novática
CEPIS UPGRADE Revista de la Asociación de Técnicos de Informática

Novática, revista fundada en 1975 y decana de la prensa informática española, es el órgano oficial de expresión y formación continua de **ATI** (Asociación de Técnicos de Informática), organización que edita también la revista **REICIS** (Revista Española de Innovación, Calidad e Ingeniería del Software). ***Novática*** edita asimismo **UP**GRADE, revista digital de **CEPIS** (*Council of European Professional Informatics Societies*), en lengua inglesa, y es miembro fundador de **UP**ENET (**UP**GRADE ***E**uropean **NET**work*).

<http://www.ati.es/novatica/>
<http://www.ati.es/reicis/>
<http://www.upgrade-cepis.org/>

ATI es miembro fundador de **CEPIS** (*Council of European Professional Informatics Societies*) y es representante de España en **IFIP** (*International Federation for Information Processing*); tiene un acuerdo de colaboración con **ACM** (*Association for Computing Machinery*), así como acuerdos de vinculación o colaboración con **AdaSpain**, **AI2**, **ASTIC**, **RITSI** e **Hispalinux**, junto a la que participa en **ProInnova**.

Consejo Editorial
Antoni Carbonell Nogueras, Juan ManuelCueva Lovelle, Juan Antonio Esteban Iriarte,Francisco López Crespo, Julián Marcelo Cocho, Celestino Martín Alonso, Josep Molas i Bertrán, Olga Pallás Codina, Fernando Piera Gómez (Presidente del Consejo), Ramón Puigjaner Trepat, Miquel Sàrries Griñó, Asunción Yturbe Herranz

Coordinación Editorial
Llorenç Pagés Casas <pages@ati.es>
Composición y autoedición
Jorge Llácer Gil de Ramales
Traducciones
Grupo de Lengua e Informática de ATI <http://www.ati.es/gt/lengua-informatica/>, Dpto. de Sistemas Informáticos - Escuela Superior Politécnica - Universidad Europea de Madrid
Administración
Tomás Brunete, María José Fernández, Enric Camarero, Felicidad López

