IEEE-UNED

BOLETÍN ELECTRÓNICO

RAMA DE ESTUDIANTES IEEE-UNED

30-ENERO-2006 (BOLETÍN Nº4)

RAMA DE ESTUDIANTES IEEE-UNED
30-ENERO-2006

COORDINADOR Y EDITOR:
Alejandro Díaz (adiazh@ieee.org)

REVISIÓN:
Manuel Castro
Eugenio López
Alejandro Díaz

DISEÑO PORTADA:
Ignacio García

AUTORES
Alejandro Díaz, Francisco García Sevilla, Eugenio López Aldea, Ignacio García Caro, Manuel Castro, Elio Sancristobal.

EN COLABORACIÓN CON EL CAPÍTULO ESPAÑOL DEL IEEE EDUCATION SOCIETY

AGRADECIMIENTOS

"Agradecemos a nuestro Catedrático de Tecnología Electrónica y profesor consejero de la Rama, Manuel Castro, todo el tiempo y la dedicación que nos presta, así como, el habernos dado la posibilidad de colaborar con el Capítulo Español del IEEE Education Society para la elaboración del mismo. Agradecemos a todos los autores y a aquellos que han colaborado para hacer posible este Boletín Electrónico".

ÍNDICE

SUMARIO

Para comenzar el boletín electrónico nº 4, se presenta como en ediciones anteriores un primer apartado de **Información** general de la Rama y **URLs** de interés general propuestas por los miembros con comentarios.

En el primer artículo,**"Visita del presidente del IEEE a la Rama de la UNED"**, escrito por nuestro secretario de la rama *Elio Sancristoba*l, se resume la afortunada visita del presidente del IEEE Dr. W. Cleon Anderson, durante la reunión de la rama realizada el 12 de Diciembre.

A continuación se expone un artículo **"El IEEE y las Ramas de Estudiantes en España"** por *Alejandro Díaz Hortelano,* miembro del IEEE y coordinador del boletín Electrónico*,* el cuál servirá para refrescar algunos hechos del IEEE, así como acercar a los lectores sobre las actividades de las diferentes ramas existentes en España.

Posteriormente, nuestro profesor y consejero, *Manuel Castro*, nos muestra un artículo de gran interés general, **"Necesidades Específicas del Estudiante de Ingeniería de Cara a su Incorporación Profesional"**. En él, se habla de algunas de las capacidades no técnicas que requieren los ingenieros hoy en día para el desarrollo de una exitosa carrera profesional, y que en los próximos años los nuevos programas de formación comenzarán a cubrir.

El siguiente artículo, escrito por *Francisco García Sevilla,* es "**Sistemas de monitorización, almacenamiento y tratamientos de datos".** En él, se nos explica la aplicación real de esta clase de sistemas en un hospital para servicio de reanimación. Comprobando que los avances en la electrónica e informática repercuten en todas las actividades humanas. La medicina es un mero ejemplo, donde se puede observar la mejora de los sistemas de control y seguimiento de pacientes a través de la utilización de éstos avances.

Más adelante, *Eugenio López*, presidente de la Rama del IEEE-UNED, nos habla de la importancia hoy en día de conocer las diferencias entre las distintas posibilidades que existen para programar aplicaciones online en su interesante artículo **"Diferencias entre Java, JavaScript, JSP y ASP".**

En el último articulo realizado por *Ignacio García-Caro,* se nos explica la herramienta denominada UML (Lenguaje Unificado de Modelado). La cual consiste en una serie de símbolos y diagramas que permiten a los creadores generar diseños que capturen la idea, haciendo más fácil la tarea del desarrollo al programador. Ignacio, en el articulo **"Diseño técnico con UML",** comparte los amplios conocimientos desarrollados en el mundo de la empresa.

INFORMACIÓN Y URLS

En esta sección se pretende dar información general de la Rama y URLs de interés general propuestas por los miembros.

Tras finalizar el pasado curso con gran satisfacción por la consolidación de la rama de estudiantes de IEEE-UNED, entramos en este nuevo curso (2006-2007) con energías renovadas y la responsabilidad por parte de todos los miembros de mejorar el buen trabajo realizado durante el pasado año por toda la junta directiva de la rama. Intentando poco a poco de mejorar lo máximo posible todas las actividades ya consolidadas como son el boletín, y de iniciar otras como puedan ser seminarios, cursos, la plataforma aLF, etc. Eso sí intentando al igual que durante el año anterior que nuevos miembros se afilien y de esta forma ir creciendo en número y en actividades. La clave del éxito de la Rama de la UNED, al igual que el resto de Ramas de todo el mundo, es el voluntariado. Por tanto, agradecer a todas las personas que hacen esto posible, y que sin su ayuda no se hubiera podido llevar a cabo. Agradecer a todos los miembros que están en la Rama, que han decidido formar parte de ella y agradecer a todos los voluntarios que han colaborado en las actividades realizándolas y llevándolas a cabo, así como, a todos los autores de los artículos.

Parte de la idea de ofrecer una diversidad cultural diferente entre los estudiantes donde nosotros mismos somos los que dedicamos los esfuerzos voluntarios en pro del bien común en cuanto a conocimientos, contactos y la posibilidad de compartir actividades técnicas, científicas y tecnológicas.

La Rama se consigue consolidar inicialmente con 37 miembros en noviembre del año 2004.

La información general sobre sus actividades e información de cómo hacerse miembro la hemos colocado en la página Web: www.ieec.uned.es/IEEE dentro del enlace de la Rama de Estudiantes.

