al construir software internacionalizado, es decir aquel que está libre de toda dependencia cultural y está optimizado para ser localizado posteriormente a una región geográfica específica.**

Inicialmente se presenta una introducción al tema y se describe la estrecha relación entre la internacionalización y la localización de software y posteriormente se describen 19 recomendaciones enmarcadas dentro de las 4 primeras áreas de conocimiento del SWEBOK[1]: Requerimientos, Diseño, Construcción y Pruebas.

***Palabras Clave*— Internacionalización de software, i18n, Localización de software, l10n, Globalización, Ingeniería de software.**

***Abstract*— The following article presents a series of recommendations which should be taken into account when internationalized software is being developed. Internationalized software is software free of any cultural dependencies and it is also optimized to be located on another geographical region.**

First we present a brief introduction to the topic and the close relationship between software internationalization and software localization, and then 19 recommendations are presented in four first stages of the framework proposed by the SWEBOK[1]: requirements, design, development and testing.

***Key Words*— Software internationalization, i18n, Software localization, l10n, Globalization, Software engineering.**

I. INTRODUCCIÓN

La Ingeniería de Software, y en particular, los procesos de construcción de software deben adaptarse con mayor rapidez a un mundo cada vez más globalizado[3] . Este artículo presenta una serie de recomendaciones que permiten la construcción de software internacionalizado considerando todas las fases incluidas en la mayoría de las metodologías de desarrollo de software abarcando desde la planeación del proyecto hasta la fase de pruebas e implantación.

Sería un grave error pretender que, de entrada, un producto de software desarrollado en un mercado pueda ser localizado para satisfacer todas las necesidades de otro mercado diferente, ya que debe estar preparado técnica y culturalmente[4] de modo que se pueda realizar una posterior localización sin manipular el código fuente y se retiren todas las características propias de la cultura para la cual fue diseñado; dicha preparación se conoce como internacionalización[5].

Es por esta razón que se hace evidente contar con algunos lineamientos generales para garantizar el éxito de la internacionalización de software con el fin de que más adelante el proceso de localización pueda ser ejecutado de manera más sencilla reduciendo costos y tiempo.

La estructura del presente artículo es la siguiente: en la Sección 2 se describe la globalización de mercados desde la perspectiva del desarrollo de software y se explica cómo debe ser el ciclo de desarrollo de un producto global describiendo sus componentes de internacionalización y localización, en la Sección 3 se enuncia cuándo es necesario internacionalizar software y se detallan los aspectos involucrados en dicho proceso, en la Sección 4 se presentan algunos detalles adicionales de la localización de software, en la Sección 5 se describen las buenas prácticas propuestas de internacionalización de software enmarcadas dentro de las áreas de requerimientos, diseño, construcción y pruebas. Finalmente se presentan unas breves conclusiones del tema de globalización de software y de las buenas prácticas de internacionalización propuestas en el presente artículo.

II. GLOBALIZACIÓN DE SOFTWARE

La globalización en el contexto del desarrollo de software es un proceso que implica tanto la internacionalización como la localización (Fig. 1). En otras palabras, es el proceso mediante

Revista Avances en Sistemas e Informática, Vol.8 No.2, julio de 2011 - Medellín. ISSN 1657-7663

el cual una compañía de software amplía su mercado de origen para perseguir oportunidades de negocio en los lugares donde estén ubicados sus potenciales clientes[6].

Figura 1. Componentes de la globalización de software

Cuando las empresas fabricantes de software se percataron de las enormes oportunidades al ingresar sus productos en nuevos mercados, comenzaron a crear departamentos internos de traducción, aproximadamente a comienzos de la década de 1980[5], donde se pretendía adaptar los programas y aplicaciones para que pudiesen ser ofrecidos en otros países (diferencias culturales e idiomáticas).

Como se observó en la Figura 1, el proceso de globalización está compuesto por las fases de internacionalización y localización, razón por la cual, para que este proceso sea exitoso se requiere que se planeen cuidadosamente dichas fases, al igual que las necesidades continuas de soporte y mantenimiento[5].

La internacionalización, a menudo abreviada como i18n, es el proceso de generalizar un producto para que pueda manejar múltiples idiomas y convenciones culturales sin la necesidad de rediseñarlo[7]. Esto significa que se eliminan suposiciones culturales y cualquier contenido específico para un país o idioma, esta información se almacena de manera externa al producto para que se pueda adaptar fácilmente.

Por otro lado, la localización, a menudo abreviada como l10n, es la adaptación lingüística y cultural de un producto para el locale (país/región e idioma) objetivo. Su objetivo principal es dar la sensación al usuario que el producto fue diseñado para su país o región[5]. Por lo tanto incluye no sólo la traducción, en el sentido convencional de la palabra, sino también cambios en los estándares regionales (el formato de fecha, hora y moneda), la adaptación de iconos y colores, etc.