Secciones Técnicas - Coordinadores
Acceso y recuperación de la información
José María Gómez Hidalgo (Optenet), <jmgomezh@yahoo.es>
Manuel J. Maña López (Universidad de Huelva), <manuel.mana@diesia.uhu.es>
Administración Pública electrónica
Francisco López Crespo (MAE), <flc@ati.es>
Arquitecturas
Enrique F. Torres Moreno (Universidad de Zaragoza), <enrique.torres@unizar.es>
Jordi Tubella Morgadas (DAC-UPC), <jordit@ac.upc.es>
Auditoría SITIC
Marina Touriño Troitiño, <marinatourino@marinatourino.com>
Manuel Palao García-Suelto (ASIA), <manuel@palao.com>
Derecho y tecnología
Isabel Hernando Collazos (Fac. Derecho de Donostia, UPV), <ihernando@legaltek.net>
Elena Davara Fernández de Marcos (Davara & Davara), <edavara@davara.com>
Enseñanza Universitaria de la Informática
Joaquín Ezpeleta Mateo (CPS-UZAR), <ezpeleta@posta.unizar.es>
Cristóbal Pareja Flores (DSIP-UCM), <cpareja@sip.ucm.es>
Entorno digital personal
Alonso Alvarez García (TID), <aag@tid.es>
Diego Gachet Páez (Universidad Europea de Madrid), <gachet@uem.es>
Estándares Web
Encarna Quesada Ruiz (Oficina Española del W3C) <equesada@w3.org>
José Carlos del Arco Prieto (TCP Sistemas e Ingeniería) <jcarco@gmail.com>
Gestión del Conocimiento
Joan Baiget Solé (Cap Gemini Ernst & Young), <joan.baiget@ati.es>
Informática y Filosofía
José Angel Olivas Varela (Escuela Superior de Informática, UCLM) <joseangel.olivas@uclm.es>
Karim Gherab Martín (Harvard University) <kgherab@gmail.com>
Informática Gráfica
Miguel Chover Sellés (Universitat Jaume I de Castellón), <chover@lsi.uji.es>
Roberto Vivó Hernando (Eurographics, sección española), <rvivo@dsic.upv.es>
Ingeniería del Software
Javier Dolado Cosín (DLSI-UPV), <dolado@si.ehu.es>
Luis Fernández Sanz (PRIS-EI-UEM), <lufern@dpris.esi.uem.es>
Inteligencia Artificial
Vicente Botti Navarro, Vicente Julián Inglada (DSIC-UPV)
<{vbotti, vinglada}@dsic.upv.es>
Interacción Persona-Computador
Julio Abascal González (FI-UPV), <julio@si.ehu.es>
Lengua e Informática
M. del Carmen Ugarte García (IBM), <cugarte@ati.es>
Lenguajes Informáticos
Andrés Marín López (Univ. Carlos III), <amarin@it.uc3m.es>
J. Angel Velázquez Iturbide (ESCET-URJC), <a.velazquez@escet.urjc.es>
Lingüística computacional
Xavier Gómez Guinovart (Univ. de Vigo), <xgg@uvigo.es>
Manuel Palomar (Univ. de Alicante), <mpalomar@dlsi.ua.es>
Mundo estudiantil y jóvenes profesionales
Federico G. Mon Trotti (RITSI) <gnu.fede@gmail.com>
Mikel Salazar Peña (Area de Jovenes Profesionales, Junta de ATI Madrid), <mikeltxo_uni@yahoo.es>
Profesión Informática
Rafael Fernández Calvo (ATI), <rfcalvo@ati.es>
Miquel Sàrries Griñó (Ayto. de Barcelona), <msarries@ati.es>
Redes y servicios telemáticos
José Luis Marzo Lázaro (Univ. de Girona), <joseluis.marzo@udg.es>
Germán Santos Boada (UPC), <german@ac.upc.es>
Seguridad
Javier Areitio Bertolín (Univ. de Deusto), <jareitio@eside.deusto.es>
Javier López Muñoz (ETSI Informática-UMA), <jlm@lcc.uma.es>
Sistemas de Tiempo Real
Alejandro Alonso Muñoz, Juan Antonio de la Puente Alfaro (DIT-UPM),
<{aalonso,jpuente}@dit.upm.es>
Software Libre
Jesús M. González Barahona, Pedro de las Heras Quirós (GSYC-URJC),
<{jgb,pheras}@gsyc.escet.urjc.es>
Tecnología de Objetos
Jesus García Molina (DIS-UM), <jmolina@um.es>
Gustavo Rossi (LIFIA-UNLP, Argentina), <gustavo@sol.info.unlp.edu.ar>
Tecnologías para la Educación
Juan Manuel Dodero Beardo (UC3M), <dodero@inf.uc3m.es>
César Pablo Corcoles Briongo (UOC), <ccorcoles@uoc.edu>.
Tecnologías y Empresa
Didac López Viñas (Universitat de Girona), <didac.lopez@ati.es>
Francisco Javier Cantais Sánchez (Indra Sistemas), <fjcantais@gmail.com>
TIC y Turismo
Andrés Aguayo Maldonado, Antonio Guevara Plaza (Univ. de Málaga)
<{aguayo, guevara}@lcc.uma.es>

Las opiniones expresadas por los autores son responsabilidad exclusiva de losmismos. ***Novática*** permite la reproducción, sin ánimo de lucro, de todos los artículos, a menos que lo impida la modalidad de © o *copyright* elegida por el autor, debiéndose en todo caso citar su procedencia y enviar a ***Novática*** un ejemplar de la publicación.

Coordinación Editorial, Redacción Central y Redacción ATI Madrid
Padilla 66, 3º, dcha., 28006 Madrid
Tlfn.914029391; fax.913093685 <novatica@ati.es>
Composición, Edición y Redacción ATI Valencia
Av. del Reino de Valencia 23, 46005 Valencia
Tlfn./fax 963330992 <secreval@ati.es>
Administración y Redacción ATI Cataluña
Via Laietana 46, ppal. 1ª, 08003 Barcelona
Tlfn.934125235; fax 934127713 <secregen@ati.es>
Redacción ATI Andalucía
Isaac Newton, s/n, Ed. Sadiel,
Isla Cartuja 41092 Sevilla, Tlfn./fax 954460779 <secreand@ati.es>
Redacción ATI Aragón
Lagasca 9, 3-B, 50006 Zaragoza.
Tlfn./fax 976235181 <secreara@ati.es>
Redacción ATI Asturias-Cantabria <gp-astucant@ati.es>
Redacción ATI Castilla-La Mancha <gp-clmancha@ati.es>
Suscripción y Ventas <http://www.ati.es/novatica/interes.html>, ATI Cataluña, ATI Madrid
Publicidad
Padilla 66, 3º, dcha., 28006 Madrid
Tlnf.914029391; fax.913093685 <novatica@ati.es>
Imprenta: Derra S.A., Juan de Austria 66, 08005 Barcelona.
Depósito legal: B 15.154-1975 – ISSN: 0211-2124; CODEN NOVAEC
Portada: "Salida de la habitación 101" - Concha Arias Pérez / © ATI
Diseño: Fernando Agresta / © ATI 2003