Las actividades principales que se pretenden realizar son: charlas, cursos, congresos, concursos, actividades educativas, visitas a empresas y organizaciones, interrelación cultural y multidisciplinar y cualquier actividad que quiera desarrollar cada uno de sus miembros.

Actualmente puede participar cualquier estudiante de las carreras de Ingeniería Informática y de Ingeniería Industrial de la UNED.

Las actividades realizadas en el año 2005 se resumen a continuación:

- Creación de los Boletines nº 2 y 3 con interesantes artículos y experiencias de diferentes miembros de la Rama y el actual Boletín nº 4.

- Contacto con Microsoft para realización de actividades. Colaboran en el boletín nº 3. Contacto con Accenture a través de un miembro de la Rama para tratar de realizar una actividad análoga a la realizada con DMR para fomentar la relación universidad-empresa (pendiente).

- Creación de varios comités dentro de la Rama: Comité de Socios y Bienvenida (coordinador Francisco García Sevilla), Comité de actividades (Coordinador Elio Sancristobal) y comité para el Boletín Electrónico (Coordinador Alejandro Díaz)

- Realización de una presentación en Director para facilitar la forma de hacerse socio por Internet de las personas interesadas. Se establece contacto con Ricardo Varela (representante de alumnos de la Región 8) que se interesa por la aplicación para tratar de establecer un proyecto en la línea de la ayuda a hacerse miembro a las personas interesadas.

- Proyecto UNITeS: Seguimos impulsando el proyecto en colaboración con la Universidad.

- Curso de Robots. Realizado con la Universidad Alfonso X el Sabio en Abril. Participan varias personas de nuestra Rama.

- Se ha establecido el contacto con otras Ramas de manera que nos apoyamos unas a otras en la realización de actividades conjuntas.

- Estamos apuntados para participar en el CNR 2005 que se celebrará en Valencia.

A continuación en URLs de interés, se expone un **e-mai**l enviado a los alumnos de la UNED en el que nos dice que tenemos la oportunidad de **hacer cursos de Sun Microsystems (que son de pago) de forma gratuita por ser de la UNED**:

Estimado alumn@,

Sun MicroSystems ofrece mediante su "The Sun Academic Initiative Program" una gama interesante de cursos a coste cero para instituciones educativas.

La UNED se ha inscrito en esta iniciativa de tal manera que sus alumnos pueden disfrutar de estos cursos en línea. A continuación se remite la lista de cursos y las instrucciones para acceder a ellos.

El portal de entrada es: https://learningcenter-sai.sun.com/
El código corporativo de la UNED es: (Quien esté interesado en realizar un curso que solicite la clave a elopez@ieec.uned.es)
(este es el "Company I.D." a consignar en el formulario de registro)

Nota importante: el sistema no soporta los caracteres especiales como ä, æ, ç, ñ, ó, @ etc.

Nótese que durante el proceso de selección de cursos se verán los precios de los que se seleccionen; no asustarse, al finalizar el proceso se aplica un descuento del 100% con lo que el resultado es cero dólares.

LISTA DE CURSOS QUE OFRECE LA PÁGINA
==========

Java[tm] Technology
Fundamentals of the Java[tm] Programming Language
Java[tm] Programming Language
Java[tm] Practice Certification Exam
Object Oriented Programming with Java[tm] Technology
Web Component Development with Java[tm] Technology
Distributed Programming with Java[tm] Technology
Java[tm] Servlets: A Technical Introduction
Java[tm] 2 Platform, Enterprise Edition (J2EE) Platform Overview for Managers
Mobile Desktop Development with Java[tm] Technologies
Consumer Devices Learning Suite
Programming with the Java[tm] 3D API: A Technical Overview
J2SE Internals and Troubleshooting
Java[tm] 2 Platform, Enterprise Edition (J2EE Platform) Technology Overview Sampler
Working with the Java[tm] 2 Platform, Micro Edition Wireless Toolkit 2.0

Networking and Security
Web Server and Security
Directory Services

Server and Storage Systems
Server and Storage Systems

Solaris[tm] 8 Operating Environment
Solaris[tm] 8 System Administration I
Solaris[tm] 8 System Administration II
Solaris[tm] 8 Operating Environment TCP/IP Network Administration
Fundamentals of Solaris[tm] 8 Operating Environment

Solaris[tm] 9 Operating Environment
UNIX Essentials Featuring the Solaris[tm] 9 Operating Environment
Intermediate System Administration for the Solaris[tm] 9 Operating Environment
Advanced System Administration for the Solaris[tm] 9 Operating Environment
Network Administration for the Solaris[tm] 9 Operating Environment
Solaris[tm] 9 Practice Certification Exam
New Features of the Solaris[tm] 9 Operating Environment

StarOffice[tm] Software
StarOffice[tm] 6.0 End User
StarOffice[tm] 7.0 End User
Getting Started with Sun[tm] Jave Desktop

Sun[tm] ONE Middleware

Identify Management Services
Web & Application Services
Collaboration and Communication Services

Sun[tm] ONE Studio and Solaris[tm] OE Development
Sun[tm] ONE Studio 4 for Java
Real-Time Programming for the Solaris[tm] Operating Environment

Sun[tm] One Middleware
Communications Services
Portal Services
Web and Application Services
Wireless Technologies

XML Perl and Web Publishing
A Developers Introduction to Web Programming
A Developers Introduction to Java Script Programming
A Developers Introduction to PERL Programming
A Developers Introduction to Advanced PERL Programming
A Developers Introduction to HTML Programming
A Developers Introduction to Web Processing

A continuación se muestra el listado de los congresos que serán realizados a lo largo del año por el IEEE.

