Versión *preprint* de una comunicación presentada en el Congreso Internacional AIETI 7
Nuevos horizontes en los Estudios de Traducción e Interpretación.
Málaga: enero de 2015

Localización de aplicaciones
Estrategias de cooperación con los desarrolladores

Christina Lachat-Leal
Universidad de Granada
clachat@ugr.es

Resumen: La internacionalización de una aplicación, requisito indispensable para ganar cuotas de mercado y generar rentas en un mercado internacional tan competitivo con una oferta tan amplia, está supeditada a una localización de calidad. Las grandes empresas del sector, conscientes de ello, han elaborado guías con recomendaciones para planificar desde la primera fase de desarrollo la internacionalización y localización de aplicaciones. En esta comunicación, proponemos la elaboración de un protocolo de actuación con el cliente/desarrollador aplicando estrategias de marketing relacional con el fin de normalizar el proceso previo a la localización. Para ello, hemos creado y diseñado unas guías y listas de comprobación personalizadas que agilizan, sistematizan y simplifican la colaboración entre el desarrollador y el traductor lo que deriva en la disminución de errores y en un aumento de la calidad de la localización, reduciendo plazos de entrega y costes.
http://www.tradulex.com/varia/AIETI7.pdf

Palabras claves: localización, Android, iOS, marketing relacional, co-makership

VERSIÓN *PREPINT*: 02/03/ 2015

1. INTRODUCCIÓN

El negocio de las aplicaciones móviles facturó en 2013 más de 19 300 millones de euros según los datos del último Mobile World Congress. Se descargaron 3,8 millones de aplicaciones al día en España (The App Date, 2014) y más de 102 000 millones en todo el mundo (Gartner, 2013). Es un negocio rentable que, sólo en España, se disputan unas 200 empresas de desarrollo. La internacionalización de una aplicación, requisito indispensable para ganar cuotas de mercado y generar rentas en un mercado internacional tan competitivo con una oferta tan amplia, está supeditada a una localización de calidad. Por localización se entiende "el proceso de adaptación de una aplicación o programa informático a una zona geográfica concreta" (Localization Industry Standards Association, 2010).

Uno de los aspectos claves de la localización es el impacto del trabajo de los desarrolladores sobre el de los traductores. En la localización de aplicaciones móviles no existe un texto origen propiamente dicho, sólo cadenas de texto insertadas en un código. Estas cadenas (desde una palabra hasta un párrafo) no tienen formato, ni contexto (Fig. 3, 4 y 5). Toda la información contextual está contenida en el código, distribuida en varios archivos. Lo que podríamos considerar el "texto completo" sólo es accesible mediante la visualización de la interfaz de usuario. El traductor no suele tener acceso a esta interfaz mientras traduce así que no cabe duda de que la colaboración del desarrollador es vital y absolutamente necesaria para asegurar una traducción de calidad reduciendo errores, pruebas, tiempo y costes. El objetivo de este trabajo es estudiar el

modo de facilitar esta colaboración para que el desarrollador pueda guiar al traductor con todos los datos contextuales relevantes.

2. MARKETING RELACIONAL Y CO-MAKERSHIP

Sostenemos que la relación entre el desarrollador y el traductor no debe ser meramente transaccional sino que debe cimentarse en la cooperación, aplicando los principios del marketing relacional y del *co-makership*.

Según los principios que rigen el marketing relacional, la calidad debe ser una preocupación de todos para lograr beneficios mutuos y por tanto debe existir un alto nivel de compromiso y de contactos con el cliente. Es necesario crear, desarrollar y mejorar las relaciones con los clientes y otros actores que participan de la relación (Boone y Kurtz, 2011, p.310). En el *co-makership* la calidad se construye en la etapa de diseño. El cliente y el proveedor trabajan juntos para elevar la calidad y en ocasiones pactan los estándares que utilizarán en los controles de calidad lo que implica un alto nivel de cooperación por ambas partes en las etapas de diseño, desarrollo y producción (Christopher et al., 1994, p. 38).

El traductor, al ser el proveedor, deberá por tanto asumir la responsabilidad de establecer cauces de cooperación ágiles y efectivos con el cliente. Pero antes de diseñar una estrategia efectiva de colaboración con los desarrolladores debemos conocer, aunque sea sucintamente, el proceso de creación de una aplicación móvil y los problemas de localización.