Nº 190, noviembre-diciembre 2007, año XXXIII

sumario

editorial
Estudiantes y jóvenes profesionales, clave del futuro de ATI > 02

en resumen
El poder de laas comunidades > 02
Llorenç Pagés Casas

monografía
Sortware libre: investigación y desarrollo
(En colaboración con **UP**GRADE*)*
Editores invitados: *Manuel Palomo Duarte, José Rafael Rodríguez Galván, Israel Herraiz Tabernero y Andrea Capiluppi*

Presentación. Sofware libre: investigación y desarrollo > 03
Andrea Capiluppi, José Rafael Rodríguez Galván, Manuel Palomo Duarte, Israel Herraiz Tabernero

La necesidad de investigar sobre software libre en Europa > 06
Israel Herraiz Tabernero, Rafael Rodríguez Galván, Manuel Palomo Duarte

De la catedral al bazar: un estudio empírico del ciclo de vida de los proyectos basados en comunidades de voluntarios > 09
Andrea Capiluppi, Martin Michlmayr

Los bienes comunes como nueva economía y lo que esto significa para la investigación > 17
Richard P. Gabriel

Software libre para la gestión de proyectos de investigación > 20
Israel Herraiz Tabernero, Juan José Amor Iglesias, Álvaro del Castillo San Félix

Innovación tecnológica en comunicaciones móviles desarrollada con Software Libre: Campus Ubicuo > 25
Javier Carmona Murillo, José Luis González Sánchez, Manuel Castro Ruiz

El modelo de la Oficina de Software Libre de la Universidad de Cádiz en la universidad española > 31
José Rafael Rodríguez Galván, Manuel Palomo Duarte, Juan Carlos González Cerezo, Gerardo Aburruzaga García, Antonio García Domínguez, Alejandro Álvarez Ayllón

Aprendiendo a introducir una innovación en un proyecto basado en Software Libre > 36
Christopher Oezbek, Lutz Prechelt

Optimización del proceso de *render* 3D distribuido con software libre > 41
Carlos González Morcillo, Gerhard Weiss, David Vallejo Fernández, Luis Jiménez Linares, Javier Albusac Jiménez

secciones técnicas

Mundo estudiantil y jóvenes profesionales
SWAML, Semantic Web Archive of Mailing Lists > 49
Sergio Fernández López, Diego Berrueta Muñoz, José Emilio Labra Gayo

TCOS: uso de terminales ligeros en las aulas > 52
Mario Izquierdo Rodríguez

Porting de GCC al microcontrolador Microchip PIC16F877 > 55
Pedro José Ramírez Gutiérrez

SubDownloader > 58
Iván García Cortijo

Software Libre en la Enseñanza: primeras jornadas organizadas por OuSLi en el ámbito de la educación > 61
José Ramón Méndez Reboredo, Enrique Estévez Fernández, Florentino Fernández Riverola, Daniel González Peña

Referencias autorizadas > 64

sociedad de la información

Nueva Economía
Las TIC y la Ciencia, Ingeniería y Gestión de los Servicios > 69
Gregorio Martín Quetglas, Vicente Cerverón Lleó, Francisco J. Gálvez Ramírez

Programar es crear
Todas las palabras son capicúas (CUPCAM 2006, problema F, solución) > 73
Óscar Martín Sánchez

Las luces de la escalera (CUPCAM 2006, problema G, enunciado) > 74
Julio Mariño Carballo

Permutaciones con un número dado de inversiones (CUPCAM 2006, problema H, enunciado) > 75
Manuel Abellanas Oar, Luis Hernández Yáñez

asuntos interiores

Coordinación Editorial / Programación de Novática > 76
Normas para autores / Socios Institucionales > 77

Monografía del próximo número: "*Gobierno de las TIC*"

Porting de GCC al microcontrolador Microchip PIC16F877

Pedro José Ramírez Gutiérrez
Ganador del I Concurso Universitario de Software Libre, categoría "Sistemas", Universidad de Málaga.

