Accesibilidad en Linux

TIFLOTECNOLOGÍA

Prometíamos al término el artículo anterior «examinar los sistemas alternativos de comunicación con el ordenador (morse, braille), la disponibilidad o no de aplicaciones destinadas a los alumnos con discapacidades, la accesibilidad en la web, y finalmente un cambio de enfoque para examinar todas estas herramientas y especificaciones desde el punto de vista pedagógico». Es el momento de cumplir en la medida de lo posible lo prometido. **POR JUAN RAFAEL FERNÁNDEZ GARCÍA**

Habiendo consultado a varios colegas sobre la entrega anterior de este artículo, me señalaron que había una imprecisión en el vocabulario y la necesidad de reconducir la presente entrega en respuestas hacia preguntas formuladas por enseñantes, sin entrar en clasificaciones de personas según sus discapacidades.
Proseguimos nuestro viaje, pues, hablando de la tiflotecnología.

Braille

¿Tiene sentido seguir hablando de Braille cuando ya hemos conocido las posibilidades que ofrecen los lectores de pantalla? Un mensaje a la lista de distribución tiflonet[1] nos da la respuesta: es el último vínculo con la lectura y la escritura de los invidentes. En el lenguaje hablado no hay ortografía, es más las palabras se confunden en grupos fónicos regidos por el ritmo y la entonación.

Como sabemos, el código Braille consiste generalmente en celdas de seis puntos en relieve, organizados como una matriz de tres filas por dos columnas [2] ¿Cómo se utiliza en informática? Mediante hardware específico, y para evitar confusiones debemos empezar por una breve clasificación: debemos distinguir dispositivos de entrada (anotadores braille, procedentes de las máquinas Perkins, y que están siendo, por su precio, sustituidos por PDAs) de los dispositivos de salida (terminales o líneas braille e impresoras braille)[3].

La entrada no presenta dificultades: pueden utilizarse teclados especiales braille, o bien adaptaciones de los teclados ordinarios para escribir Braille. O el teclado completo y un programa conversor (en la lista de distribución de blinux y sin buscar mucho he encontrado *nfbtrans*, *Turbo Braille*, *Duxbury for*

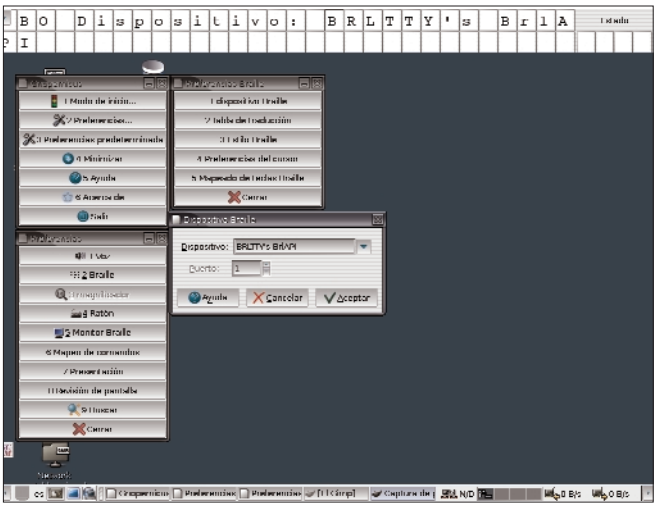

Figura 1: Configuración de una terminal braille con gnopernicus.

Linux). Las dificultades están en la salida. Una primera solución ya la vimos en el artículo anterior: los lectores de pantalla. La alternativa son las llamadas «líneas braille». Son dispositivos que se conectan por el puerto serie o usb al ordenador, basados en un mecanismo electro-mecánico capaz de representar caracteres Braille mediante la elevación de puntos a través de una superficie plana con agujeros hechos a tal efecto (cada célula contiene ocho pequeñas agujas, por lo que la representación se realiza mediante Braille de ocho puntos). Complejos y de elevado coste, suelen tener 20, 40 u 80 celdas; también dispone de botones para desplazar el texto y realizar otras funciones especiales.

