

ROMs para dos Zaurus

ELIGIENDO MI ROM



Con el paso del tiempo los usuarios de las Zaurus han aprendido a diseñar y a construir su propias variantes de la ROM Linux que proporciona Sharp. El panorama actual está repleto de ROMs cada vez más completas y potentes, llegando a ser un problema el poder responder a la pregunta ¿qué ROM puedo ponerle hoy a mi Zaurus?. **POR ALBERTO PLANAS**

Estamos acostumbrados a adaptar nuestras máquinas Linux añadiendo una nueva versión del kernel, cambiando de gestor de ventanas o habilitando el Composite de las XOrg. Todas estas tendencias también existen en el pequeño mundo de las Zaurus. Los usuarios han aprendido las técnicas básicas para poder crear versiones de ROMs que puedan ser almacenadas de manera correcta dentro de la memoria Flash del equipo. Naturalmente las primeras versiones de estas ROMs eran simples modificaciones de la que trae por defecto la Zaurus. Se eliminaban los juegos y apli-

caciones PIN consideradas superfluas para dejar espacio libre a nuevos y necesarios drivers para las tarjetas wireless y bluetooth, nuevas librerías ncurses, consolas shell, utilidades de compresión, etc. Mientras se jugaba con estas variantes se aprendía más y más sobre la estructura interna de la Zaurus y sobre el funcionamiento de muchos de los periféricos de la misma (dispositivos CF, pantalla, puerto de infrarrojos, pantalla táctil, batería) llegando a plantear arquitecturas alternativas que optimizaban el rendimiento global de la PDA. En este artículo repasaremos algunas de

estas ROMs alternativas de las Zaurus, que en algunos casos son un valor añadido que deberemos tener en cuenta a la hora de decidirnos por la adquisición de un determinado modelo.

Un Saurio en Libertad

Sin duda una de las ROMs más valiosas es OpenZaurus[1]. Como la mayoría de los proyectos de software libre, éste parte de un solo desarrollador que desea tener una ROM completamente libre y mejorada para su Zaurus SL-5500. Poco a poco se ha ido convirtiendo en una distribución LFS (Linux From Scratch)

basada en entornos Opie[2] o GPE[3]. Para quien no lo sepa, las PDAs Linux tienen también su propio entorno gráfico, que es la parte encargada de gestionar las ventanas y eventos producidos por dispositivos tales como la pantalla táctil, el teclado o la introducción de una nueva tarjeta de memoria CF. La empresa TrollTech ha desarrollado su propio entorno de ventanas para sistemas empotrados, Qtopia[4], basado en una versión simplificada de las QT denominada QT/Embedded[5]. Opie no es más que un *fork* de la versión 1.5 de Qtopia que ha ido adquiriendo su propia identidad por medio de modificaciones más o menos importantes tanto a nivel de API cómo por la selección de aplicaciones que le acompañan. Así Opie presenta opciones de configuración y personalización a través de estilos de las que carece Qtopia, además de aplicaciones para el visualizado de PDFs, navegadores webs y multitud de juegos. No todo el mundo desea programar en QT/E y C++, así que un grupo de programadores decidieron poder realizar sus programas en GTK+ y C por medio de un nuevo entorno denominado GPE. A diferencia de Qtopia / Opie éste necesita de un entorno X para poder ejecutarse (XFree86 / KDrive), lo que lo convierte en una aplicación más pesada. Por desgracia GPE no está en la actualidad tan evolucionado como Opie,

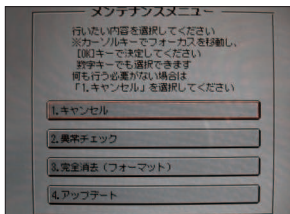


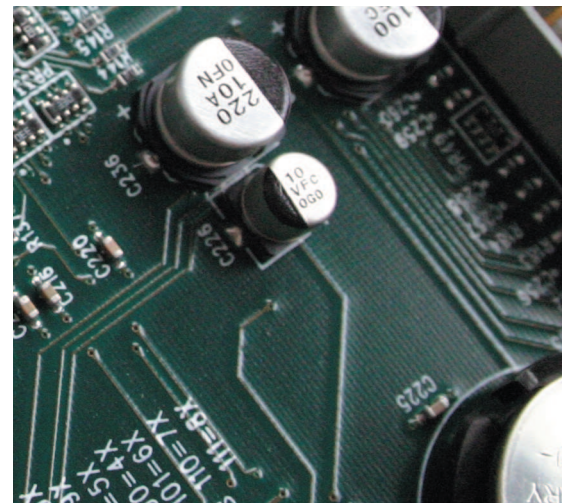
Figura 1: Menú de actualización de la Zaurus. De las tres opciones solo nos interesa la número 4, que es la que procederá a la carga de la ROM desde la tarjeta SD o CF.