<pjanragu@telefonica.net>

Resumen: *El Porting de GCC a los microcontrolador Microchip supone, frente a numerosas propuestas no libres o a propuestas incompletas y sin vida como SDCC, la única posibilidad de programar en lenguaje C con licencia GPL para microcontroladores de la casa Microchip.*

Palabras clave: *back-end, GCC, lenguaje ANSI C, microcontrolador Microchip, PIC16, porting.*

Autor

Pedro José Ramírez Gutiérrez ha desarrollado el Porting de GCC a PIC16F877 como proyecto fin de carrera en la Universidad de Málaga, bajo la dirección de David Santo Orcero, profesor del departamento de Lenguajes y Ciencias de la Computación. Con la defensa del proyecto fin de carrera, en el que recibió la calificación de Matrícula de Honor, ha obtenido finalmente el título de Ingeniero en Informática en el año 2007. Este mismo proyecto resultó ganador del I Concurso Universitario de Software Libre en la categoría de "Sistemas". Dentro de su experiencia está haber trabajado en el desarrollo de un sistema de sensores distribuidos como parte de un proyecto OTRI (*Oficina de Transferencia de Resultados de Investigación*)**.** Entre sus planes futuros está mantener y extender el soporte de GCC a la mayor cantidad posible de microcontroladores de Microchip.

1. Introducción

Un microntrolador es un circuito integrado que alberga en su interior las funciones de un computador completo de limitadas prestaciones. Un microcontrolador típico tendrá un generador de reloj integrado y una pequeña cantidad de memoria fija y de memoria RAM, con lo cual lo único que necesita para funcionar es un programa de control y un cristal de reloj. Los microcontroladores están extremadamente especializados para abaratar costes de hardware. Por ello, hay una amplia variedad de modelos que difieren en tamaño de memoria, tipo de memoria de programa y dispositivos de entrada/ salida. Como dispositivos de E/S tenemos una gran variedad como convertidores analógico-digital, temporizadores, UARTs (*Universal Asynchronous Receiver-Transmitter*) y buses especializados, como I2C, CAN o USB.

Para comenzar el Porting de GCC a la arquitectura PIC de Microchip [1] se ha escogido la gama media; ya que son los que más se emplean en desarrollos comunes, y además no disponen de compiladores en C libres y plenamente funcionales. La gama baja tiene demasiada poca memoria y poca potencia como para que el compilador de C sea funcional. De la gama media se ha escogido el PIC16F877 por ser un microcontrolador extremadamente barato y popular entre los desarrolladores de soluciones para microcontroladores, al tener la mejor relación entre precio y prestaciones.

GCC [2], el compilador de GNU, tiene una arquitectura fuertemente modular [3]. Esto le permite ser portado a cualquier nuevo lenguaje de programación o a cualquier arquitectura hardware tanto existente como teórica, con solo una pequeña especificación; lo que es mucho más sencillo que desarrollar un compilador desde cero. La estructura de GCC consta de tres partes. La primera, conocida como *front-end*, toma el código en un lenguaje dado (como C) y lo transforma en un árbol intermedio de un lenguaje de transferencia de registros. La segunda parte o *middle-end* optimiza el árbol intermedio; y la tercera y última parte conocida por *back-end* [4] convierte el lenguaje intermedio en lenguaje ensamblador de la arquitectura destino.

El *porting* de GCC a una nueva arquitectura no es más que el desarrollo del código que permita adaptar el lenguaje intermedio al soportado por la arquitectura destino.

2. Motivaciones

A nivel académico y profesional encontramos principalmente dos compañías que fabrican microcontroladores, Atmel y Microchip. GCC está ya portado para Atmel y con bibliotecas de dispositivos probadas y fiables lo que permite crear proyectos, en un entorno libre, para la gran mayoría de sus microcontroladores. Para Microchip solo existía una alternativa libre, SDCC (*Small Device C Compiler*), que no tiene mantenimiento tangible y no cumple con unos objetivos mínimos de rendimiento y completitud del código C admitido. Por lo tanto la motivación era obvia, hacía falta crear un compilador C completo, correcto y libre para los microcontroladores de Microchip.