El ciclo de desarrollo propuesto por la Asociación de Estándares de la Industria de la Localización (LISA) en "The Globalization Industry Primer" [6] que se muestra en la Figura 2, inicia con el análisis de requerimientos del producto global/ local para determinar el alcance del proyecto y divide el desarrollo en las fases de internacionalización y localización. Es importante que las organizaciones traten la globalización como un ciclo para que comprendan que este proceso se debe relacionar con todos los niveles de la organización con el fin de ser exitosa.

Figura 2. Ciclo de Desarrollo de un Producto Global adaptado de [6]

En la Figura 2, se puede observar que el primer paso para desarrollar un producto global es un análisis básico de requerimientos donde se determine cuál es la funcionalidad, el contenido y los requerimientos de calidad que los usuarios necesitan (sin importar en qué lugar del mundo se encuentren). La diferencia entre un análisis de requerimientos convencional y uno global es que el segundo se debe realizar no sólo para el mercado local sino para todos los mercados potenciales[7]. También se debe incluir una evaluación de obstáculos y riesgos potenciales en cada mercado y cómo se debe actuar para superarlos[6].

Desde el punto de vista económico, se deben analizar cuidadosamente los mercados potenciales, su tamaño y los costos y beneficios de la globalización[8] para determinar si se lleva a cabo la localización para un mercado específico y, en caso afirmativo, cuál va ser el volumen de dicha localización.

El segundo paso dentro del ciclo es el diseño del producto internacionalizado donde se eliminarán todas las características que son específicas para una cultura de los diferentes elementos incluidos en el software incluyendo gráficas, colores, íconos, sonidos entre otros.

En las pruebas y el proceso de aseguramiento de calidad, se verifica que efectivamente el producto haya quedado correctamente internacionalizado, que sea válido para una audiencia global y que esté libre de dependencias culturales[9].

Al pasar a la fase de localización, primero se realiza la localización del producto para los mercados específicos incluyendo todos los aspectos técnicos, lingüísticos y culturales, y después se realizan las pruebas y el aseguramiento de calidad de la localización. Este aseguramiento idealmente debe ser realizado por expertos técnicos junto con un equipo local[5] que determine la validez del software para el mercado objetivo.

Finalmente, una vez el producto esté localizado e ingrese a un nuevo mercado, inicia la fase de mercadeo local y soporte en la cual se requiere desarrollar canales claros para que los clientes informen acerca de errores[6] junto con procesos que aseguren que dichos errores sean resueltos por el equipo central de desarrollo.

III. INTERNACIONALIZACIÓN (I18N) DE SOFTWARE

Existen dos razones fundamentales por las cuales puede ser necesario realizar el proceso de internacionalización de software[3]:

1. Para asegurar que el producto sea funcional y sea aceptado en los mercados internacionales.
2. Para asegurar que el producto sea localizable.

Para contextualizar el alcance de un proyecto de internacionalización (i18n) de software se pueden analizar las capas de internacionalización propuestas por Gerhard Chroust[10] y que incluyen entre otros, los siguientes aspectos:

- Aspecto tecnológico: está relacionado con procesos técnicos como separación de archivos localizables del código, reserva suficiente de espacio para textos, adecuada codificación de caracteres entre otros. Es importante resaltar que la arquitectura de la aplicación de software que será internacionalizada puede ser objeto de modificaciones a nivel de base de datos y otros componentes.
- Aspecto gramatical: son las reglas literarias que se aplican tanto para el texto incluido en el software como para el que se genera en tiempo de ejecución con el fin de evitar errores en las posteriores traducciones.
- Aspecto semántico: debe haber una adecuada definición de de la terminología pertinente (técnica vs. común), y las abreviaturas utilizadas en la aplicación y en la documentación.
- Aspectos culturales: son los elementos que son específicos para una cultura y que se deben eliminar para evitar confrontaciones y poder enfocar el producto de software a una audiencia global; a continuación se presentan algunos ejemplos[11]:
  - Imágenes: Las figuras humanas y señas de mano pueden tener diferentes significados según la cultura; las personas se identifican con diferentes colores de piel y cabello, las mujeres puede que no realicen ciertas tareas debido al grado de machismo de la cultura, no existe una seña de manos aceptada universalmente y toda seña de manos puede ser ofensiva en alguna cultura.
  - Color: Los colores tienen diferentes significados según la cultura y el contexto en el cual se presenten. Por ejemplo, el blanco usualmente se asocia con la bondad y la pureza en la cultura occidental mientras que en culturas orientales representa la muerte.
  - Sonidos: Mientras que un sonido de timbre utilizado en concursos de TV es reconocido en la cultura occidental y representa respuestas incorrectas, en otras culturas no es más que un sonido cacofónico desagradable sin significado alguno.

IV. LOCALIZACIÓN DE SOFTWARE

Para muchas personas, la localización parece ser solamente un proceso lingüístico idéntico o similar a la traducción. Sin embargo, abarca una mayor complejidad puesto que se debe integrar con otros procesos del negocio para ser efectiva; es una parte integral de la globalización y sin ella los esfuerzos de globalización no arrojan ningún resultado[6].