carece de aplicaciones esenciales de administración y presenta numerosos errores que necesitan ser corregidos antes de plantearse un uso serio del entorno. De todas maneras el potencial de la opción GTK+ y X Window System es impresionante. ¿Os imagináis importar una sesión X de Thunderbird aplicando sombras y transparencias por medio del Composite? Es interesante ver cómo hasta en los sistemas tan pequeños como la Zaurus aparecen competi-

ciones tecnológicas entre KDE (Qtopia y Opie) / Gnome (GPE), lo que da una idea de la riqueza del software que nos podemos encontrar para estas PDAs.

Instalar OpenZaurus requiere un proceso de escritura en la memoria Flash ROM interna de la Zaurus. Lo primero que hay que hacer es descargar de la web de OpenZaurus[6] la última versión disponible de esta distribución (actualmente la 3.5.2, aunque posiblemente se pueda descargar la 3.5.3 para cuando se publique este artículo). Para instalar OZ en una SL-5500 tendremos primero que elegir el entorno gráfico que deseamos usar y el perfil de la memoria del sistema. Para ser productivos desde un primer momento, es aconsejable usar el entorno Opie y una distribución de memoria que reserve unos 40M para RAM y el resto de los 64M de la Zaurus, 24M, para poder instalar nuestras aplicaciones.

Nos aseguraremos de almacenar los dos ficheros elegidos de la lista de descarga en la tarjeta CF bajo los nombres "zImage" para el kernel y "initrd.bin" y procederemos al *flasheado* siguiendo los pasos descritos en la Cuadro 1 para el modelo SL-5500. Si deseamos usar esta



ROM para otros modelos de Zaurus, descargaremos los ficheros adecuados y procederemos a ejercer una presión en el teclado al estilo vulcaniano dependiente del modelo de la Zaurus. En la Figura 3 se puede ver qué aspecto tiene una OpenZaurus 3.5.2 con Opie.

No nos confundamos: esta ROM es para hackers, pero también puede ser usada por usuarios normales deseosos

de tener un abanico de aplicaciones libres y actualizadas. Realmente es una distribución que necesita un poco más de trabajo, pero para admirar su potencia debemos entender cómo está construida en la actualidad.

OpenZaurus no es más que una de las múltiples distribuciones que pueden generarse a partir de otro proyecto menos conocido: bitbake. Éste consta de un conjunto de scripts en Python que, a partir de un recetario o ficheros *.bb*, es capaz de generar programas compilados para múltiples arquitecturas. Con un sencillo comando podremos ir generando desde paquetes para máquinas Zaurus hasta una imagen ROM de la última versión de OpenZaurus en desarrollo. Es, por tanto, un framework para el nunca cómodo cross-compiling. Bitbake constituye el núcleo del proyecto OpenEmbedded[7], que se

Listado 1: install.sh

```
01 #!/bin/sh
02 mkdir zaurus
03 cd zaurus
04 mkdir openembedded
05 mkdir bitbake
06 mkdir build
07 mkdir sources
08 svn co
   svn://svn.berlios.de/bitbake/t
   runk/bitbake
09 bk clone
   bk://openembedded.bkbits.net/o
   penembedded
```

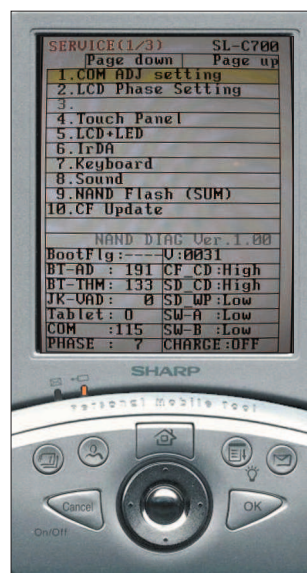


Figura 2: Peligroso menú de servicio con múltiples opciones de calibración, testeo y backup/restore de la Zaurus.

ha separado de éste por cuestiones de comodidad. Ahora el objetivo de OpenEmbedded es el de aglutinar en un solo proyecto el conjunto de descripciones de paquetes y arquitecturas que deberá compilar bitbake. Para aclarar los conceptos vamos a instalar los proyectos bitbake y openembedded, y diseñaremos un fichero *.bb* para el programa *aircrack* (ver artículo anterior) que añadiremos a nuestro repositorio para una compilación cruzada automática.