Hay dos beneficios fundamentales al usar software libre: desarrollo y mantenimiento. Este proyecto se basa en la extensión de GCC para dar soporte a los microcontroladores PIC. Por ello el desarrollo es drásticamente más corto que comenzar a crear un compilador de C desde la nada. En menos de un año hemos conseguido ampliar GCC, o lo que es lo mismo crear un compilador de C para PIC. Hacer un compilador desde la nada hubiese consumido más del doble y con la mitad de calidad. El segundo beneficio es el mantenimiento. Ningún software privativo puede competir con el mantenimiento realizado por una innumerable cantidad de usuarios repartidos por el globo. Desde el que simplemente crea tutoriales de uso o reporta un fallo, hasta el que amplía el software para soportar nuevas funcionalidades, todos ayudan porque quieren, voluntariamente. En este sentido principalmente, la comunidad de software libre constituye el equipo de trabajo más perfecto existente actualmente en el mundo. Aquel que trabaja es porque le gusta su trabajo.

3. Objetivos

El objetivo del proyecto fue inicialmente menos ambicioso que el existente en la actualidad. Comenzó con la idea relativamente simple de crear un compilador de C para el microcontrolador PIC16F877 sin aritmética de punteros. Tras las primeras etapas de desarrollo, este objetivo se hizo bastante más exigente al decidir que se debía incluir el manejo de punteros. Los punteros, siempre que sean bien usados, permiten trabajar con grandes cantidades de memoria con un mínimo gasto en el tratamiento de la misma.

Suponen una pieza clave para el programador de C, sobre todo en dispositivos con memoria tan reducida como los PIC. Por lo tanto, pese a ampliar considerablemente la dificultad del desarrollo, eran una pieza golosa del software a desarrollar.

La segunda ampliación de los objetivos iniciales suponía un reto aún mayor que la inclusión de los punteros. Decidimos en su momento no solo conseguir un compilador para el microcontrolador PIC16F877, sino sentar las bases que permitieran generar código para toda la familia de microcon-

Pedro José (izquierda) junto con David Santo el dia de la presentación de su Proyecto Fin de Carrera.

troladores PIC16F. La diversidad de características en los mismos implicaba un rediseño a nivel interno del trabajo realizado hasta la fecha; además de la parametrización de todos los valores diferenciables de los distintos modelos. Tras estas dos ampliaciones el objetivo actual es crear un compilador de C completo para la gama de microcontroladores PIC16F. Como veremos, el objetivo está más que cumplido.

4. Resultados

Actualmente se puede compilar código para todos los microcontroladores PIC16 de forma directa. En el código generado se comprueba el acierto en la elección del compilador GCC como base del desarrollo. Las optimizaciones realizadas por GCC, la mayoría en un principio extrañas, consiguen generar código extremadamente óptimo. Reducen el perjuicio que ha sido siempre atribuido a los programas en C frente a los programas en ensamblador, en concepto de lentitud y gasto de memoria. Ésto unido al beneficio de la reutilización de código y la simplicidad de la programación en C consiguen que la programación de microcontroladores PIC en lenguaje C en un entorno totalmente libre sea un hecho.

5. Próximos pasos

Se puede ya generar código para todos los microcontroladores del tipo PIC16. Sin embargo tras afianzar el funcionamiento del compilador, los futuros pasos serán la inclusión del resto de familias de microcontroladores de Microchip. Antes se deben adaptar ciertas funciones de las bibliotecas de C para conseguir una mejor optimización y soporte.

Pese a estar totalmente fuera de la filosofía del proyecto, se realizará un paso más o un alto en el camino para el desarrollo de las bibliotecas de dispositivos. Las bibliotecas de proyectos similares para otros microcontroladores siempre han pecado de precipitadas. En una biblioteca no debe haber cambios de cara a la usabilidad entre distintas versiones. Por tanto el primer paso debe ser sentar, clara y definitivamente, la forma de las funciones y rutinas que constituirán las bibliotecas y su funcionamiento. Para conseguirlo se debe crear un modelo inicial con la documentación y exponerlo a votación por la comunidad de usuarios. Una vez acordadas las rutinas y su funcionamiento es cuando se tiene que desarrollar el código de la bibliotecas y no antes. Por tanto el desarrollo de las bibliotecas de dispositivos realmente consisten en un proyecto grande y independiente al compilador pero, lo suficientemente entrelazado con éste, para que sea muy conveniente su desarrollo.