2006 2nd International Conf. on Testbeds and Research Infrastructures for the Development of Networks & Communities (Tridentcom)	01 Mar - 03 Mar 2006
2006 2nd Conference on Next Generation Internet Design and Engineering (EURO-NGI)	03 Apr - 05 Apr 2006
INFOCOM 2006	23 Apr - 29 Apr 2006
MELECON 2006 - 2006 IEEE Mediterranean Electrotechnical Conference	16 May - 19 May 2006
2006 IEEE Symposium on Virtual Environments, Human-Computer Interfaces and Measurement Systems (VECIMS)	10 Jul - 12 Jul 2006
2006 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications (CIMSA)	12 Jul - 14 Jul 2006
2006 International Symposium on Electromagnetic Compatibility - EMC EUROPE	04 Sep - 08 Sep 2006

2006 International Workshop on Satellite and Space Communications (IWSSC)	14 Sep - 15 Sep 2006
2006 IEEE International Conference on Cluster Computing (CLUSTER)	25 Sep - 28 Sep 20

Otra dirección de interés para todos los miembros, es http://www.ieee-r8sac.org/sbc2006/.

En ella se puede consultar información sobre el SBC 2006 ("*Student Branch Congress*"), es decir el congreso de ramas de estudiantes de toda la región 8. Celebrado cada 2 años, y el cual este año se celebrará en la ciudad de París del 31 de Agosto al 3 de Septiembre. Cualquier interesado en asistir se deberá poner en contacto con la junta directiva de la rama.

También puede resultar atractivo para aquellos que estén interesados por la innovación educativa en la formación de ingenieros en Electrónica, el consultar la siguiente dirección
http://www.euitt.upm.es/taee06/,
donde se puede consultar información sobre el *TAEE 2006* (Tecnologías Aplicadas a la Enseñanza de la Electrónica).

La Web del IEEE en la UNED es:
http://www.ieec.uned.es/IEEE/

donde podéis encontrar información sobre qué es el IEEE y la Rama de estudiantes, cómo hacerse miembro, ver información sobre otras Ramas y una sección de eventos y actividades donde se irán colocando las actividades que se van a ir realizando y que se han realizado. **Todos las personas interesadas en saber más información o en hacerse miembro, escribir un mail a:**

elopez@ieec.uned.es

PROMOCIÓN DE LA DIRECTIVA PROPUESTA DE LA RAMA DE ESTUDIANTES IEEE-UNED PARA EL AÑO 2006

Eugenio López. Presidente de la Rama de Estudiantes del IEEE-UNED. Estudiante de Doctorado en el DIEEC. Ingeniero Industrial. elopez@ieec.uned.es

Ignacio García. Vicepresidente de la Rama de Estudiantes del IEEE-UNED. Estudiante de Ingeniería Industrial. nachogcg@hotmail.com

Elio Sancristobal. Secretario de la Rama de Estudiantes del IEEE-UNED. Estudiante de Doctorado en el DIEEC. Ingeniero Informático. esancr@yahoo.es

Javier García. Tesorero de la Rama de Estudiantes del IEEE-UNED. Estudiante de Ingeniería Industrial. garciagimenez@hotmail.com

Manuel Castro. Catedrático de Tecnología Electrónica. Profesor Consejero de la Rama de Estudiantes del IEEE-UNED. Miembro Señor del IEEE y actual presidente del capítulo Español de la IEEE Education Society recién creada en España. mcastro@ieec.uned.es

Alejandro Díaz. Coordinador del Boletín Electrónico de la Rama de Estudiantes del IEEE-UNED. Ingeniero Industrial por la UNED.adiazh@ieee.org

Durante el año 2006 el Comité de socios y bienvenida, será llevado por la totalidad de la junta directiva.

VISITA DEL PRESIDENTE DEL IEEE A LA RAMA DE LA UNED

El día 12 de Diciembre en el salón de actos de la Facultad de Económicas y Empresariales de la UNED tuvo lugar la última reunión, del año 2005, de la Rama de estudiantes del IEEE de la UNED a la que se pudo asistir de forma presencial o vía Web. En dicha reunión tuvimos el honor de contar con la presencia del Dr. W. Cleon Anderson, presidente del IEEE, que nos hablo sobre las ventajas que ofrece el IEEE para un estudiante o ingeniero a la hora de desarrollar su carrera profesional.

En este artículo se resumen los principales temas tratados por el Dr. Cleon. Recordar que la charla completa se puede bajar de la página de eventos de la Rama del IEEE de la UNED:

http://www.ieec.uned.es/investigacion/eventos_ieee/eventos.htm

1. EL IEEE

No es solo una institución americana, ni una institución para los ingenieros eléctricos y electrónicos. El IEEE pretende ser y es una institución global donde se tratan una gran diversidad de temas: industriales, informáticos, telecomunicaciones, etc.

Así se puede decir, entre otras cosas, que el IEEE:

- Es uno de los más importantes editores de publicaciones técnicas en el mundo.
- Patrocina un gran número de conferencias técnicas a nivel mundial.
- Es una fuente de estándares internacionales.
- Ayuda a los estudiantes de ingeniería a entender y dominar conceptos de ingeniería que pueden resultarles difíciles de entender.
- Provee de material educativo a los estudiantes e ingenieros que les facilite el desarrollo de su actividad y la consolidación de nuevos conceptos.

Por tanto se puede afirmar que la unión de los diferentes miembros del IEEE pretende:

- Fomentar la innovación tecnológica.
- Permitir el crecimiento profesional de sus miembros.
- Promover la colaboración de los miembros a nivel mundial.