La localización involucra la adaptación de cualquier aspecto de un producto o servicio requerida para vender o utilizarlo en otro mercado. Este proceso impacta significativamente las funciones técnicas y administrativas dentro de las organizaciones incluyendo la forma como se organizan los planes, se realizan las ventas y se diseñan, construyen y soportan los productos y servicios[12].

Un estudio realizado por LISA[13] demuestra que las compañías desarrolladoras de software y de tecnologías de la información subcontratan en un 52% la localización de software y en un 49% la localización de páginas web mientras que la internacionalización tan solo la subcontratan en un 37%; esto puede presentarse por el hecho de que la internacionalización debe estar presente en todas las fases del desarrollo de un proyecto de software mientras que la localización se puede realizar en la fase final.

V. BUENAS PRÁCTICAS DE INTERNACIONALIZACIÓN DE SOFTWARE

Las buenas prácticas propuestas en esta sección, tienen como objetivo permitir que una persona que de algún modo esté relacionada con el desarrollo de software, empiece a considerar todas las diferencias entre un proceso de construcción de software para un mercado local y un proceso equivalente para un mercado internacional. Además, cualquier desarrollador de software que aplique las consideraciones descritas aquí podrá simplificar su trabajo, evitando cometer errores que ya se han identificado en el pasado y que pueden representar altos costos en la construcción de software internacionalizado.

Desde el momento en que se identifica un requerimiento de internacionalización (i18n) para el desarrollo de un producto de software, es necesario que la metodología de desarrollo esté alineada con dicho requerimiento[14].

Debido a que este artículo busca ser de utilidad en todos los proyectos de desarrollo de software internacionalizado sin importar la metodología empleada, se ha planteado utilizar el SWEBOK: "Guide to the Software Engineering Body of Knowledge"[1] publicado en 2004 por la IEEE (Institute of Electrical and Electronics Engineers) como documento base para asociar las buenas prácticas aquí propuestas.

Las áreas de conocimiento que tienen relación con las buenas prácticas de internacionalización que se presentan en este documento, se ilustran en la Figura 3.

Figura 3. Áreas de Conocimiento SWEBOK relacionadas con la i18n

5.1 Requerimientos de Software

Los Requerimientos de Software expresan las necesidades y restricciones asociadas con un producto de software que contribuye a solucionar algún problema del mundo real[15]. Esta área de conocimiento refleja el hecho de que un proceso de requerimientos exitoso debe ser considerado como un proceso que involucra actividades complejas y altamente acopladas (tanto secuenciales como concurrentes), en vez de ser una actividad llevada a cabo al comenzar un proyecto de desarrollo de software[1].

La internacionalización de un producto de software se define desde esta área del conocimiento como una característica que debe incluir el software una vez se haya finalizado su desarrollo. Con base a estos requerimientos se planea el diseño, se ejecuta la construcción y se realizan las pruebas para verificar que el producto final de software pueda manejar múltiples idiomas y convenciones culturales[5].

5.1.1 Recomendaciones de Internacionalización

1) **Determinar el alcance de la internacionalización del producto para apoyar los requerimientos de localización:** Se debe determinar cuál es la necesidad real que generó el requerimiento de internacionalización para ajustar las actividades que se desarrollarán de manera consecuente.

   El alcance de las actividades necesarias para la internacionalización depende de los mercados en los cuales se desee adaptar el producto, si la internacionalización de una aplicación realizada en Colombia se lleva a cabo para apoyar la posterior localización al mercado japonés, la complejidad será mayor que si el software se internacionaliza para luego adaptarlo al mercado norteamericano.

   Se recomienda hacer un estudio de factibilidad para identificar mercados potenciales para los cuales se puede localizar el software una vez se encuentre internacionalizado[16].

2) **Garantizar que no se violen condiciones legales específicas para ciertos países:** Se debe eliminar cualquier característica del software que pudiese comprometer su ingreso en mercados locales desde un punto de vista legal[17].

   Ciertos tipos de funcionalidades o características del producto de software desarrollado pueden afectar la distribución internacional del producto, por esta razón es importante considerar los países en los cuales se pretende localizar el producto posteriormente y hacer un estudio legal que determine los posibles obstáculos en estos mercados.

   Dichos obstáculos pueden incluir afirmaciones comparativas con productos de la competencia[18], algoritmos de cifrado[4] y términos de licencia[17], entre otros.

3) **Asegurarse de que el software permita cumplir con estándares regionales:** El software internacionalizado debe soportar todos los estándares regionales para los países o regiones donde sea probable su posterior localización, eliminando cualquier suposición respecto a formatos que sean específicos para cada región.

   Esta recomendación implica ciertas consideraciones técnicas como el uso de un entorno de desarrollo que soporte el concepto de locale (el cual permite definir ciertos formatos de datos que son específicos para determinado idioma/región) y el desarrollo de funciones internas que permitan convertir datos según los diferentes formatos de medición.