Suponiendo que se tiene instalado los programas de control de versiones Subversion[8] y BitKeeper[9], instalar y usar bitbake y openembedded va a resultar de lo más sencillo. Solo hay que ejecutar el pequeño script del Listado 1. Éste se encargará de crear los directorios adecuados y descargar los dos proyectos

antes mencionados. Realmente estos proyectos son muy dinámicos y cambian con mucha frecuencia por lo que es aconsejable ir actualizando cada cierto tiempo nuestras copias locales con estos comandos:

```
cd bitbake
svn up
cd ../openembedded
bk pull; bk -r co -q
```

Ahora necesitamos crear un fichero de configuración dentro del directorio *build/conf* (ver Listado 2) que contendrá las principales directivas del entorno. En este fichero indicaremos cual es el procesador para el que deseamos generar binarios, para qué distribución en particular vamos a crear imágenes ROMs y

paquetes y, finalmente, qué versión del kernel usará nuestra distribución. El proceso es muy flexible, por lo que es recomendable consultar la documentación[10] de la web de *openembedded*. Antes de compilar nada necesitaremos establecer la siguiente variable de entorno:

```
export BBPATH=$HOME/zaurus/
build:$HOME/zaurus/
openembedded
```

Perfecto, ahora si queremos generar la imagen ROM de Opie y OpenZaurus solo tendremos que escribir, dentro del directorio *build* el comando *bitbake opie-image*. Observaremos tranquilamente cómo se van a ir descargando los paquetes en código fuente de los compiladores, el kernel y del resto del sistema de manera automática. En una primera

Cuadro 1: Preparando los Flashers

El proceso de escritura en la memoria Flash ROM generalmente requiere introducir combinaciones secretas de teclas y acceder a peligrosos menús del sistema. Si bien es cierto que se corre un riesgo al cambiar el contenido de la memoria Flash, si se respeta el procedimiento no sucederá nada irreparable. Vamos a describir los pasos que hay que dar para *flashear* la SL-5500 y la C860.

Modelo SL-5500

- Asegurarse que se tiene una CF de 32M o más formateada en FAT16.
- Almacenar en la CF los ficheros *zimage* e *initrd.bin* (o solo es *Ospack* si se va a recuperar la ROM original); introducirla en la ranura de la PDA.
- Enchufar la Zaurus a la corriente.
- Abrir la pestaña trasera que impide la apertura de la tapa de la batería.
- Pulsar las teclas "C" y "D" mientras se presiona el botón trasero de *reset*.
- Esperar dos o tres minutos mientras los dos LEDs de la Zaurus se mantienen fijos y encendidos. Este es el momento crítico del procedimiento, hay que dejar que el proceso continúe sin interrupciones.
- Pulsar otra vez el botón trasero de *reset* de la PDA.
- Colocar la tapa, cerrar la pestaña de seguridad y encender la Zaurus.

Modelo C860 – Modo Flash

- Almacenar en una SD o CF de 64M los ficheros *updater.sh*, *zimage.bin* e *initrd.bin*

- Abrir la pestaña trasera para poder sacar la tapa de la batería y la propia batería. Esperar 5 segundos.
- Introducir la batería y la tapa, cerrar la pestaña y enchufar la Zaurus.
- Pulsar el botón "OK" del teclado mientras se pulsa el botón trasero de encendido.
- Seleccionar la cuarta opción del menú en Japones (ver Figura 1). Por si alguien tiene curiosidad la primera opción es para salir de este menu, la segunda realiza algún tipo de chequeo en la máquina y la tercera formatea la memoria flash interna.
- Ahora sale otro menú en Japones, selecciona la opción 2 si la ROM está en la CF, si está en la ranura SD selecciona la opción 3.