El uso de terminales braille bajo GNU Linux pasa a bajo nivel por *brltty*[4], aunque puedan utilizarse interfaces como *gnopernicus* (imagen 1). La documentación muestra un amplio listado de líneas braille contempladas. ¿Cómo funciona? *Brltty* reproduce una porción rectangular de la pantalla del ordenador como texto Braille en el dispositivo, y permite utilizar los controles del dispositivo para moverse por la ventana reproducida.

Brltty solamente funciona con consolas y aplicaciones basadas en texto. ¿Cómo lograr reproducir imágenes? Una respuesta es imprimirlas en relieve, lo veremos ahora; pero en una terminal con refresco es posible también utilizar un desarrollo proporcionado por *See by Touch*[5]. SeebyTouch proporciona la posibilidad de percibir imágenes mediante el sentido táctil. Presentado como proyecto de fuente abierta en marzo de 2004, consiste en una unidad movible por el usuario y controlada de forma paralela con dos barras (foto 2). Esta unidad contiene celdas braille que hacen la función de matriz táctil. Dependiendo de su posición el usuario percibe mediante el tacto partes de la imagen; moviendo la unidad el usuario puede identificar distintas partes de la foto o dibujo y hacerse una composición mental de la imagen.

Figura 2: SeebyTouch para *ver con el tacto*.

Existen también las impresoras braille[6], que imprimen en relieve (por eso se llaman «embossers» en inglés). Se configuran en Linux como impresoras de agujas (cf. https://www.redhat.com/archives/blinux-list/2005-January/msg00053.html). Y una tecnología llamada «horno fúser» que permite la plasmación en soportes especiales de dibujos de línea resaltada.

Morse

¿Por qué hablar aquí del código Morse? Porque es uno de los métodos de adaptación de la interrelación con el ordenador que se han demostrado eficaces para ayudar a personas con necesidades específicas (figura 4). Y porque nos va a servir para mostrar una de las paradojas presentes en el mundo del software libre.

¿En qué medida es práctico y útil el uso de morse? En el artículo *Morse Code Demystified: a Powerful Alternative for Access to AAC and Computers*[7], de Bruce Fleming et al., se señalan las ventajas del Morse frente a otras formas adaptadas de interacción con el ordenador (vid. la tabla 1). Al hablar de morse como herramienta de comunicación en atención a la diversidad debemos romper varios prejuicios:

- Se cree que se tarda mucho en aprender (en realidad sólo hacen falta de dos a cuatro horas)
- Se dice que es pesado y difícil de usar (¿recuerdan lo que es aprender a escribir a máquina sin mirar?)
- Se piensa que es una mecanismo lento (en realidad son normales velocidades de 10 a 25 palabras por minuto, más rápido que muchos escáners; hay usuarios que llegan a superar las 30 palabras por minuto).

Ahora cambiamos de idioma: la comunicación mediante código Morse se llama «onda continua» (continuous wave), de ahí los «cw» presentes en los nombres de las herramientas disponibles. Y es que en GNU Linux como de costumbre el tema del Morse está en manos de *hackers*, en este caso de los peores, los radioaficionados (adviértase el tono irónico, por favor). Lo que hace que sea posible utilizar Morse con las más variadas y complejas tecnologías, pero que no haya una interfaz atractiva para niños

Tabla 1:Ventajas y desventajas del Morse como herramienta de comunicación

Ventajas	Desventajas
Como método directo de selección es eficaz	Con un ratón no es intuitivo
A medida que se coge soltura, el proceso se vuelve automático	Obliga al estudio previo del código
Requiere muy poco esfuerzo físico	Se percibe como obsoleto o pasado de moda
Es fácilmente adaptable a la diversidad de los usuarios	Prejuicio generalizado de medio difícil de aprender
Puede utilizarse desde una diversas posiciones	El éxito de la comunicación depende al 100% depende al 100% de la entrada
Requiere poco procesamiento visual de la información	
Puede combinarse con otras adaptaciones	
Es una excelente opción como medio de entrada de texto	

Listado 1: El juego *morse*

```
01 $ morse casa
02  daw dit daw dit
03  dit daw
04  dit dit dit
05  dit daw
06
07  dit dit dit daw dit daw
08
09 $ morse -s casa
10  -.-.
11  .-
12  ...
13  .-
14
15  ...-.-
16
17 $ morse -d
18 -.-.
19 C
20 .-
21 A
22 ...
23 S
24 .-
25 A
```

similar a la del programa *Preescritura Morse*, donde por ejemplo el punto son pelotas y la raya son coches y el niño juega hasta aprender a reconocer y a introducir los caracteres como si tirara al plato. Tampoco he encontrado nada similar en los listados de actividades para (j)clic, atnag, childsplay o gcompris.