   Para que la definición de un locale sea aceptable[18], ésta debe incluir por lo menos formatos de:

- Medidas
- Hora y Fecha
- Calendario
- Moneda
- Dirección

  Números (separadores decimales, de miles, negativos)

4) **Concientizar al equipo humano de la importancia de su participación para la internacionalización del software:** Todo el equipo que esté involucrado con la construcción del producto de software debe estar comprometido con el éxito de la internacionalización de software. A su vez, la conformación de un equipo eficiente involucrado en los diversos aspectos del desarrollo de software

internacionalizado permite aumentar las probabilidades de éxito del producto final[16].

5.2 Diseño de Software

Se describe como el "proceso de definir la arquitectura, componentes, interfaces y otras características de un sistema o componente"[19]. El diseño de software se puede resumir como el área donde se analizan los requerimientos para producir una descripción de la estructura interna del software que servirá como la base para su construcción[1].

5.2.1 Recomendaciones de Internacionalización

1) **Uniformidad de terminología en la documentación del software:** Se debe garantizar que los autores de los textos incluidos en menús, diálogos, botones, etc. y de la documentación (manuales, ayuda en línea, etc.) mantengan una terminología consistente con el tema y en todas las secciones del software.

   La presentación de la documentación al usuario final es vital para garantizar la calidad del software[5]. Es inaceptable que se utilicen ciertos términos con un nombre y que en otra sección del software se utilice otra palabra para hacer referencia al mismo elemento[20].

   Para cumplir con esta recomendación, se debe considerar la creación de un glosario terminológico relacionado con la aplicación de software; dicho glosario se convierte en una herramienta muy poderosa para poder transmitir el mensaje que se desea al usuario final en su idioma, sin confundirlo con diferentes términos para un mismo elemento.

2) **Expansión de las interfaces gráficas:** Se debe garantizar que las interfaces gráficas de usuario admitan una expansión para permitir que el texto incluido en ellas se adapte correctamente una vez se haya localizado, de manera que no se afecte la presentación ni la funcionalidad de dichas interfaces[5].

   Cuando no se considera el espacio adicional que se requiere al traducir los textos de ciertos idiomas a otros (hay casos donde se duplica la cantidad de caracteres necesarios para expresar la misma idea en otro idioma), se afecta el diseño de las interfaces ya que muchas veces el texto desborda sus posiciones iniciales y se deben asignar recursos para rediseñar las interfaces una vez se han localizado[7].

   Es importante considerar no sólo la expansión horizontal debido a un mayor número de caracteres sino la expansión vertical que se presenta cuando se localiza el software a un idioma que utiliza caracteres de doble-byte como chino, japonés y coreano.

3) **Soporte adecuado para caracteres internacionales:** Además de contar con la capacidad de mostrar todos los caracteres necesarios en la interfaz de usuario, los usuarios deben poder ingresar datos internacionales usando su método preferido de entrada.

   Una aplicación de software no puede ser considerada como internacionalizada si desde su fase de diseño no define la necesidad de contar con un estándar de codificación de caracteres que le permita ingresar y mostrar la cantidad de caracteres requerida para escribir texto en la mayoría de idiomas utilizados en el mundo[21].

   Para cumplir con esta recomendación, se debe considerar el uso del estándar Unicode que le provee al usuario la capacidad de codificar la mayoría de los caracteres usados en los lenguajes escritos de todo el mundo[2]. Básicamente, Unicode proporciona un número único para cada carácter, sin importar la plataforma, el programa, ni el idioma, permitiendo una fácil transmisión entre distintos sistemas de codificación y plataformas.

   Unicode soluciona el problema que se generaba cuando un carácter tenía valores diferentes de codificación según el sistema utilizado[22].

4) **Diseño Internacional de Interfaces de Usuario:** Todas las interfaces de usuario deben ser diseñadas para que tengan aceptación por parte de un público internacional. Esto implica considerar aspectos técnicos y culturales.

   Las interfaces de usuario son el medio principal mediante el cual el usuario final interactúa con el software; si un usuario percibe que la interfaz tiene cierta predisposición cultural no se puede afirmar que el software esté internacionalizado[5]. Además, el diseño internacional de las interfaces de usuario es primordial para facilitar el posterior proceso de localización.

   Para cumplir con esta recomendación se deben tomar en cuenta las siguientes consideraciones[18][23]:

   1. Minimizar la concatenación de cadenas de caracteres para combinar elementos en tiempo de ejecución y formar nuevas cadenas.
   2. Los comandos de combinación de teclas deben quedar en un repositorio como recurso traducible y por lo tanto no se deben introducir directamente en el código fuente.
   3. Las coordenadas de los elementos de texto en la interfaz no deben ser fijas sino parametrizables.
   4. Se debe evitar el uso de texto o letras incrustadas en los archivos de tipo gráfico.
   5. No se debe restringir el tamaño de la fuente.

5) **Separar la capa de datos de la capa de presentación:** La capa de presentación es la que define cómo se va a mostrar la información al usuario final; dicha información se debe separar de algún modo para disminuir la complejidad en el proceso de localización[5].

   Desde la fase de diseño del software se debe definir una estrategia clara para lograr separar la capa de presentación

Revista Avances en Sistemas e Informática, Vol.8 No.2, julio de 2011 - Medellín. ISSN 1657-7663

de la de datos, ya que en el proceso de localización generalmente sólo es necesario adaptar la capa de datos y no se modifica la capa de presentación en las interfaces.