Modelo C860 – Modo NAND

- Abrir la pestaña de la batería y extraer la tapa y la batería de la Zaurus.
- Esperar 10 segundos (en algunos casos se ha documentado que hay que esperar hasta 5 minutos la primera vez que se realiza este procedimiento).
- Introducir la batería y la tapa.
- Pulsar las teclas "D" y "M" a la vez que se coloca la pestaña trasera en su posición original (cerrado).
- En la pantalla sale un menú en inglés (ver Imagen 2) extremadamente peligroso. Es aconsejable usar solo la opción 10 de la tercera página (NAND Flash Restore).

Listado 2: local.conf

```
01 # Usar como plantilla
    openembedded/conf/local.conf.sample
02 DL_DIR =
    "${HOME}/zaurus/sources"
03 BBFILES =
    "${HOME}/zaurus/openembedded/packages/**/*.bb"
04 BBMASK = ""
05 PREFERRED_PROVIDERS =
    "virtual/qt:qte
    virtual/libqpe:libqpe-opie"
06 PREFERRED_PROVIDERS += "
    virtual/libSDL:libSDL-qpe"
07 PREFERRED_PROVIDERS += "
    virtual/${TARGET_PREFIX}gcc-initial:gcc-cross-initial"
08 PREFERRED_PROVIDERS += "
    virtual/${TARGET_PREFIX}gcc:gcc-cross"
09 PREFERRED_PROVIDERS += "
    virtual/${TARGET_PREFIX}g++:gcc-cross"
10 MACHINE = "collie"
11 DISTRO = "openzaurus-3.5.3"
12 IMAGE_FSTYPES = "jffs2 tar"
13 BBINCLUDELOGS = "yes"
14 CVS_TARBALL_STASH =
    "http://www.treke.net/oe/source/"
```

etapa bitbake compilará el paquete GCC de manera nativa para nuestra máquina, pero indicando que el código que debe generar ha de ser para el procesador que usa la Zaurus (procesadores ARM). Con este compilador se irán generando los binarios para nuestra plataforma de destino y todo esto lo tenemos automatizado gracias a bitbake.

Incluir un paquete es tan sencillo como crear un directorio para la nueva aplicación y poner dentro de él un fichero `.bb` que indique cómo generarla. El programa `aircrack` para la búsqueda de claves WEP no está dentro de `openembedded`. Lo incluiremos nosotros creando el directorio con un `mkdir openembedded/packages/aircrack` y crearemos el fichero `.bb` adecuado (ver Listado 3). Generaremos de manera rápida un paquete instalable directamente en nuestra Zaurus con el comando:

Listado 3: aircrack_2.1.bb

```
01 DESCRIPTION = "802.11 sniffer
    and WEP key cracker for
    Windows and Linux"
02 SECTION = "console/network"
03 PRIORITY = "optional"
04 LICENSE = "GPL"
05 SRC_URI =
    "http://www.cr0.net:8040/code/
    network/aircrack-${PV}.tgz"
06
07 EXTRA_OEMAKE = "'CC=${CC}'
    'BIND=${CC}' 'AS=${CC}' -c'
    'CPP=${CPP}' \
08 'CFLAGS=-I. -DUNIX
    ${CFLAGS}' 'INSTALL=install' \
09 'BINFLAGS=0755'
    'INSTALL_D=install -d'"
10
11 do_compile() {
12     oe_runmake
13 }
14
15 do_install() {
16     oe_runmake
    prefix=${D}${prefix} \
17     BINDIR=${D}/${bindir}
    MANDIR=${D}/${mandir}/man1 \
18     install
19 }
```

```
bitbake -b $HOME/openembedded/
packages/aircrack/
aircrack_2.1.bb
```

Creo que es ahora cuando ya podemos entender la importancia, la versatilidad y la potencia de este proyecto. ¡A crear ficheros `.bb` para la Zaurus!

