GRUPO DE TRABAJO

"Herramientas Open Source para la carrera de Ingeniería Electrónica"

Sebastián Viviani (guilly@electron.frba.utn.edu.ar)

IMPORTANTE: en el presente documento se usa el término Open Source para referirse de forma indistinta a software de código abierto, a software libre o a software comercial con licencia gratuita. Si bien la generalización es incorrecta, ya que se mezclan licencias y filosofias diferentes, a efectos del resultado final si pueden agruparse.

Introducción

Gran parte de las herramientas usadas para el diseño electrónico, en todos los niveles y campos, son aplicaciones comerciales de elevada potencia y costo. Mientras cursan la carrera, los alumnos recurren muchas veces a estos softwares utilizando versiones piratas, para resolver los diferentes proyectos o trabajos prácticos.

Esta clase de iniciativas causan varias consecuencias negativas para la formación profesional del alumno, por ejemplo:

- la erronea creencia de que no existen alternativas más baratas o económicas,
- el abuso de recursos para solucionar los problemas, ya que normalmente utilizan programas excesivamente potentes para los simples problemas que tienen que resolver.
- la incapacidad del alumno de detectar la real dimensión de un problema.
- el desprecio del alumno por el trabajo realizado por la empresa que desarrolló el software.
- la noción de que utilizar cualquier software es un derecho adquirido.

Y lo que es más grave, muchas de estas costumbres a veces son transmitidas por los mismos docentes, y se continuán aplicando en el campo profesional. Se suele argumentar que es valioso que un alumno aprenda una herramienta comercial potente, que se usa en el ámbito profesional, aunque se olvida que allí tampoco se paga su licencia.

Es por eso que se desea plantear la necesidad de incorporar herramientas de desarrollo opensource a la carrera, identificando los proyectos que existen actualmente en Internet, en un primer paso, y luego colaborando para el crecimiento de los mismos.

Ejemplos de proyectos que pueden ser útiles, y de los que la UTN puede beneficiarse con su uso y desarrollo son, por ejemplo: SDCC, PikLab, Magic, Alliance, Icarus, gEda, octave, Kdevelop, Anjuta, Codeblocks, etc.

Objetivos

- Analizar e identificar software OpenSource que pueda utilizarse para desarrollos electrónicos.
- Testear y documentar su instalación y uso.
- Realizar seminarios para capacitar a los alumnos y docentes.
- Generar documentación para facilitar su incorporación a las materias de grado.
- Colaborar en el desarrollo de las herramientas.

Recursos necesarios

Página web y server virtual dedicado.
Espacio físico para reuniones periódicas (semanal/quincenal/mensual)
Espacio físico para seminarios y/o capacitaciones
4 o más personas (la mitad graduados y la mitad alumnos, todos rentados)

Alternativas de financiamiento

- Por parte de la facultad, de la misma forma que se maneja un grupo de investigación.
- Externo, de forma autónoma, buscando recaudar fondos a través del dictado de seminarios y capacitaciones a graduados y/o empresas.
- Mediante la publicación de apuntes y/o material bibliográfico.
- Mediante donaciones y aportes espontáneos de terceros.

Metodología y etapas de trabajo

Como primer tarea, es necesario relevar las necesidades del actual plan de estudios: qué se usa actualmente en cada materia y porqué. Se debe consultar también a profesionales con amplia experiencia en la industria que herramientas utilizan en el día a día, y que aspectos o características valoran de los mismos.

De esta forma, se podrá estimar mejor el potencial de cada software a evaluar, y seleccionar mejor las herramientas en las que se realizarán las actividades. Se buscará cubrir las necesidades académicas y aprovechar las mejores perspectivas pensando en el desarrollo profesional.

Luego, es necesario encarar cada herramienta por separado, en ciclos acotados en el tiempo y que incluyan a todos los integrantes, que básicamente serán:

- Instalar y configurar el programa en varias plataformas, documentando el proceso.
- Usar el programa con un proyecto-testigo, para documentar el uso y las capacidades.
- Analizar ventajas y desventajas con respecto a otros programas del mismo rubro, y cómo se podría mejorar el mismo.
- Realizar un seminario de capacitación para docentes o alumnos o profesionales, exponiendo los resultados.
- Recibir feedback de los participantes de el/los seminarios, para poder mejorar y/o enfocar el trabajo.

Es importante realizar el ciclo completo, aunque la herramienta demuestre en etapas tempranas problemas sin solución, para al menos poder advertir sobre esos problemas o desventajas. Una recomendación negativa también es una ayuda.

Estas etapas deberían poder desarrollarse en el intervalo de un cuatrimestre, o menos. Una vez desglosada la herramienta, se evaluará si vale la pena emplear tiempo para colaborar con su desarrollo y crecimiento. En ese caso se deberá comenzar un segundo ciclo, más relacionado al desarrollo de software, y que involucrará trabajar con el código fuente (si es posible) del programa, realizando los aportes en nombre de la UTN-FRBA.

Es importante mantener contacto con el/las personas que mantienen el proyecto original, para que estos también puedan aprovechar el trabajo realizado, y aportar el punto de vista de parte del desarrollador.

Ejemplos de aplicación

- *Diseño de PCB*
- *Desarrollo de Software Embedded*
- *Simulación Matemática*
- *Diseño, simulación y síntesis en VHDL*
- *Simulación de circuitos analógicos*
- *Diseño de IC's*
- *Análisis de Líneas de transmisión y parámetros distribuidos*
- *Analizadores de protocolos de datos*