2. BENEFICIOS QUE IEEE OFRECE A SUS MIEMBROS

Se ha analizado anteriormente de forma breve que es el IEEE y cuales son sus objetivos. A continuación se comentan algunos de los beneficios que el IEEE ofrece a sus miembros.

- Permite acceder a información técnica a través de la plataforma IEEE Xplore.

- Los miembros pueden acceder a una gran cantidad de información técnica debido a la existencia un gran número de publicaciones en línea, artículos, manuales, etc. También los miembros pueden acceder vía Web a cursos on-line que les permitan iniciarse o desarrollar temas de actualidad.

- Cada miembro puede disponer de una cuenta de correo electrónico con protección de antivirus y de antispam.

- Existen comunidades en línea donde los miembros del IEEE pueden colaborar e intercambiar experiencias y conocimiento.

3. COMO ENFRENTARSE AL MUNDO REAL

El ingeniero a lo largo de su vida se ve obligado a moverse de empresa debido a motivaciones intelectuales, económicas, etc., y a realizar distintos trabajos en distintos campos de su actividad.

Es importante que el ingeniero en su etapa de estudiante haya adquirido un aprendizaje básico y un manejo adecuado de las herramientas que va a utilizar para desarrollar su trabajo. Estos conceptos básicos se deben ir incrementando a lo largo de su carrera profesional, además de probarse a si mismo que es capaz de aplicar dichos conocimientos.

Por tanto es importante que el ingeniero tenga en cuenta los siguientes elementos en una carrera de ingeniería:

- Educación: Desarrollo de las habilidades y de las herramientas necesarias para desarrollar su trabajo.

- Experiencia: Le permitirá basarse en soluciones exitosas que aplico a problemas, para resolver nuevos problemas que son similares. También debe ser capaz de valorarse, para que le valoren en su empresa.

- Colaboración: No se puede tener éxito de forma independiente. Es importante colaborar y trabajar con personas para obtener objetivos a nivel global y a nivel personal. El IEEE facilita dicha colaboración.

También debe tener en cuenta a la hora de su desarrollo profesional los conceptos de: fuerzas, debilidades, oportunidades y amenazas.

4. CONCLUSIÓN

El IEEE por tanto es una institución mundial cuyos objetivos son: fomentar la innovación tecnológica y permitir el crecimiento profesional de sus miembros mediante la formación y colaboración de estos, para ello se ofrecen un gran número de publicaciones en línea y se promueven conferencias a nivel mundial.

La ingeniería esta en continuo avance y por tanto los ingenieros deben formarse continuamente. Además deben saber posicionarse en una buena situación para poder optar posteriormente a posibles oportunidades importantes para su profesión, tanto a nivel intelectual, profesional, económico, etc.

El IEEE pretende ayudar a sus miembros a obtener los conocimientos y habilidades necesarias para mejorar y así mismo ir creciendo como institución.

Os recordamos de nuevo que podéis seguir en diferido la charla completa del Dr. Anderson a través de la página de eventos de la Rama del IEEE de la UNED:

http://www.ieec.uned.es/investigacion/eventos_ieee/eventos.htm

Por último animar a los miembros a seguir colaborando de manera activa para que la rama del IEEE de la UNED sea un lugar donde poder exponer y obtener conocimiento útil para nuestros estudios y actividad profesional. También indicar a todas aquellas personas que deseen formar parte de la rama que pueden obtener información en la siguiente dirección Web:

http://www.ieec.uned.es/IEEE/

Elio Sancristobal
Secretario
Rama de Estudiantes de la UNED del IEEE

EL IEEE Y LAS RAMAS DE ESTUDIANTES EN ESPAÑA

¿Qué es el IEEE?

Se trata de una asociación Americana, con sede en New York, sin ánimo de lucro, y con más de 365000 miembros repartidos a lo largo de 150 países. De los cuales, más del 40 por ciento provienen de fuera de los Estados Unidos. Las siglas IEEE, provienen de Instituto de Ingenieros Eléctricos y Electrónicos (Institute of Electrical and Electronics Engineers, Inc). Cuyo objetivo primordial es la divulgación de la *ciencia y la tecnología*.

Gracias a la ayuda de innumerables profesionales (ingenieros, matemáticos, físicos, etc.), y a futuros ingenieros que colaboran a través de diversas ramas de estudiantes (Student Branch), el IEEE se convierte en una institución líder en áreas tecnológicas, como puedan ser, la informática, la electrónica, las telecomunicaciones, o la ingeniería aeroespacial. El IEEE produce alrededor del 30 por ciento de las publicaciones en Ingeniería Electrónica y Eléctrica en todo el mundo, y realiza más de 300 conferencias anualmente. Además, posee aproximadamente cerca de 900 estándares activos, como por ejemplo el IEEE 811.2 (Wi-Fi), y más de 500 en desarrollo.

El último cuarto del siglo XIX fue prolífero en el desarrollo de tecnología eléctrica y de las disciplinas que de ella derivan. En 1884, el crecimiento en la tecnología, y la concentración de algunos de los mejores Ingenieros eléctricos de la época en la “Internacional Electrical Exhibition” en Philadelphia, dio lugar a la aparición de AIEE, “American Institute of Electrical Engineering”. La fusión del AIEE y el IRE, “Institute of Radio Engineers”, en 1963 provocó la creación de la institución conocida como IEEE.