Para cumplir con esta recomendación se debe considerar el uso de hojas de estilo por cascadas (CSS) para definir conjuntos de páginas HTML[24] o definir si se puede usar un patrón de arquitectura de software que permita separar las capas como el Modelo-Vista-Controlador (MVC)[25].

6) **Internacionalizar el Diseño del Contenido Web que haga parte del Software:** Para construir software que utiliza elementos web tales como páginas HTML o páginas dinámicas, dichos elementos se deben diseñar de modo que permitan simplificar el alojamiento y la presentación de información localizada en fases posteriores[5].

Debido a que el contenido de los sitios web, generalmente es generado de manera dinámica usando scripts, plantillas y bases de datos, es importante planear el diseño de un sitio web de manera adecuada desde un principio para evitar sobrecostos y retrasos en los procesos de mantenimiento en las versiones localizadas.

Para cumplir con esta recomendación se debe considerar la construcción de una estructura multilingüe para el sitio web, desarrollar un sistema de navegación para el sitio según el idioma preferido por el usuario final[23], garantizar que la base de datos y la plataforma que soporta las páginas web dinámicas permitan soportar caracteres de doble byte[5] y desarrollar un plan de actualización y mantenimiento del sitio[7].

7) **Optimizar los esquemas de la base de datos para que soporten la internacionalización:** Para diseñar una base de datos que soporte el almacenamiento de contenido localizado se deben identificar cuáles valores se deben almacenar de forma dependiente de la cultura y cuáles valores se deben almacenar en una representación uniforme que posteriormente se pueda convertir por la lógica de la aplicación[26].

La representación de datos contenidos en la base de datos debe ser independiente del locale y la conversión de datos a partir de la representación original debe ser realizada sin que haya pérdida de información.

El esquema Entidad-Relación debe incluir datos del locale en el modelo de datos, siendo parte de la entidad que representa al recurso localizado tal y como se muestra en la Figura 4 tomada de [27] donde los detalles de las entidades "categoría" y "producto" permiten separar los datos por cada locale implementado.

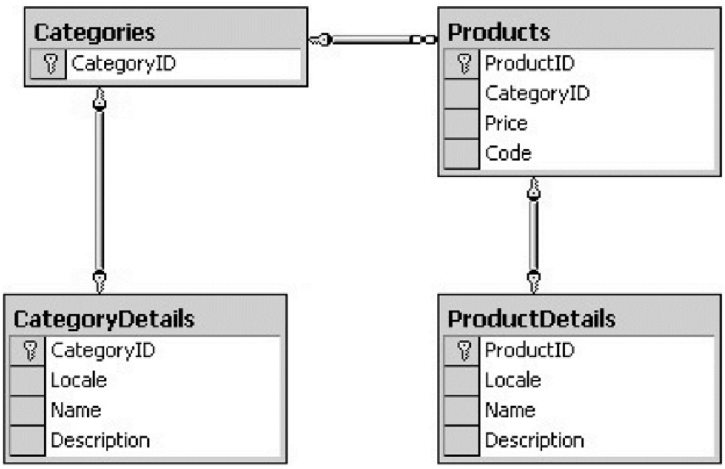

Figura 4. Esquema E-R que soporta la i18n tomado de [27]

5.3 Construcción de Software

Esta área por lo general, produce el mayor volumen de entregables relacionados con el proyecto de software [29]. Se relaciona con el trabajo del desarrollador y su resultado más tangible es el código fuente que siguiendo unas reglas de un lenguaje de programación específico permite cumplir con los requerimientos iníciales siguiendo la arquitectura establecida en la fase de diseño[1].

5.3.1 Recomendaciones de Internacionalización

1) **Estilo internacional en la construcción de documentos:** Se debe garantizar que los autores de la documentación (texto incluido en ayuda, menús, diálogos, botones, etc.) mantengan un estilo adecuado para permitir un correcto uso por parte de los usuarios finales en cualquier región del mundo y de usuarios locales cuando en fases posteriores se les presenten contenidos localizados.

Los documentos deben ser escritos con un estilo que sea aceptable a nivel global[28]. La funcionalidad del software depende en gran medida de que el estilo utilizado sea aceptable para todos los usuarios. Se deben minimizar las probabilidades de errores de traducción en fases posteriores de localización.

Para cumplir con esta recomendación se debe evitar el uso de acrónimos o abreviaturas, incluir conversiones de

medidas para los sistemas de unidades más comunes (i.e. 1 m = 3,28 pies) y evitar la referencia a estaciones del año, festividades, días feriados, etc.[18][29]

2) **Correcta gestión de elementos traducibles:** Toda la información que no pertenece al bloque de código en la fase de desarrollo, pertenece al bloque de datos. Dicho bloque es el que requiere localización y se puede centralizar en un archivo de recursos[30].