3. APLICACIONES MÓVILES

En esencia, una aplicación móvil (*app*) es un programa informático diseñado para educar, entretener o asistir a un usuario que se ejecuta en teléfonos inteligentes, conocidos popularmente como *smartphones*, tabletas u otros dispositivos móviles.

3.1. Proceso de creación

El proceso de diseño y desarrollo de una aplicación abarca desde la concepción de la idea hasta el análisis posterior a su publicación en las tiendas. Durante las diferentes etapas, diseñadores y desarrolladores suelen trabajar de manera simultánea y coordinada (Cuello and Vittone, 2013, p.17).

3.1.1. Conceptualización y definición

En esta primera etapa, se realiza una investigación para determinar las necesidades del mercado y de los usuarios. Tras este primer proceso, se define la aplicación, se estudia su viabilidad y se determina su funcionalidad. En esta fase se describen con detalles los perfiles de usuarios (Cuello y Vittone, 2013, p.18).

3.1.2. Diseño

En esta etapa, se crean los primeros prototipos en forma de *wireframe* o plano de plantilla. Estas representaciones muy simplificadas de una pantalla de un teléfono o de un reloj inteligente (Fig. 2) son una herramienta muy útil para el diseño de interfaces. El diseñador también se encarga de seleccionar las fuentes y redactar los textos. Cuando el diseño visual está acabado se entrega al desarrollador en forma de archivos separados para la programación del código (Cuello y Vittone, 2013, p.19).

Figura 1. Etapas de desarrollo de una aplicación

3.1.3. Desarrollo

En esta etapa, se realiza la programación del código. Primero, el desarrollador debe determinar qué tipo de aplicación quiere desarrollar, nativa, web o híbrida, ya que de ello dependerá el código.

Las aplicaciones nativas se desarrollan con el programa que ofrece cada sistema operativo (SO) a los programadores, llamado genéricamente *Software Development Kit* (SDK) y en el lenguaje nativo utilizado por el SDK (Tabla 1).

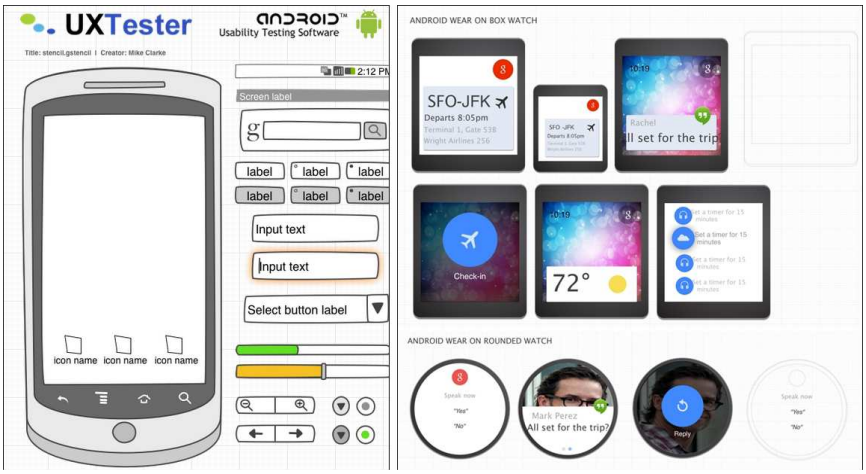

Figura 2. *Wireframe*. Izda.: IU móvil Android;
dcha.: IU smartwatch Android Wear

Las aplicaciones web, conocidas también como *web apps*, se desarrollan utilizando los lenguajes de programación HTML, JavaScript y CSS lo que permite programar de forma independiente el sistema operativo en el cual se usará la aplicación. Por último, las aplicaciones híbridas combinan el desarrollo nativo con tecnología

multiplataforma HTML, JavaScript y CSS. Al igual que en las aplicaciones nativas, el código, una vez creado, se compila en un ejecutable.

En esta etapa se incluye la fase de prueba, una de las más importantes del proceso de desarrollo ya que trata de asegurar la funcionalidad y calidad de la aplicación evaluando su ejecución y resultado, corrigiendo errores (*bugs*).