En un futuro menos próximo, se planea realizar el *porting* de GCC a los PIC18. Estos microcon-troladores son más próximos al modelo de procesador innato en la filosofía de GCC por lo que, unido a la experiencia ganada durante este proyecto, facilitarán en mucho la etapa de desarrollo inicial. Con las rebajas en el precio, la mejora del rendimiento y la inclusión de un puerto USB en algunos modelos, la familia PIC18 gana adeptos día a día.

Más adelante en el tiempo, podremos centrarnos en los PIC de gama alta. Para éstos, Microchip ha desarrollado un compilador también basado en GCC y por tanto GPL. El trabajo para soportar los mismos en un entorno libre se traspasaría a la creación de los ensambladores y enlazadores para el código ensamblador generado.

6. Conclusión

En la actualidad el Porting de GCC a PIC compila código para toda la familia de microcontroladores PIC16 [5]. Además planeamos activamente la ampliación del proyecto a otras familias. Tenemos el código liberado en la forja del concurso y en cuanto tengamos el código completamente documentado en inglés lo liberaremos también en *sourceforge*. Ello implicará a su vez tener buenas posibilidades de que sea aceptado en la próxima versión de GCC. Técnicamente, hemos tenido que resolver muchos problemas en cuanto al desarrollo. Estamos planeando publicar varios artículos al respecto, y hemos sido invitados por ***Novática*** para exponer en un próximo artículo esas dificultades técnicas que hemos conseguido resolver.

Referencias

[1] Microchip. Página principal de Microchip. <http://www.microchip.com/>.

[2] GCC, the GNU Compiler Collection <http://gcc.gnu.org./>.

[3] D. Santo Orcero. Arquitectura interna de gcc toolchain. *Mundo Linux, 78*:58-63, 2005.

[4] D. Santo Orcero. Generación de código ensamblador en la gcc toolchain. *Mundo Linux, 79*:56-61, 2005.

[5] P.J. Ramírez Gutiérrez. Página principal del proyecto Porting de GCC a PIC16F877. <http://www.pjmicrocontroladores.es>.

Entrevista

¿Cómo y cuando surgió la idea de tu proyecto?
En las primeras conversaciones con mi orientador del proyecto fin de carrera, David Santo Orcero, le expuse mis intereses y expectativas de futuro. A partir de esa información David me planteó la idea de un compilador C para PIC16F y así nació el proyecto.

¿Y la de tu presentación al concurso?
David Santo me avisó del concurso y vista la coincidencia de la fecha de inicio era un paso lógico. En el peor de los casos no perdía nada.

¿En qué medida la presentación al concurso te ha permitido avanzar en tu proyecto?
Poco o nada. Realmente el proyecto es muy especializado y no he tenido ninguna ayuda externa. El concurso le ha dado publicidad y he conseguido ser el ganador, pero no ha cambiado los planes que existían sobre el desarrollo.

¿En qué han mejorado tu formación y tus capacidades durante el desarrollo del concurso?
No iba a utilizar una forja externa, ni a escribir artículos tan complejos en el blog. El ser participante me ha exigido más trabajo, pero la práctica del mismo me ha dado más experiencia a nivel de desarrollo en proyectos de software libre.

¿Cuáles prevés que van a ser los próximos pasos en el desarrollo de tu proyecto?
El proyecto ha crecido desde el concurso de forma que ahora se cubren todos los microcontroladores PIC16 y no solo unos cuantos. Hasta aquí era donde debía crecer este proyecto. Estamos David y yo, estructurando el porting de GCC completo a todos los microcontroladores de Microchip. Para ello, se crearán subproyectos que serán desarrollados en grupos de software libre o como proyectos fin de carrera. El objetivo es crear entornos completos de desarrollo totalmente libres.

¿Has podido compaginar el trabajo en tu proyecto con tus estudios universitarios?
Ha sido un sobreesfuerzo. Junto a las atenciones propias del concurso debía crear paralelamente el texto del proyecto fin de carrera y cumplir con las obligaciones de las diversas becas que he realizado. No ha sido fácil pero, ¿Qué mejor experiencia?