Externalizar los elementos traducibles en un archivo de recursos ofrece las siguientes ventajas[5]: Eficiencia (para cada nueva versión de idioma de la aplicación, sólo se deben adaptar los archivos de recursos); Seguridad (los localizadores o traductores no tienen acceso al código fuente); Calidad (Reduce la probabilidad de cadenas traducibles faltantes y limita el nivel requerido de pruebas funcionales).

Se proponen las siguientes consideraciones para poder cumplir con esta buena práctica:

1. Mantener un registro de todos los archivos de recursos que se localizarán posteriormente.
2. Usar nombres de variables que no sean palabras "reales" para evitar la traducción de cadenas no traducibles.
3. Usar comentarios en el código para proporcionarle contexto a los traductores
4. Agrupar los recursos de manera lógica, preferiblemente por pantalla y función.

3) **Eliminar dependencias culturales de los elementos audiovisuales del software:** Los elementos gráficos usados en cualquier interfaz de usuario generalmente contienen referencias a alguna cultura en especial, lo cual es adecuado si se desea que el usuario final se sienta identificado, pero en la versión internacionalizada se debe evitar la relación de las gráficas con cualquier cultura en particular.

Las imágenes gráficas son costosas de crear y su modificación también representa un alto costo[5]. Por este motivo las imágenes usadas en la versión internacionalizada son casi siempre las mismas que se utilizan en las versiones localizadas, razón por la cual se debe evitar transmitir cualquier dependencia con alguna cultura en específico.

Para no transmitir dicha dependencia cultural, se deben usar con cautela los siguientes elementos[11][18][23]:

1. Animales, objetos religiosos, referencias a ideologías o símbolos políticos.
2. Figuras humanas.
3. Partes del cuerpo y gestos con las manos.
4. Colores asociados con países o movimientos políticos.

Aunque las anteriores consideraciones pueden sonar muy difíciles de implementar, no se está proponiendo evitar el uso de esos elementos, sino que más bien se propone analizar el contexto ya que por ejemplo un perro puede ser considerado como una mascota en Colombia pero como alimento en China[23].

4) **Crear un kit de localización:** Un kit de localización es un "subconjunto de herramientas, archivos fuente y archivos binarios que se pueden usar para crear una versión localizada de un programa. Generalmente se le entrega a traductores o a proveedores que suministran servicios de localización"[5]. El kit de localización proporciona el material necesario para preparar una propuesta de localización de software y para realizar las actividades de localización.

La preparación de un kit de localización permite disminuir el esfuerzo requerido por los proveedores de servicios de localización para evaluar y preparar el material del proyecto de localización[30];

Entre más completo sea un kit de localización, mayor será la precisión por parte del proveedor de servicios de localización para cotizar y planear el proceso, y menor será el tiempo invertido por los gerentes de proyecto y los ingenieros.

Se propone que dicho kit de localización deba incluir, por lo menos, lo siguiente[31]:

1. Instrucciones de Localización
2. Archivos fuente a localizar
3. Material de referencia
4. Lista de contactos involucrados en el proyecto
5. Estructura de directorios para organizar el kit.

5) **Usar Lenguaje Controlado:** Controlar el uso del lenguaje es un factor importante para reducir costos en la escritura de la ayuda en línea y la documentación, y el tiempo necesario para desarrollarla y actualizarla[32].

El uso de un lenguaje controlado implica, entre otras cosas, lo siguiente[33]:

1. Mantener las frases cortas (20 palabras o menos)
2. Repetir los nombres en vez de utilizar pronombres siempre que sea posible.
3. No utilizar argot o lenguaje coloquial
4. Utilizar voz activa y no voz pasiva

El lenguaje controlado tiene las siguientes ventajas[34]:

a) La información presentada sólo se puede interpretar de una manera con lo cual se reduce la ambigüedad y la facilidad de lectura

b) Se impone el uso de un estilo y una terminología consistente

Revista Avances en Sistemas e Informática, Vol.8 No.2, julio de 2011 - Medellín. ISSN 1657-7663

c) Se mejora la facilidad de traducción del texto, especialmente con herramientas de traducción automatizada

d) Se reduce el tiempo para lanzar el producto de software al mercado puesto que el proceso de traducción se hace un 25% más rápido

6) **Internacionalizar el Código XML:** La W3C ha desarrollado una guía de buenas prácticas para la internacionalización de documentos XML[35]. Dicha guía proporciona un conjunto de lineamientos para desarrollar documentos y esquemas XML de manera que queden internacionalizados correctamente. Incluye buenas prácticas tanto para el desarrollador de aplicaciones XML como para el autor de contenido XML con el propósito de crear material en diferentes idiomas.

Se propone considerar el documento incluido en [35] al desarrollar aplicaciones de software internacionalizadas.

7) **Internacionalizar los Servicios Web:** La W3C ha desarrollado un documento que muestra cómo adaptar la mensajería SOAP para proporcionar operaciones internacionalizadas y localizadas usando preferencias locales e internacionales. Dichos mecanismos se pueden utilizar para acomodarse a una amplia variedad de modelos de desarrollo para su uso internacional.