El IEEE se divide en 10 regiones que abarcan todo el mundo. Concretamente, la región 8 es a la que pertenece nuestra rama y el resto de ramas de España, y abarca Europa, África y Oriente Próximo.

Respecto a la actividad técnica, la asociación consta de 9 Divisiones.

Division I - Circuits and Devices
Division II - Industrial Applications
Division III - Communications Technology
Division IV - Electromagnetics and Radiation
Division V - Computer
Division VI - Engineering and Human Society
Division VII - Energy and Power Engineering
Division VIII - Signals and Applications
Division IX - Systems and Control

Del IEEE dependen más de 40 Sociedades Técnicas (“Technical Societies”), que son grupos interesados en determinados aspectos concretos de la tecnología, como pueden ser la Robótica, el electro óptico, la medicina, la ingeniería aeroespacial, etc.

Algunas de las sociedades más destacadas de las 39 disponibles son:

IEEE Aerospace and Electronic Systems Society
IEEE Circuits and Systems Society
IEEE Communications Society
IEEE Computer Society
IEEE Control Systems Society
IEEE Electron Devices Society
IEEE Education Society
IEEE Engineering in Medicine and Biology Society
IEEE Industrial Electronics Society
IEEE Industry Applications Society
IEEE Power Electronics Society
IEEE Power Engineering Society
IEEE Robotics & Automation Society
IEEE Signal Processing Society
IEEE Solid-State Circuits Society

¿Qué son las ramas de Estudiantes?

Las ramas de estudiantes son asociaciones de alumnos de Escuelas Técnicas, cuyo objetivo es la divulgación de la ciencia y tecnología, y donde se realizan todas aquellas actividades, cursos, conferencias, etc., que no suelen estudiarse en las clases ordinarias y que complementan su formación. Todas las actividades desarrolladas en las ramas, son elegidas y realizadas por los propios miembros. Es decir, las ramas deciden que actividades realizar en función de las inquietudes de sus miembros. Actualmente, existen más de 1300 ramas de estudiantes, repartidas en 80 países. Además, de los 365000 miembros del IEEE aproximadamente 68000 son miembros estudiantes. Y de los 50000 miembros de la Región 8, más de 10000 son estudiantes.

Para poder ser miembro de una rama de estudiantes, el interesado ha de hacerse miembro del IEEE. La suscripción al IEEE como miembro estudiante, únicamente cuesta 25$, frente a los 124$ de los miembros no estudiantes, y tienen derecho a los mismos privilegios que los miembros normales.

Para hacerse miembro puede dirigirse a las siguientes direcciones:

http://services1.ieee.org/membersvc/member/mem_intro.htm

http://www.ieec.uned.es/ES/index_IEEE.htm

¿Cuales son los beneficios de ser miembro del IEEE?

- Supone un realce para el currículum de cada uno.
- Acceso a valiosísimas fuentes de información.
- Recibir mensualmente la revista IEEE Spectrum de divulgación tecnológica.
- Acceso a los servicios on-line que brinda el IEEE.
- Recepción mensual de diferentes boletines electrónicos sobre diversos temas, en función de los intereses de cada persona.
- Descuentos en libros, congresos, simposios, etc.
- Estar en contacto con profesionales del mismo sector en todo el mundo.
- Acceso a una de las Webs mas completas para la búsqueda de empleo cualificado, la IEEE Job Site
- Acceso al buscador Xplore de literatura técnica del IEEE
- Un alias de correo electrónico @ieee.org, sin virus y sin spam, relacionado con la organización profesional
- Acceso a becas y premios de investigación del más alto nivel internacional.
- Acceso a más de 40 Comunidades Virtuales, donde ciertos miembros con un mismo interés participan en foros de discusión y comparten información.
- Pertenecer a la más renombrada organización INTERNACIONAL de ingenieros y tecnólogos del mundo.

Para mayor información consultar:

http://www.ieec.uned.es/ieee/investigacion/ieee_dieec/sb/10%20Grandes%20razones%20ieee.pdf

¿Qué otras ramas de Estudiantes del IEEE existen en España y que actividades realizan?

Las ramas de estudiantes son delegaciones de miembros estudiantes del IEEE. Actualmente, existen unas 1000 ramas en Universidades en todo el mundo, donde se desarrollan localmente actividades técnicas y/o sociales: cursos, seminarios, conferencias, etc. Estas suelen contar con el asesoramiento de un profesor, miembro del IEEE.

Entre las ramas de estudiantes en España y las actividades que éstas realizan, se pueden citar:

- **Universidad Carlos III. Escuela Politécnica Superior de Madrid. (2000)**

Entre las principales actividades de la rama destacan la creación de dos grupos de trabajo: El grupo de robótica y el grupo del espacio.

El grupo de robótica pretende fomentar el desarrollo de microrobots y autómatas. Para ello, la universidad pone a disposición de todos los miembros, tanto a los expertos como a los recién llegados, de recursos y medios. Realizando, de forma anual, un concurso entre los miembros pertenecientes al grupo.

Por otro lado, *el grupo del espacio*, creado en Noviembre del 2003, intenta reunir a todos aquellos interesados en el espacio, en cualquiera de sus vertientes.

Se puede consultar más información a través de la página Web de la rama: http://www.ieee.uc3m/

- **Universidad de Castilla la Mancha. (2001).**

Esta rama ha creado tres grupos de trabajo para el desarrollo de actividades.

- Robótica: Grupo encargado de fabricar un robot capaz de participar en Hispabot.

- Juego de Rol: Donde se pretende desarrollar un juego de rol, basado en Dragones y Mazmorras y con arquitectura cliente/servidor. El servidor sería el programa del Dungeon Master (DM), que además tendría capacidad para desarrollar sus propias aventuras.