	Apple iOS	**Android**	**Windows Phone**
Lenguajes	Objective-C, C, C++	Java (C, C++)	C#, VB.NET, etc.
Herramientas	Xcode	Android SDK	Visual Studio, Windows Phone
Formato	.app	.apk	.xap

Tabla 1. Principales SO móviles (IBM Software, 2012, p.3)

3.1.4. Internacionalización

El resultado de la internacionalización "es un código genérico que se puede utilizar con cualquier configuración regional y que podrá aceptar, procesar y mostrar correctamente cualquier formato de dato e idioma" (Juárez Barrena, 2009, p.7). Tradicionalmente, la fase de internacionalización se iniciaba tras la compilación de los archivos por el programador en el idioma original. Actualmente, esta fase de integra en el proceso de diseño y desarrollo de la aplicación ya que con ello se reducen costes y se agiliza el lanzamiento al mercado internacional.

3.2. Diseñar para la localización

A continuación expondremos de forma resumida los principales problemas de la localización de aplicaciones que pueden evitarse o solucionarse en la etapa de diseño de la interfaz del usuario (IU) y en la fase de programación.

3.2.1. Interfaz

Uno de los aspectos más problemáticos y de fácil resolución, si el diseñador es consciente de ello en la etapa de diseño, es que según el idioma, el tamaño de elementos como cuadros de diálogo puede incrementar bastante lo que produce cambios en la interfaz de usuario. Es por lo tanto importante que el equipo de diseño tenga en cuenta el posible redimensionamiento de la interfaz. Una posible solución es la asignación de búferes dinámicos (Windows, 2014). Otra solución para afrontar este problema y evitar solapamientos consiste en crear cuadros de diálogo con el máximo de espacio posible de reserva, desde un mínimo del 30% (Android, n.d.) hasta un 200% si el idioma fuente es el inglés (Tabla 2).

Lo que debe evitarse es el uso de abreviaturas porque tal como comprobamos en la tabla 2, cuantos menos caracteres tiene el texto original, mayor es la probabilidad de expansión del texto traducido. Además no todas las abreviaturas tienen correspondencia en otros idiomas.

3.2.2. Iconos y mapas de bits

Los iconos y mapas de bits no suelen tener necesidad de texto pero en el caso de que las imágenes contengan texto, el diseñador debe evitar los mapas de bits, difíciles de manipular, y usar gráficos vectoriales (Mata Pastor, 2009, p.519)

Nº de caracteres texto inglés	**Espacio adicional necesario**
<10	del 100% al 200%
11 a 20	del 80% al 100%
21 a 30	del 60% al 80%
31 a 50	del 40% al 60%
51 a 70	del 31% al 40%
>70	el 30%

Tabla 2. Expansión de la IU (IBM, 2014)

3.2.3. Cadenas de texto

Las cadenas de texto no están contextualizadas lo que suele producir errores de traducción. Para evitarlos, los programadores pueden introducir información mediante comentarios que aclaran la función, forma y dimensión de la cadena de texto (Fig.3).

```
<!-- The action for submitting a form. This text is on a button that can fit 30 chars -->
<string name="login_submit_button">Sign in</string>
```

Figura 3. Ejemplo de comentario (Android, n.d.)

Otros dos aspectos de la codificación que dificultan enormemente el proceso de traducción son la concatenación y las variables dentro de las cadenas. Si el traductor recibe las cadenas de texto si conocer las posibles combinaciones y el orden de la concatenación, o el valor de las variables, el texto localizado podrá producir frases sin sentido o contener incorrecciones lingüísticas. Para evitar estos problemas, el programador puede optar por no utilizar estos dos procesos o por añadir toda la información pertinente en un comentario (Fig. 4).

```
/* Name and height (in meters) of a mountain */
"%@ is %d meters tall" = "%1$@ is %2$d meters tall";
```

Figura 4. Contextualización de una variable de cadena (Hanson and Foley-Fischer, 2014)

Por último, para evitar errores en el funcionamiento de la aplicación es necesario señalar las cadenas que no deben ser modificadas durante la traducción mediante comentario o una etiqueta (Fig.5).

```
<string name="countdown">
    <xliff:g id="time" example="5 days>%1$s</xliff:g>until
holiday
</string
```

Figura 5. Etiqueta <xliff:g> para cadena de texto no traducible (Android, n.d.)