A primera vista, se podría pensar que los servicios web están internacionalizados ya que usan una representación de los datos que generalmente no depende del locale en forma de un esquema XML, sin embargo se requiere que el consumidor y el proveedor de servicios web intercambien preferencias internacionales y acuerden un locale común cuando el servicio web ofrezca funcionalidades dependientes del locale.

Es por ello que se propone considerar el documento incluido en [36] al desarrollar aplicaciones de software internacionalizadas.

5.4. Pruebas de Software

Las pruebas consisten en una verificación dinámica del comportamiento de un programa en un conjunto finito de casos seleccionados apropiadamente de un dominio de escenarios usualmente infinito respecto al comportamiento esperado[1].

5.4.1 Recomendaciones de Internacionalización

1) **Planear y realizar pruebas de internacionalización:** El área de aseguramiento de calidad (QA) debe garantizar que se incluyan pruebas que permitan determinar si el software ha quedado correctamente internacionalizado manteniendo la funcionalidad original y la correcta presentación al usuario final.

Puesto que la internacionalización modifica el diseño, y la codificación del software, es necesario comprobar que esos cambios no afecten la calidad del software y se garantice que el software internacionalizado sea adecuado para ejecutar el proceso de localización en fases posteriores.

Para cumplir con esta recomendación se deben tomar en cuenta las siguientes consideraciones [9][37][38]:

1. Analizar si se afecta el desempeño de la aplicación una vez esté internacionalizada
2. Definir las características y components del software que deben ser probados
3. Definir las características y componentes del software que no se requieren probar
4. Establecer una metodología para abordar las pruebas en un proyecto de internacionalización
5. Establecer entregables, cronograma y recursos para las pruebas

VI. CONCLUSIONES

Es importante resaltar que la internacionalización de software no se reduce a externalizar los recursos de texto incluidos en las interfaces y en la documentación para traducirlos posteriormente y, mediante la inserción de dichas traducciones, ofrecer versiones "localizadas" del software. Por el contrario, las actividades que se deben realizar para garantizar la correcta internacionalización de una aplicación de software abarcan todas las fases de un proceso de desarrollo de software.

Por esta razón es tan importante que las empresas que desarrollan software empiecen a capacitar al personal que ejecuta sus proyectos en temas de globalización de software ya que, en el entorno de la industria del software, una compañía que desee mantener sus operaciones a mediano y largo plazo debe considerar la internacionalización de software no sólo como una oportunidad sino como una necesidad impuesta por las condiciones actuales de competitividad y globalización.

REFERENCIAS

[1] P. Bourque and R. Dupuis, "Guide to the Software Engineering Body of Knowledge 2004 Version," *Guide to the Software Engineering Body of Knowledge, 2004. SWEBOK*, 2004.

[2] M. Sasikumar, R. Aparna, K. Naveen, and M. Rajendra Prasad, "FOSS Guide to Localization." UNDP-APDIP, CDAC Mumbai, 2005.

[3] A. Arora and A. Gambardella, "The Globalization of the Software Industry: Perspectives and Opportunities for Developed and Developing Countries." NBER Working Paper Series.

[4] J. Daunt, "Legal Issues," presented at the Proceedings of rhe Globalizing Software Product Management Conferenc, San Jose, CA.

[5] B. Esselink, *A Practical Guide to Localization*. John Benjamins Publishing Company, 2000.