- Física: Programa de ayuda a la enseñanza y el aprendizaje de la Física. Resolución de los típicos problemas de vectores, tiro parabólico, dinámica, campos (gravitatorio, eléctrico y magnético), óptica, etc. Representación gráfica del problema y la solución, para una mejor comprensión.

Además, de los grupos mencionados, también realizan diversos cursos, como por ejemplo: Curso de montajes de Cable RJ45, proyección de largometrajes, curso y concurso de Robocode, exposición y curso de

robótica, conferencia: "The Women in Engineering", inclusión de JAMES en la web, exposición UMTS, experimentos de física, museo de informática, taller de malabares, concurso de videojuegos, curso LaTeX, exhibición de capoira, …

La página Web de la rama se puede consultar en:

http://arco.inf-cr.uclm.es/ieeesb/

- **Universidad de Málaga. ETSI de Telecomunicación. (1999).**

Se funda en la E.T.S.I. de Telecomunicación de Málaga en Marzo de 1999. Inicialmente 54 fueron el número de sus miembros que la integran. Por lo que sabemos, en estos momentos su número es de 103.

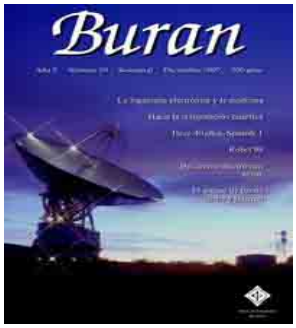
Entre algunas de sus actividades se pueden citar, por ejemplo las desarrolladas durante el curso 2002-2003.

- Concurso de Microbots UMABOT.
- Curso de Introducción al retoque fotográfico.
- Curso de introducción a 3D Studio.
- Conferencia: El cine, una perspectiva técnica.
- Conferencia: introducción a las redes Wireless.
- Conferencia: J. C. Bose y la Radiocomunicación.
- Conferencia: Medios 'zurdos'. O cómo hacer pasar un camello por el ojo de una aguja.
- Conferencia sobre Microbótica.

Su página Web se puede consultar en: http://ieee.etsit.uma.es/

- **Universidad Politécnica de Cataluña. ETSI de Telecomunicación UPC. (1979).**

La rama de estudiantes de la UPC, se trata de una de las primeras ramas fundadas en España. Esta rama, desde 1993 edita e imprime en Barcelona, la revista *Burán*, su principal actividad.



En estos momentos, ya se encuentran por el número 22 de la revista. La actividad es realizada íntegramente por estudiantes. Algunas de sus tareas son la búsqueda de artículos, animando a profesores, doctorandos, y a estudiantes que escriban interesantes artículos

Otra interesante actividad desarrollada por la rama es el *Grupo de Robótica,* donde los estudiantes pueden iniciarse en el mundo de la robótica, y participar en diferentes concursos nacionales y/o internacionales.

También, dentro de la rama existe el Grupo de Desarrollo Web. Su nacimiento es reciente, y allí se realizan diversos cursos: Flash, Java, etc.

Otros cursos que se imparten son los cursos de Robótica, y los cursos de Linux.

Su página Web se puede encontrar en: http://ieee.upc.es/

- **Universidad Alfonso X el Sabio.**

La universidad Alfonso X el Sabio, al igual que otras ramas de estudiantes se ha seccionado en diversos grupos:

- Grupo de desarrollo Web
- Grupo de Robótica
- Grupo de Programación
- Grupo de Telemática
- Grupo de Wireless

Robot PI2

Actualmente, el grupo mas activo de la rama es el de Robótica, donde ya se han realizado dos concursos.

- **Universidad Politécnica de Madrid. ETSI de Telecomunicación UPM. (1964).**

Esta rama junto a la de universidad politécnica de Valencia y Cataluña, fueron las primeras ramas de estudiantes creadas en España. Los grupos de trabajo de la rama son:

- **Grupo de Robótica**: Su principal objetivo es el desarrollo de robots móviles. Tanto para ser presentados en concursos, como para realizar investigación. Además, se busca mejorar la formación de los miembros en áreas como la electrónica y programación. En el grupo se trabaja con microcontroladores (HC11 de motorota, PIC de microchip, etc.).
- **Grupo Agassi:** Todas las actividades son desarrolladas en torno al sistema operativo Linux y las redes de ordenadores. Anualmente realizan un seminario de GNU/Linux para acercar a los estudiantes de la escuela a los procesos de instalación y configuración del sistema operativo.
- **Grupo WebFactory:** Se dedican a casi cualquier cosa que tenga una interfaz Web: HTML, CSS, JavaScript, PHP, MySQL y servidores Apache.

- **Grupo de programación en C:** Sus intereses son básicamente dos: la formación en los lenguajes C/C++ y el desarrollo de proyectos conjuntos. Actualmente se están realizando cursos de Iniciación al lenguaje C, programación de videojuegos y programación avanzada.
- **Grupo de programación en JAVA:** Su nacimiento es reciente, con el fin de que sus miembros mejoren sus conocimientos en JAVA (programación orientada a objetos). Este año se han iniciado diversos proyectos como por ejemplo un "jareador", un compresor de archivos ejecutables ".jar" mediante entornos de ejecución de java, o realizar la programación de un "risk" para jugar vía Internet a través del servidor de la escuela. Otros proyectos pendientes son la creación de un reproductor mp3 y un javaSMS.