3.2.4. Archivos localizables

Otro aspecto que puede alargar el proceso de localización es la preparación de una aplicación para ser localizada. Es importante que el desarrollador aísle y entregue al traductor todos los archivos de recursos localizables. Una aplicación preparada para ser localizada tendrá un bloque de código y un bloque de datos o archivos de recursos. El bloque de código contendrá exclusivamente el código de la aplicación, válido para todas las culturas o configuraciones regionales. Los archivos de recursos contendrán todos los recursos de cadena de la interfaz de usuario (*strings)* pero también otros recursos como audio, vídeo e incluso imágenes. El desarrollador deberá tener presente que las imágenes no son culturalmente neutras y pueden resultar inadecuadas o incluso ofensivas en otras culturas. Es, pues, importante que el desarrollador identifique todos los *drawable* (imagen objeto) y los remita al traductor (Android, n.d.).

4. INSTRUMENTOS DE COOPERACIÓN TRADUCTOR-DESARROLLADOR

La necesidad de dar a conocer a los desarrolladores la labor de los traductores no es nueva. Numerosas agencias de traducción incluyen guías para sus clientes desarrolladores en sus webs (Icanlocalize, n.d.). Sin embargo, nuestro objetivo es conseguir que los desarrolladores no sólo conozcan la labor de los traductores, sino que cooperen y confíen en ellos. Para que esta colaboración sea factible y efectiva debe reunir una serie de requisitos.

En primer lugar, la información proporcionada a los desarrolladores debe ser relevante y precisa. Segundo, los instrumentos de comunicación elegidos deben ser ágiles, flexibles, personalizables y fácilmente integrables el proceso de diseño de la aplicación. Las guías para desarrolladores elaboradas por las grandes empresas del sector del software, IBM (2014), Microsoft (Windows, 2014), Apple (Apple Developer, 2014), Google (Android, n.d.) son un gran punto de partida pero deben adaptarse a la idiosincrasia del cliente y del propio traductor. Entre otros aspectos se debe ofrecer en la lengua del cliente. Nos parece pertinente la entrega de guías personalizadas a un cliente nuevo junto con el presupuesto.

Esta guía personalizada resultará muy útil como herramienta de consulta en caso de problema, pero no se puede integrar en el proceso de diseño y por tanto no servirá para anticiparse a los problemas y evitar pérdidas de tiempo. Necesitamos encontrar un instrumento lo suficientemente ágil para que el desarrollador pueda utilizarlo mientras trabaja.

Tras estudiar varias herramientas como folletos, infografías, presentaciones, vídeos y listas de comprobación, hemos seleccionado estas últimas como el instrumento más adecuado. Las listas nos recuerdan los pasos mínimos necesarios y los hace explícitos. Son, por encima de todo, prácticas, y no intentan explicarlo todo, sino que son un recordatorio de los pasos críticos (Gawande, 2011, p.112). Pueden diferir en cuanto a formato o diseño pero todas tienen el mismo objetivo: por un lado, reducir

errores que pueden acarrear costes y, por otro, mejorar el resultado general (Hales y Pronovost, 2006: 231-232). Otro factor que nos ha decidido a adoptarla ha sido que los propios desarrolladores de software las suelen utilizar (Gawande, 2011, p.39; Esselink, 2000).

Las buenas listas deben ser precisas, eficientes y fáciles de cumplimentar y por ello, hemos seguido unas pautas para su confección:

- Breve (entre 5 y nueve apartados)
- Redacción sencilla y precisa con terminología de la profesión
- Buena tipografía (Gawande, 2011, pp.114–115)
- Específica para cada etapa: diseño y programación
- Específica para un sistema operativo