¿Hasta qué punto crees que lo aprendido en la Universidad te ha ayudado en la realización de tu proyecto?
La universidad da una base para desempeñar el trabajo propio de un ingeniero. A raíz de ese conocimiento todo es posible. Durante los años que he estado en la universidad uno de mis puntos preferidos y por el que me he orientado es la programación a muy bajo nivel. Esto es lo que me ha ayudado a hacer un proyecto tan complejo en un tiempo tan corto.

¿Has obtenido la ayuda de algún profesor o de otros compañeros?
David Santo ha realizado excelentemente su labor. Como director del proyecto y orientador del mismo, me ha guiado en todo el proceso de desarrollo. No ha hecho ni una línea de código pera ha establecido los pasos y consejos necesarios en el instante justo para mantener vivos los avances del proyecto.

¿Hasta qué punto se usa el Software Libre en las Universidades y cómo ves la evolución en este sentido?
En mi experiencia he de dar un cero a la universidad. Me costó trabajo encontrar un profesor que me ofreciera un proyecto fin de carrera con software libre. Y a este profesor me consta que le cuesta defender el software libre (aunque últimamente eso está cambiando). Durante todos los años y en todas las prácticas se ha usado software privativo (una o dos excepciones y por requerimientos de hardware) y mayoritario de Microsoft para más inri. Para casi todas las prácticas existían alternativas libres que son mejores para el aprendizaje y que hubieran evitado un gran despilfarro en licencias que se podrían haber traducido en sueldos o becas para técnicos de soporte.

A la luz de tu experiencia, ¿Qué cambiarías del sistema de enseñanza universitario actual?
A medida que la carrera avanza, se observa cierta especialización hacia un tipo particular de desarrollo de software. En mi opinión, éste no es ni el único tipo de desarrollo que debe conocer un ingeniero, ni el más adecuado. El cambio necesitaría ser progresivo y muy bien estudiado.

¿Cómo vislumbras en este momento tu futuro profesional?
Mi presente profesional (ya he acabado la carrera) es trabajar. Las barreras principales son la experiencia y los idiomas.

¿Te preocupa el intrusismo profesional en la Informática?
Es un problema grande. Sin embargo el tener el título tampoco asegura un buen trabajo. Cuando un simple aficionado a la informática con poca base teórica afronta un desarrollo demasiado grande, podrá a lo mejor entregarlo con poco o ningún retraso, pero los fallos introducidos minan a la larga (y a corto plazo) la supervivencia del sistema creado. Lo que se ahorran los clientes al principio, lo pagarán en ingenieros que deben reparar el sistema y en tiempo perdido. Lo malo es que los clientes seguirán llamando a unos y a otros "informáticos" y la fama de ambos caerá aún más.

¿Cómo ves el futuro del software libre y su aplicación a nivel empresarial?
Es el paso lógico pero la lógica no suele traducirse en realidad. Supongo que en empresas de bajo o medio tamaño seguirán usando habitualmente software privativo y que las empresas “grandes” mudarán a software libre si consiguen romper las cadenas de los formatos privativos.

¿Cómo ves la evolución del desarrollo tecnológico en España?
La impresión que tengo es que la evolución tecnológica en España es muy inerte. Surgen proyectos, mueren otros, pero no dan la sensación de crecimiento sino simplemente de mantenimiento e importación. Siempre nos mantenemos detrás en desarrollo comparados con Europa y a años luz de distancia de Estados Unidos o Japón. Como ejemplo, simplemente las conexiones ADSL, o más fácil, la televisión digital que se va a implantar, realmente vergonzosa.

¿Qué piensas de la compartición de experiencias con terceros y de la integración de los profesionales bajo estos objetivos, en asociaciones de informáticos como por ejemplo ATI?
Puede ser buena. Sin embargo hasta que los ingenieros en informática no tengan colegio oficial no conseguiremos una estabilidad. En el caso de ATI, le falta publicidad en las universidades y claridad sobre lo que realmente hacen. A fin de cuentas los universitarios son los futuros profesionales.

Muchas gracias Pedro José. Muchos ánimos para seguir adelante y recuerda que en *Novática* estamos interesados en tu artículo sobre la resolución técnica de tu proyecto.