La primera opción disponible en software libre para aprender Morse es utilizar el juego (entiéndase juego como «juego para hackers» *morse*, de bsdgames. No es más que un conversor, pero es útil si el objetivo es aprender. En el cuadro 1 tenemos un ejemplo aclaratorio del funcionamiento de *morse*.

La interfaz más amigable, créanme, es *xcwcp*, un programa tutor interactivo de código Morse para Xwindows (figura 3). Nos permite elegir varias modalidades de práctica, incluidos el envío de caracteres aleatorios, palabras al azar o caracteres introducidos mediante el teclado. También permite utilizar el teclado o el ratón para enviar Morse (creo que la función se llama a Morse keyer), y permite monitorizar este código y mostrar los caracteres que ve.

Figura 3:xcwcp, programa para aprender Morse.

¿Pero entonces no hay ninguna aplicación Linux que permita utilizar el Morse para manejar el ordenador? Sí la hay: *morseall* (existe paquete rpm). Es una aplicación para gnome que permite controlar una terminal pulsando con el teclado (una breve pulsación del botón izquierdo es el punto, una pulsación más larga o un click con el botón derecho es la raya, el programa presenta una ventana con la equivalencia de los caracteres en puntos y rayas como ayuda al usuario).

La opción preferible (aunque evidentemente más cara) es usar hardware específico. Los fabricantes de Darci Too y Darci USB (recordemos la figura 4) han respondido afirmativamente a mi pregunta de si tenían constancia de su funcionamiento correcto bajo GNU Linux.

Otros Sistemas de Interacción y Comunicación

Es el momento de replantearse el artículo en la forma en que va a ser publicado en la revista. El espacio prohíbe continuar la exploración detallada de los sistemas de interacción con el ordenador. El espacio y el sentido común: el detalle del material encontrado durante la fase de preparación lleva a huir de una aplastante acumulación de tecnologías que requerirá una plasmación web más pausada y nos aleja de nuestros objetivos primeros. Podríamos discutir de la madurez del OCR en GNU Linux y hablar de la existencia de aplicaciones comerciales y de otras libres (*clara*, *gocr*, *ocrad*) y de la utilidad de la tecnología del Reconocimiento Óptico de Caracteres para permitir que las personas con problemas serios de visión puedan acceder a libros e impresos y volcarlos ya como textos en lectores de pantalla o líneas braille, pero deberemos limitarnos a enunciarla.

O podemos hablar de las pantallas táctiles (touch screens), totalmente funcionales en Linux. *http://www.tldp.org/HOWTO/XFree86-Touch-Screen-HOWTO.html*, escrita por Christoph Baumann, nos da los detalles necesarios.

O de las tecnologías de dispositivos hápticos (relativos al tacto) bidimensionales, que pueden utilizarse para ayudar a usuarios con discapacidades visuales proporcionando ligera resistencia en los bordes de ventanas y botones de manera que el usuario «sienta» la interfaz gráfica o pueda de algún modo diferenciar texturas en las imágenes digitales. Esta tecnología ya es una realidad en ratones como el iFeel Mouse y el IFeel Mouseman de Logitech.

O del resto de sistemas aumentativos y alternativos de comunicación, tableros de conceptos…

Sobre la Ausencia de Software

Es verdad que el de las necesidades específicas es un nicho de mercado muy limitado, donde el desarrollo de software está subordinado al más lucrativo de hardware, y donde se dan características propias de los primeros tiempos de esta era de lo digital. Recordemos la explosión de software de pequeñas empresas y de autores independientes durante los años del DOS y Windows 3.1 y los primeros Mac. Multitud de autores crearon y distribuyeron sus pequeñas aplicaciones educativas. La situación es similar, predominan las pequeñas empresas y los desarrollos de uno o dos autores que son profesionales de la atención a la diversidad. El riesgo es que vuelva a caer en la trampa del software gratuito de fuente cerrada: un cambio en los estándares de programación o de interfaz de usuario o en los sistemas operativos utilizados mayoritariamente hacen que ese software quede inutilizable.