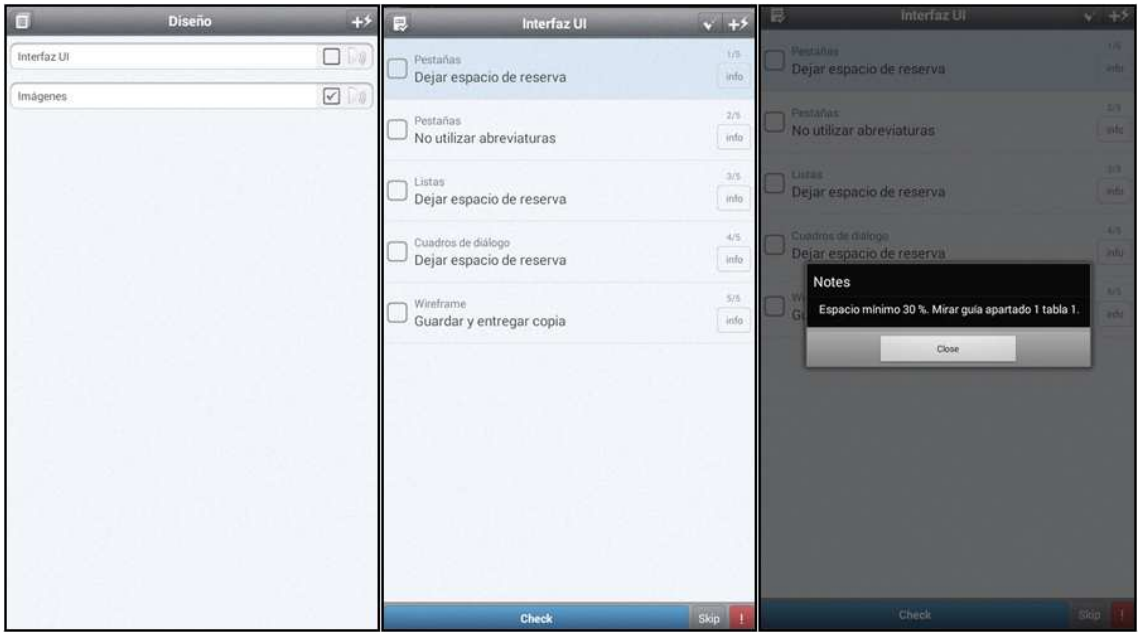

Figura 6. Listas de comprobación para diseñador

En cuanto, al formato hemos optado por el soporte electrónico. Tras probar varias opciones, nos hemos decantado por la aplicación móvil gratuita *Checklist* (Yeno, 2010). Esta aplicación se distingue por:

- Interfaz simple
- Tipografía clara
- Posibilidad de agrupar listas
- Edición de listas en la aplicación y/o en PC
- Inserción de notas informativas
- Listas exportables

El resultado ha sido el diseño y creación de seis listas de comprobación y de dos guías: la *Guía del diseñador* y la *Guía del programador*. Para la etapa de diseño, hemos confeccionado dos listas, una dedicada a la interfaz y otra a las imágenes (Fig.6). Para la etapa de programación, hemos creado dos categorías de listas, una dedicada a las cadenas de texto y otra para la compilación de archivos localizables. Para la categoría de compilación de archivos localizables hemos elaborado unas listas específicas para tres SO (Fig.7). En el futuro se podrán incorporar otros SO, según necesidades.

Estas listas recogen todos los elementos problemáticos descritos anteriormente (§3.2). Cada elemento problemático viene acompañado de una breve instrucción a modo de recordatorio. En caso necesario, el desarrollador puede ampliar información pulsando el botón *info* que despliega una nota. Si, por ejemplo, el diseñador necesita saber cuánto espacio de reserva debe prever, la nota informativa le recuerda el espacio mínimo y le remite a una tabla de la *Guía del diseñador* (Fig.6).

Figura 7. Listas de comprobación para el programador

5. CONCLUSIONES

El cliente y el proveedor que trabajan juntos elevan la calidad del producto que se construye en la etapa de diseño. La colaboración entre traductor y desarrollador puede evitar errores que alargan el proceso de localización, elevan los costes y repercuten en la calidad de las aplicaciones. El traductor, al ser el proveedor, debe asumir la responsabilidad de establecer cauces de cooperación ágiles y efectivos con el desarrollador. Para conseguir que éste coopere y confíe en el traductor, los instrumentos de comunicación elegidos deben ser ágiles, flexibles, personalizables y fácilmente integrables el proceso de diseño de la aplicación. No deben entorpecer la labor de los desarrolladores, ni suponerles una inversión de tiempo y la información proporcionada a los desarrolladores debe ser relevante y precisa.