- **Universidad de Sevilla**

La Universidad de Sevilla también ha creado diversos grupos de trabajo:

- **Taller de Microelectrónica:** Trabajan con PIC, que se trata de una mini CPU con capacidad para ser programado y con capacidad de cálculo. Los asistentes al taller aprenden a montar los programadores, hacen pruebas del estilo "hola mundo", etc. También, se están planteado construir un mando a distancia universal.
- **Women in Engineering:** Grupo de trabajo en formación.
- **Comités:** Se han creado diversos comités para que los miembros de la rama se puedan agrupar y colaborar dentro de la rama. Éstos son:

  - *Comité Informático*: Para el mantenimiento de la Página Web, el servidor, desarrollos en James, administración de la listas de correos.

  - *Comité de Relaciones*: Desde aquí, se mantienen las relaciones con otras ramas, con empresas, u otras universidades.

  - *Comité de promoción*: Su objetivo es mantener informados al resto de los miembros mediante ciertos documentos de los asuntos tratados en las reuniones. También, aclarar las ventajas de ser miembro del IEEE.

  - *Comité de Actividades*: Su misión es ayudar a gestionar y desarrollar cada una de las actividades de la rama, y hacer la máxima publicidad de éstas.

- **Ingeniería Biomédica:** Su objetivo es acercar al resto de los miembros la ingeniería biomédica, tecnología que integra las

ciencias de la ingeniería con las ciencias biomédicas y la práctica clínica.

La página Web de la rama se puede consultar en:

https://ieeesb.us.es/ieeesb/james1/modules/ieeeInterface/init.php

Alejandro Díaz Hortelano
Coordinador del boletín Electrónico de la rama IEEE-UNED
Ingeniero Industrial
adiazh@ieee.org

NECESIDADES ESPECÍFICAS DEL ESTUDIANTE DE INGENIERÍA DE CARA A SU INCORPORACIÓN PROFESIONAL

Manuel Castro
Catedrático de Tecnología Electrónica / UNED
Profesor Consejero de la Rama de Estudiantes de la UNED del IEEE
Presidente del Capítulo Español de la Sociedad de Educación del IEEE
mcastro@ieec.uned.es

En el presente artículo se van a recoger una serie de necesidades específicas del estudiante de Ingeniería en España que deberían incluirse en su formación y que son necesarias de cara a su correcta incorporación profesional, denominadas habitualmente "*non-technical skills*" (capacidades no técnicas) en la literatura anglosajona.

Contenidos en Ingeniería

Actualmente estamos inmersos en España en un proceso de reforma de las enseñanzas universitarias, orientadas al denominado Espacio Europeo de Educación Superior, EEES (*Higher Education European Space*), donde se primará la movilidad de los agentes implicados en la formación universitaria (alumnos, profesores y personal de administración y servicios) así como la personalización de la tutorización y el cambio de mentalidad de la formación, pasándose de la clase presencial y magistral a la tutorización personalizada, el trabajo en grupo y la colaboración, contabilizándose la labor docente en vez de por las horas de clase recibidas por el alumno, por las horas que el alumno dedica a su formación de forma completa (ECTS, *European Credit Transfer System*, Sistema Europeo de Transferencia de Créditos).

Entre los cambios previstos están la integración de las actuales ingenierías de primer y segundo ciclo en una única ingeniería de grado, con una posterior especialización en estudios de Postgrado o Master. Los comentarios siguientes se basan en los planes de estudio actuales, que habrán de ser revisados de cara a la convergencia europea anterior, pero que debido a su previsible reducción de contenidos y de dedicación por parte del alumno, nos tememos que las propuestas finales de incorporación de contenidos no técnicos seguirán quedando fuera de los nuevos planes de estudio renovados.

Así, actualmente los contenidos formativos en Ingeniería se pueden clasificar en Troncales, Obligatorios, Optativos y de Libre Configuración. Los primeros los define el Ministerio de Educación para la organización básica del título y del Plan de Estudios, los segundos los incorpora la Universidad para su adaptación a los objetivos de ésta, los terceros los elige el alumno (entre los ofertados por la Universidad) para su mejor especialización y los últimos, los elige el alumno como ampliación de los anteriores (igualmente, entre los ofertados) como una personalización de su currículo. Todo esto, de forma estructurada, es lo que deberá evaluar un posible empresario el día de mañana para decidir si contratar o no a un ingeniero formado en nuestras escuelas.

Entre los contenidos troncales de las Ingenierías se pueden citar las siguientes materias: Matemáticas, Álgebra, Física e Informática básica, (en cualquier rama de la ingeniería), Química, Mecánica y Materiales (en Industriales) y Tecnología y Programación (en Informática). Nos centraremos en estas ramas de la Ingeniería que son las que tienen una mayor presencia en nuestra Universidad, la UNED, y entre los miembros del IEEE (junto a Telecomunicaciones).

Dentro de los contenidos especializados, se pueden incluir estos según las especialidades, intensificaciones o las ramas de los estudios anteriores: así, en Industriales, la Electrónica y Automática, o la Eléctrica, o la Mecánica, o la Energética, Construcción, Fabricación o Diseño Industrial, y en Informática, los Sistemas o la Gestión.

Este conjunto de conocimientos, acumulados entre los 60 y 75 créditos cada año, conforman las enseñanzas de Ingeniería actuales.

Pero quedan algunas lagunas BÁSICAS y NECESARIAS para el correcto devenir futuro del estudiante en su entorno laboral, que se denominan las capacidades no técnicas (*non-technical skills*).