Durante la fase de preparación de este artículo he escrito a algunos de los más importantes desarrolladores de lengua hispana de software específico para el tratamiento de las necesidades especiales. Amablemente algunos han contestado y hemos discutido qué impedía el paso de las licencias freeware con código cerrado que tradicionalmente utilizan a software libre. Sin citar nombres sí citaré frases textuales, para analizar brevemente la problemática. Afirman simpatía y estar meditando seriamente

Figura 4: Darci usb, dispositivo de acceso al ordenador mediante morse.Morse.

la posibilidad de dar el paso. ¿Qué problemas ven?

«el primero de estos programas que desarrollé (…) pusimos el código fuente en Internet (…) Lo quitamos porque nos encontramos con un impensado problema, consistente en que hubo personas que comenzaron a tomar el desarrollo, modificarlo sin un análisis con profesionales del área de la educación especial (lo cual era sumamente peligroso para algunos posibles usuarios, por una cuestión de desconocimiento que, aunque no debería existir, es real aquí), y luego proponían venderlo. (…) "comerciantes" que ante un campo con relativamente poco desarrollo como el de la tecnología y las necesidades especiales, sólo quieren ganar dinero sin tener en cuenta la calidad y potenciales riesgos que generan.» Argumento de responsabilidad sobre la obra, totalmente independiente de que sea libre o no.

«(…) el problema de que, ante una registración de la propiedad intelectual sólo en el país (como está hecho ahora por cuestión de costos), personas con fines exclusivamente lucrativos lo registren en otro lugar y lo vendan, cosa que los autores originales no queríamos.» Esto es un problema de protección del copyright, no de licencia. Luego también es independiente de que la obra sea libre o no.

«(…) el desarrollo en formato tenedor. Es decir si dejo, pongamos un ejemplo, el código de P libre y alguien hace un cambio en un sitio y otro alguien en otro, ¿qué es lo que exactamente queda de P? Hasta ahora la solución que he adoptado es que cualquier persona que precisa una modificación me la propone y yo la hago y es una manera de tener controlado el programa». La habitual y comprensible confusión del derecho de reconocimiento de autoría con licencia, permisos que el dueño de la obra concede sobre el uso, copia, distribución de la misma.

«¿Cómo se gana la vida la gente que desarrolla el software libre?» Pues igual, trabajando. Como no existe la killer-application el modelo de negocio se centra en encargos y en servicios, y se trabaja sobre lo que han hecho otros y se sabe que otros trabajarán sobre lo que hacemos nosotros. ¿No era esto el progreso?

Accesibilidad en la Web

La primera en la frente, y en pocas palabras: existe software de creación de páginas web que debería haber dejado de utilizarse hace mucho tiempo ya, si es que en algún momento tuvo una justificación (estoy siendo diplomático), enfoques de la creación de páginas web que no conocen y anulan la separación entre contenido y presentación. Y ahora explicaré porqué.

El estándar *html* no lo ha creado una empresa para hacerse con el mercado. Lo ha creado un consorcio de organizaciones (el W3C, hay oficina española *http://www.w3c.es/*) por impulso de Tim Werners-Lee, el inventor del protocolo http y de la web. Fue creado en octubre de 1994 «para guiar la Web a su máximo potencial mediante el desarrollo de protocolos comunes que promoviesen su evolución y asegurasen su interoperabilidad». La principal tarea del consorcio es, pues, definir estándares (recomendaciones, en sus palabras) que funcionen con cualquier sistema operativo y cualquier navegador. Así a lo largo de los años han ido definiendo nuevas versiones del lenguaje de etiquetado *html* con el que está escrita la WWW y han creado *CSS*, las hojas de estilo en cascada para definir la presentación de las páginas.