La Cacko ROM

Cuando adquirimos una Zaurus de la familia de las Cxxx lo más probable es que venga en Japonés. El primer paso es, por tanto, adaptarlo a algún idioma que podamos entender (vale, nos conformamos con el inglés). De entre todas las opciones posibles de traducción la mejor es directamente sobre escribir la ROM de Sharp por la Cacko ROM. Ésta es una variante de la ROM original en la que se han eliminado algunas aplicaciones como el diccionario inglés-japonés para dejar espacio a otras como el reproductor multimedia Kino2. Esta ROM ha pasado por varios procesos de refinamiento y ahora es posible descargar la versión 1.22[11] junto con un primer paquete de corrección de errores[12]. El proceso de instalación de esta ROM tiene dos partes, en una primera parte particionaremos la memoria y en la siguiente escribiremos los datos de la ROM en la memoria Flash de la Zaurus. Lo primero que hay que hacer es asegurarse de descargar y almacenar los tres ficheros que componen la ROM en una CF de 64Mtal y como se indica en la Tabla 1 (epí-

grafe **Modelo C860 – Modo Flash**). Si seguimos los pasos ahí indicados la Zaurus arrancará desde la tarjeta de memoria CF y mostrará un menú ncurses con diversas opciones. Debemos seleccionar *5 Flash Repartition* e introducir 27 cuando nos pregunte por el tamaño de la partición raíz. Finalizada esta etapa el sistema se reiniciará solo y, antes de que empiece de nuevo la carga del sistema operativo, deberemos repetir todos los pasos hasta volver a visualizar el menú en modo texto anterior. Es en esta nuestra segunda visita cuando podremos seleccionar la opción *1 Install new ROM*. Ya sólo nos queda esperar a que resetee de nuevo y dejar, ahora sí, que cargue el nuevo sistema operativo para poder disfrutar de su nueva estética (ver Figura 4).

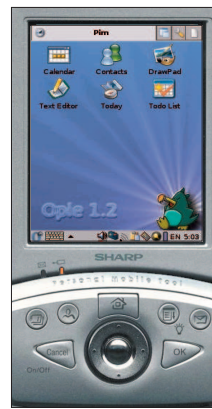
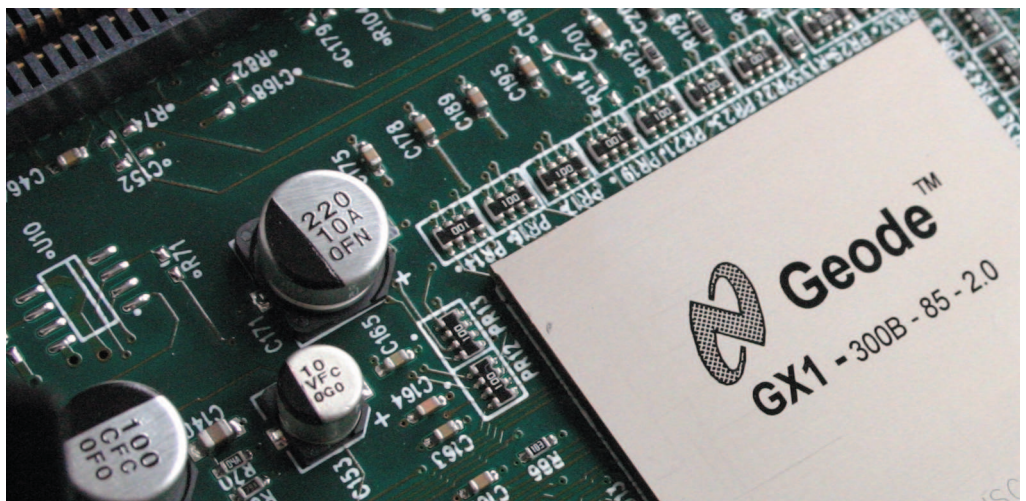


Figura 3: Escritorio por defecto de una OpenZaurus con Opie.

Otra opción, pero por ahora menos utilizada, consiste en instalar la versión de OpenZaurus para este modelo. Esta alternativa irá adquiriendo cada vez más interés en cuanto tengamos un kernel 2.6 completamente estable, ya que esta versión promete rendimientos nunca vistos antes en este tipo de dispositivos. Por desgracia debido a temas de patentes y especificaciones cerradas no podremos disfrutar, a falta de algún milagro, de las tarjetas SD en las máquinas SL-5500 bajo kernel 2.6.