[6] A. Lommel and R. Ray, "The Globalization Industry Primer." The Localization Industry Standards Association - LISA, 2007.
[7] R. Lommel and D. Fry, "The Localization Industry Primer." Localization Industry Standards Association - LISA, 2003.
[8] A. Asnes, "Business Perspectives on Internationalization," *ClientSide News Magazine; ClientSide Publications*, 2003.
[9] A. Vine, "Internationalizing Software Testing." [Online]. Available: http://developers.sun.com/dev/gadc/des_dev/Intl_Testing.html. [Accessed: 01-Jul-2011].
[10] G. Chroust, "Software Like a Courteous Butler – Issues of Localization under Cultural Diversity," 31-Jul-2007. [Online]. Available: http://journals.isss.org/index.php/proceedings51st/article/view/569. [Accessed: 01-Jul-2011].
[11] Sun Microsystems, "I18n in Software Design, Architecture and Implementation." [Online]. Available: http://dsc.sun.com/dev/gadc/technicalpublications/articles/archi18n.html. [Accessed: 01-Jul-2011].
[12] Kenneth Keniston, "Software Localization," *SOFTWARE LOCALIZATION: NOTES ON TECHNOLOGY AND CULTURE*, 1997. [Online]. Available: http://stuff.mit.edu/people/kken/papers1/Software%20localization.htm. [Accessed: 01-Jul-2011].
[13] Localization Industry Standards Association - LISA, "Results of MIIS/LISA Global Business Practices Survey." Globalization Insider, 2006.
[14] John Papaioannou, "The Localization Outsourcing Decision," Degree of master of business administration, University of Warwick, 2002.
[15] G. Kotonya and I. Sommerville, *Requirements engineering: processes and techniques*. J. Wiley, 1998.
[16] S. Jantunen, K. Smolander, and D. C. Gause, "How Internationalization of a Product Changes Requirements Engineering Activities: An Exploratory Study," in *Requirements Engineering, IEEE International Conference on*, Los Alamitos, CA, USA, 2007, vol. 0, pp. 163-172.
[17] D. A. DePalma, "Translation is the Law," 2004. [Online]. Available: http://www.multilingual.com/articleDetail.php?id=1156. [Accessed: 01-Jul-2011].
[18] N. Kano and M. Press, *Developing International Software for Windows 95 and Windows NT*. Microsoft Press, 1995.
[19]"IEEE Standard for Information Technology-Systems Design-Software Design Descriptions." 2009.
[20] N. L. Hoft, *International Technical Communication: How to Export Information about High Technology*. Wiley, 1995.
[21] J. K. Korpela, *Unicode explained*. O'Reilly Media, Inc., 2006.
[22] T. U. Consortium, *Unicode Standard, Version 5.0, The*, 5th ed. Addison-Wesley Professional, 2006.
[23] N. Singh and A. Pereira, *The Culturally Customized Web Site: Customizing Web Sites for the Global Marketplace*. Butterworth-Heinemann, 2005.
[24] E. Castro, *HTML con XHTML y CSS (Diseño y Creatividad): Todo el Código para Crear sitios Web Efectivos y Originales*, 1st ed. Madrid: Ediciones Anaya Multimedia, 2003.
[25] L. Bass, P. Clements, and R. Kazman, *Software Architecture in Practice*. Addison-Wesley Professional, 2003.
[26] T. Welzer, I. Golob, M. Druzovec, and A. Kamisalic, "Internationalization as a part of the database development," in *IEEE 3rd International Conference on Computational Cybernetics, 2005. ICCC 2005.*, Mauritius, pp. 145-148.
[27] A. Gut, L. Miclea, S. Enyedi, M. Abrudean, and I. Hoka, "Database Globalization in Enterprise Applications," in *2006 IEEE International Conference on Automation, Quality and Testing, Robotics*, Cluj-Napoca, 2006, pp. 356-359.
[28] B. Haara, "Challenging the way we learn to write for a global audience," in *IPCC 98. Contemporary Renaissance: Changing the Way we Communicate. Proceedings 1998 IEEE International Professional Communication Conference (Cat. No.98CH36332)*, Quebec City, Que., Canada, pp. 293-303.
[29] M. Flammia, "Preparing Technical Communication Students to Play a Role on the Translation Team Tutorial," *Ieee Transactions On Professional Communication*, vol. 48, no. 4, pp. 401-412, 2005.
[30] Olcer, "How to Prepare a Localization Kit." School of Translation and Interpretation (ETI). University of Geneva.
[31] L. Muzii, "Building a Localization Kit," *Building a Localization Kit*, 2011. [Online]. Available: http://www.slideshare.net/muzii/building-a-localization-kit-7874044. [Accessed: 01-Jul-2011].
[32] U. Muegge, "Controlled language: the next big thing in translation?," *ClientSide News Magazine; ClientSide Publications*, vol. 7, no. 7, 2007.
[33] E. Torrejon and C. Rico, "Controlled Languages in the Localisation Industry," in *Localization Reader 2004-2005*, Limerick, Ire.: Localisation Research Centre, 2004, pp. 53-56.
[34] U. Reuter and A. Schmidt-Wigger, "Designing a Multi-Purpose CL Application," presented at the Controlled Language Application Workshop (CLAW), 2000, pp. 72-82.
[35]"Extensible Markup Language (XML) 1.0 (Fifth Edition)." [Online]. Available: http://www.w3.org/TR/xml/. [Accessed: 01-Jul-2011].
[36]"Web Services Internationalization (WS-I18N)." [Online]. Available: http://www.w3.org/TR/ws-i18n/. [Accessed: 01-Jul-2011].
[37] M. L. Hutcheson, *Software Testing Fundamentals: Methods and Metrics*, 1st ed. Wiley, 2003.
[38] R. Patton, *Software Testing*, 2nd ed. Sams, 2005.

Universidad Nacional de Colombia Sede Medellín
Facultad de Minas

Escuela de Ingeniería de Sistemas

Pregrado

- Ingeniería de Sistemas e Informática.

Posgrado

- Doctorado en Ingeniería-Sistemas.
- Maestría en Ingeniería de Sistemas.
- Especialización en Sistemas con énfasis en:
  - Ingeniería de Software.
  - Investigación de Operaciones.
  - Inteligencia Artificial.
- Especialización en Mercados de Energía.

Áreas de Investigación

- Ingeniería de Software.
- Investigación de Operaciones.
- Inteligencia Artificial.

Escuela de Ingeniería de Sistemas
Dirección Postal:
Carrera 80 No. 65 - 223 Bloque M8A
Facultad de Minas. Medellín - Colombia
Tel: (574) 4255350 Fax: (574) 4255365
Email: esistema@unalmed.edu.co
http://pisis.unalmed.edu.co/