Para los que nunca hayan examinado el código de una página web quizás sirva un breve ejemplo: esto sería un fragmento de texto html

```
<p>
Un punto
<strong>importante</strong>.
</p>
```

Hemos visto cómo se marca el comienzo de un párrafo (<p>), su contenido y una etiqueta semántica para señalar la importancia de una expresión. Esto es importante: en esta fase el autor debe preocuparse de la estructura semántica de su documento, que será independiente de su salida impresa. ¿Y la hoja de estilo? Una forma sencilla sería

```
p {
text-indent: 1.8em;
text-align: justify;
color: black;
background-color: white;
font-family: verdana,sans-serif;
}
```

Se definen aquí las propiedades que tendrá el párrafo (todos los párrafos del documento, no será necesario ir copiándolas párrafo a párrafo): sangría, tipo de letra, su color y el de su fondo, justificación…

¿Qué tiene que ver esto con la accesibilidad? Todo: su *Iniciativa de Accesibilidad Web (WAI)* ha publicado las *Directrices de Accesibilidad para el Contenido* (*http://www.w3.org/TR/WCAG/*), que no sólo están dirigidas a atender a las discapacidades sino a independizar el contenido del llamado «agente de usuario». Mediante hojas de estilo es posible definir distintas salidas o presentaciones del contenido (el navegador, sí, pero también un sintetizador de voz, un teléfono móvil y un dispositivo instalado en el coche, etc.). Para ejemplo basta un párrafo.

```
p {
voice-family: paul;
stress: 20;
richness: 90;
cue-before: url("ping.au");
volume: x-soft;
azimuth: center-left;
}
```

Balance Pedagógico

Debemos volver a nuestra pregunta inicial: ¿en qué grado está el software libre maduro para responder a las necesidades de atención a todos nuestros alumnos, tengan las barreras que tengan y las dificultades que tengan? Si hemos dispuesto en esta segunda parte del artículo de cuatro o cinco páginas, debemos llegar a conclusiones aunque sean provisionales y abiertas a la discusión.

En primer lugar podemos concluir que la interfaz de usuario ha llegado a su madurez en cuanto al soporte de la tec-

nología de asistencia. Todo hardware que funciona mediante el uso de controladores estándar o abiertos funcionará con GNU Linux. Y con un enfoque correcto: no hacen falta aplicaciones específicas de teclados virtuales, emulaciones de teclado o ratón, etc., porque es el sistema operativo o el entorno integrado (Gnome, KDE) los que se ocupan de ello; tampoco es una tarea de la que deban ocuparse las aplicaciones específicas, salvo cumplir unas reglas en su desarrollo (lo que equivale a seguir adecuadamente una API y unas instrucciones). Por tanto no hace falta desarrollar juegos específicos para jugar mediante sólo teclado o sólo joystick o sólo switches, sino que cualquier juego debe poder ser jugado mediante el dispositivo que elija el usuario.

Existen distribuciones específicas para por ejemplo personas con discapacidades visuales, caso de Oralux (*http://www.oralux.org/*). Aunque comprendo su utilidad, entiendo que lo que hay que hacer es desarrollar las distribuciones generales para que puedan ser utilizadas por todos, una distribución específica nunca podrá mantenerse al día y ser completa. En este sentido no se ha solucionado el segundo problema, la configuración del soporte de asistencia para cada persona: yo mismo he tenido que bucear en la documentación y en las profundidades del entorno para poder configurar algunas opciones. Sugiero la creación de perfiles que de forma sencilla y autónoma permitan la configuración y la adaptación a las capacidades de cada uno del entorno de trabajo (y de juego y de comunicación y aprendizaje).

En consecuencia deja de ser tan cierto que no existan aplicaciones específicas[8]: en primer lugar porque aquellas aplicaciones que no dependen directamente de la presencia de determinados controladores de hardware pueden funcionar perfectamente con *wine* (figura 5) o *dosemu* o incluso con máquinas virtuales *qemu*, sino porque además las aplicaciones educativas libres «generalistas» pueden ser utilizadas tras una correcta configuración de la interfaz de usuario. Sí es verdad que hay que trabajar en el desarrollo de estos núcleos de aplicaciones, *jclic*, *atnag*, *wims*, *childsplay*, *gcompris*, *leterrier*, para disponer de actividades y juegos de familiarización con el hardware adaptativo por un lado y para proporcionar ejercicios y prácticas de todos los niveles cognitivos y para todas las capacidades. Este es un tema que no podemos pasar por alto, porque es la clave del éxito y el sentido del software educativo, tanto libre como privativo.