Unas X solo para mayores

Hay un desarrollo que realmente merece la pena mencionarlo aquí, y es la



pdaXrom[13]. Su objetivo es entregar una distribución Linux completamente basada en XFree86 (KDrive realmente) con un gestor de ventanas estándar. En la actualidad se pueden usar los gestores Matchbox, IceWM, Fluxbox, FVWM y XFCE entre otros. La ventaja esencial de usar XFree frente a las opciones Qtopia / Opie / GPE es la inmediatez a la hora de portar nuevas aplicaciones a esta plataforma, de hecho solo hay que ver la lista de aplicaciones existentes para esta ROM (AbiWord, Dillo, XEmacs, Firefox, The Gimp ...). Realmente impresionante es también, por desgracia, la cantidad de recursos de memoria que necesita tanto en RAM como en espacio de almacenamiento. Es por eso que esta ROM tiene más sentido en otros modelos como la C3000 que incorpora un disco duro de 4.4G, aunque en la actualidad no hay ninguna versión que soporte completamente este modelo tan moderno. Todo llegará.

Qué Hacer Cuando Algo va Mal

En cierta manera nos sentimos responsables de incitar al lector a jugársela con su amado cacharrito. El procedimiento de escritura de la ROM no está exento de riesgos, y la variedad y atractivo de las ROMs es tan grande que es difícil de evitar el deseo de ir cambiando cada poco tiempo la de nuestra Zaurus. Si por alguna razón algo falla y la PDA deja de arrancar tras un *flasheado interruptus*, el



Figura 4: Cacko ROM en una SL-C860. Bonita selección de iconos ¿verdad?

mejor consejo que podemos dar es intentar restaurar la ROM original. Si estamos trabajando con una SL-5500 podemos encontrar la ROM de Sharp en <http://www.elsix.org/downloads/5500v313Ospack.zip>. Descargaremos ese fichero, lo descomprimiremos y copiaremos el fichero así obtenido en una CF para volver a *flashear* desde ahí siguiendo los pasos ya indicados en la Tabla 1. La ROM de Sharp viene acompañada por una serie de aplicaciones más o menos útiles que tendremos que descargar desde otra dirección[14].

En caso de que el error se presente en el modelo C860 tendremos que acceder a otro de los menús secretos de la Zaurus. Usaremos el procedimiento descrito en la Tabla 1 bajo el epígrafe **Modelo C860 – Modo Flash**. La idea consiste en volver a volcar el contenido íntegro de la memoria NAND de la Zaurus que esta traía de fábrica. Podemos descargar este con-

tenido desde [15] y copiarlo a una tarjeta SD o CF de más de 128M. Con ella introducida en la ranura correspondiente, activaremos el menú de servicio e iremos a la tercera página / opción 10 (*NAND Flash Restore*) para completar la restauración de la ROM original. ■

RECURSOS

- [1] OpenZaurus: <http://www.openzaurus.org/>
- [2] Opie: <http://opie.handhelds.org/cgi-bin/moin.cgi/>
- [3] GPE: <http://gpe.handhelds.org/>
- [4] Qtopia: <http://www.trolltech.com/products/qtopia/>
- [5] QT/E: <http://www.trolltech.com/products/embedded/index.html>
- [6] OpenZaurus 3.5.2 <http://www.openzaurus.org/official/unstable/3.5.2/sl5000,sl5500/>
- [7] OpenEmbedded: <http://www.openembedded.org/>
- [8] Subversion: <http://subversion.tigris.org/>
- [9] BitKeeper: <http://www.bitkeeper.com/>
- [10] OpenEmbedded GettingStarted: <http://openembedded.org/cgi-bin/moin.cgi/GettingStarted>
- [11] Cacko ROM 1.22: <http://cacko.oesf.org/downloads/rom/1.22/slc7x0-Qtopia-1.22-1346311204.zip>
- [12] Cacko ROM 1.22 HotFix A: http://cacko.oesf.org/downloads/rom/1.22/cacko-qtopia-rom-hotfix_1.22a_arm.ipk
- [13] pdaXrom: <http://www.pdaxrom.org/>
- [14] Sharp ROM Utilidades: <http://www.elsix.org/downloads/5500v310Apps.zip>
- [15] Backup NAND: <http://downloads.conics.net/pda/zaurus-sl-c700/service-menus/original-backups/>



EL AUTOR

Alberto Planas es desarrollador de aplicaciones bajo entornos libres desde hace varios años. Aficionado a la tecnología desde siempre, alterna sus horas de sueño con las horas dedicadas al estudio de las Redes Bayesianas, programación con las QT, perfeccionamiento de C++, desarrollo en Java y mil cosas más.