Las listas de comprobación acompañadas de guías personalizadas para diseñadores y programadores son el instrumento más eficaz para prevenir o resolver problemas de localización durante el proceso de creación de una aplicación. Unas listas bien diseñadas y efectivas agilizan, sistematizan y simplifican la colaboración entre el desarrollador y el traductor cuya consecuencia directa es la disminución de errores y un aumento de la calidad de la localización, reduciendo plazos de entrega y costes. Indirectamente, el traductor transmite una imagen de alta cualificación profesional a su cliente y asegura su fidelización.

BIBLIOGRAFÍA

ANDROID, n.d. *Localization Checklist | Android Developers*. [online] Disponible en: <http://developer.android.com/distribute/tools/localization-checklist.html>.

APPLE, 2014. *Build apps for the World*. [online] Disponible en: <https://developer.apple.com/internationalization/#internationalize> [25 Nov. 2014].

APPLE DEVELOPER, 2014. *Internationalization and Localization Guide*. [online] Disponible en: <https://developer.apple.com/library/ios/documentation/MacOSX/Conceptual/BPInternational/BPInternational.pdf>.

BOONE, L. y KURTZ, D., 2011. *Contemporary Marketing*. Boston: Cengage Learning.

CHRISTOPHER, M., PAYNE, A. y BALLANTYNE, D., 1994. *Marketing relacional: integrando la calidad, el servicio al cliente y el marketing*. Madrid: Ediciones Díaz de Santos.

CUELLO, J. y VITTONE, J., 2013. *Diseñando apps para móviles*. Catalina Duque Giraldo.

ESSELINK, B., 2000. *Practical Guide to Localization.* Amsterdam: John Benjamins.

GAWANDE, A., 2011. *El efecto Checklist. Cómo una simple lista de comprobación elimina errores y salva vidas*. Traducido del inglés por F. Corriente Basús. Barcelona: Antonio Bosch.

HANSON, C. y FOLEY-FISCHER, Z., 2014. *Localizing with Xcode 6. Best practices and new workflow*. [online] Disponible en: <http://devstreaming.apple.com/videos/wwdc/2014/412xx80au1lrfcn/412/412_localizing_with_xcode_6.pdf?dl=1.> [28 Nov. 2014].

IBM, 2014. *Guideline A: User interface*. [online] Disponible en: <http://www-01.ibm.com/software/globalization/guidelines/a3.html> [29 Nov. 2014].

IBM SOFTWARE, 2012. *El desarrollo de aplicaciones móviles nativas, Web o híbridas*. [online] Disponible en: <ftp://ftp.software.ibm.com/la/documents/gb/commons/27754_IBM_WP_Native_Web_or_hybrid_2846853.pdf>[29 Nov. 2014].

ICANLOCALIZE, n.d. Guía para la Localización de Aplicaciones iPhone [online] Disponible en: http://www.icanlocalize.com/site/es/tutoriales/guia-para-la-localizacion-de-aplicaciones-iphone/ [29 Nov. 2014].

JUÁREZ BARRENA, E., 2009. *Globalización de software*. [online] UPM. Disponible en: <http://oa.upm.es/1802/>[2 Nov. 2014].

LOCALIZATION INDUSTRY STANDARDS ASSOCIATION, 2010. *Glossary. Localization*. [online] Disponible en: <http://web.archive.org/web/20100315061742/http://www.lisa.org/Glossary.108.0.html?tid=1> [17 Feb. 2015].

MATA PASTOR, M., 2009. Algunas pautas para el tratamiento de imágenes y contenido gráfico en proyectos de localización (I). *Entreculturas*, 1, pp.513–532.

WINDOWS, 2014. *Directrices para globalización y localización*. [online] Disponible en: <http://msdn.microsoft.com/>[29 Nov. 2014].

YENO, 2010. Checklist. [online] Disponible en: http://www.yeno.eu/ [17 Feb. 2015].