Y en el Próximo Número

Examinaremos qué se esconde detrás del proyecto ATNAG, del francés Geard Sellès: una herramienta de autor, destinada a crear juegos y actividades destinados a educación infantil, pero también un conjunto de actividades y un sistema de planificación y seguimiento del trabajo con los alumnos. Veremos en concreto cómo se está utilizando por el equipo de trabajo que lidera María Dolores Almansa en el Colegio Sagrado Corazón de Palencia y nos servirá para ver cómo se aplica la filosofía del software libre en la realidad de las aulas. ■

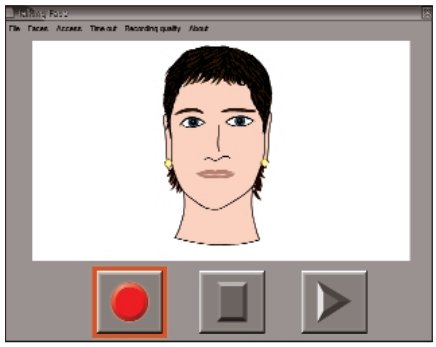

Figura 5: Una aplicación específica reutilizada bajo *wine*.

EL AUTOR

Juan Rafael Fernández García es profesor de educación secundaria y tiene una larga experiencia en la traducción y documentación del software libre. Ha sido coordinador de uno de los Centros que participan en la experiencia andaluza de integrar las TIC en la educación y actualmente trabaja como asesor de formación del profesorado.

RECURSOS

[1] Citado en Rafael Sánchez Montoya, *Ordenador y discapacidad*, 2ª edic., ed. CEPE, Madrid 2002, página 158.

[2] Existen numerosas ampliaciones y adaptaciones del código Braille original. Ha sido ampliado a un código de ocho puntos (Sánchez Montoya le llama *braille computerizado*), de tal manera que una letra individual puede ser codificada con una sola celda, lo que permite 256 combinaciones posibles codificadas según el estándar Unicode. Y disponemos de extensiones para representar fórmulas matemáticas o música. O (lo encuentro en *http://www.tcts.fpms.ac.be/synthesis/w/*) un Braille de grado II utilizado desde 1829 en el que existen contracciones y abreviaturas, con un ahorro del 30 al 50%. El documento que cito lo defiende con ventaja comparativa respecto a los algoritmos de predicción de palabras (los vimos en el artículo anterior): frente a estos y la necesidad de comprobar y corregir la predicción, es fácil aprender de memoria las abreviaturas.

[3] Más información en *http://cidat.once.es/*, en *http://snow.utoronto.ca/technology/products/refreshable-braille.html* y claro, en la wikipedia: teclados Braille (Perkins Brailler, máquinas Perkins); Braille Notetakers: anotadores electrónicos, portátiles Braille; líneas Braille = Terminales Braille = (refreshable) Displays Braille; Impresoras (embossers) Braille.

[4] *http://dave.mielke.cc/brltty/*. Hay manual en español: *http://rt001pvr.eresmas.net/Manual-1.html*.

[5] *http://see-by-touch.sourceforge.net/index_.html*.

[6] *http://www.utoronto.ca/atrc/reference/tech/brailleemb.html*. Sobre hornos fúser cf. *http://www.cepmalaga.com/actividades/Interedvisual/recursos_didacticos_adaptados.htm*.

[7] *http://www.csun.edu/cod/conf/2003/proceedings/71.htm*, recogido en las Actas de la Conferencia de 2003 de la Technology and Persons with Disabilities Conference de la Universidad del Estado de California Northridge.

[8] Con un matiz: es verdad que no existen aplicaciones para logopedia, o para el aprendizaje de los sistemas pictográficos de comunicación por poner dos ejemplos. Pero la tecnología y el conocimiento están ahí, y las aplicaciones gratuitas también, es cuestión de que las dos tradiciones se encuentren.