Las capacidades no técnicas y su relación con la profesión de la Ingeniería

La Figura 1 muestra lo que no se ve hoy en día en la carrera de un Ingeniero y DEBERÍA verse, pero que sí se evalúa (y en muchos casos de forma determinante) a la hora de entrar o progresar en la vida laboral y profesional de un ingeniero.

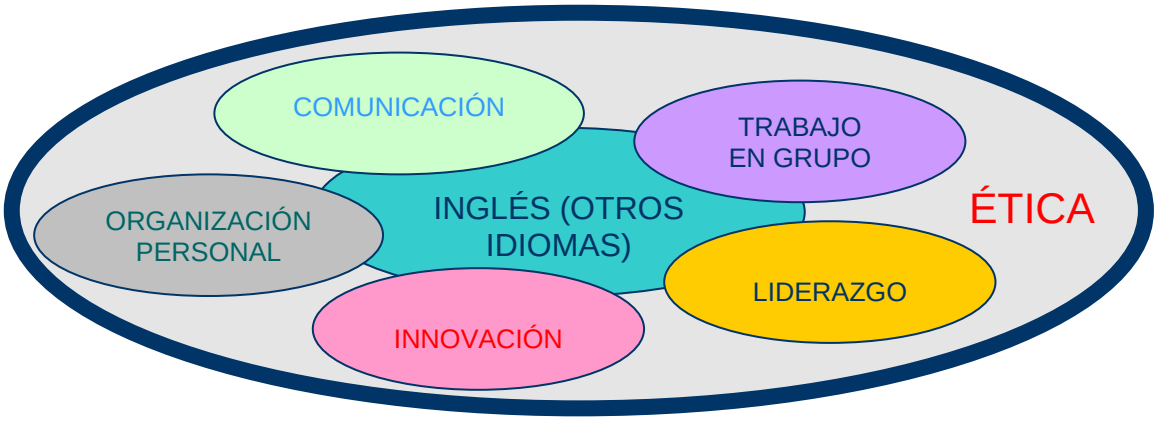


Figura 1. Capacidades no técnicas necesarias en la profesión del Ingeniero.

En primer lugar, y empezando por la parte interior de la figura 1, y saliendo hacia el exterior, hablaré de los **Idiomas**. El tiempo dedicado (en caso de existir esta materia) a los idiomas es muy pequeño, o nulo en algunas Escuelas. Y el inglés sigue siendo el idioma más necesario de cara a cualquier trabajo de ingeniero que tenga una mínima proyección internacional, hecho que en nuestro actual mundo globalizado, es imposible de omitir.

En muchos casos, un tercer idioma (alemán, chino, árabe, francés, etc.) es preciso en función de la mayor o menor relación con estos países y su tecnología.

Antes de seguir desglosando las capacidades o necesidades de diferentes áreas, comentaré brevemente las referencias bibliográficas incluidas en este artículo. Los tres libros recogidos se encuentran escritos en ingles, [1], [2], [3], estando co-publicado el primero por el IEEE y Wiley, recogiendo de forma amplia la mayoría de ideas incluidas en este artículo. El segundo amplía estos temas, y el tercero orienta de forma más concreta el tema de la comunicación en ingeniería.

Y las cuatro URLs o enlaces de Internet incluidos, apuntan en el primer caso al propio IEEE, así como el último que señala a su código ético, resumen y marco de actuación de cualquier grupo profesional, como es en este caso la ingeniería eléctrica (en su sentido amplio americano) o el propio IEEE. La segunda de las referencias apunta al ABET, comité acreditador americano y el tercero, a la ANECA; agencia española de acreditación y calidad universitaria.

Continuando con el tema de las capacidades no técnicas necesarias, en el segundo nivel se encuentran cinco (Organización personal, Comunicación, Trabajo en grupo, Liderazgo e Innovación), y en un nivel superior, le Ética. Seguiremos este orden enunciado para irlas comentado de una en una.

La **Organización personal** es una de las necesidades actuales en cualquier actividad profesional, pero que se es mucho más necesaria en la ingeniería. Así, la gestión del tiempo, tanto en lo profesional como en lo personal, y su interrelación con al necesidad de mejora, capacitación y actualización de conocimientos constante, hacen de la organización personal una necesidad en todos los frentes del ingeniero.

El uso de agendas personales (PDA), ordenadores portátiles, móviles y otros aparatos tecnológicos, implica una movilidad constante y una comunicación constante, tanto con sus compañeros o supervisores, como con el resto de la profesión (asociaciones, etc.) y fundamentalmente, con la información. El correo electrónico sigue siendo la mejor herramienta dentro de esta comunicación, así como una pesadilla si es incorrectamente utilizada (como el teléfono móvil).

La determinación de la actividad cotidiana, el conocer el día a día, la priorización y el minimizar las interrupciones en el contexto laboral, permiten una optimización de la actividad profesional. Pero el conocimiento de la eficacia (hacer lo correcto) y la eficiencia (hacer bien la actividad) permiten alcanzar el óptimo (hacer lo correcto bien).

Por último, en el desarrollo de cualquier actividad de ingeniería, el establecimiento de la planificación de la actividad, y del conocimiento de las contingencias que pueden surgir (como contraposición de la ley de Murphy) permitirán alcanzar las metas elegidas en la actividad profesional (y personal).

La **Innovación** es otra de las áreas más demandadas hoy día en nuestra sociedad tecnificada. De hecho se ha pasado de las necesidades en I+D (Investigación y Desarrollo) a las necesidades en I+D+i (Investigación, Desarrollo e Innovación) como áreas de evolución tecnológicas y de desarrollo de una nación.

Como factores principales para la innovación se pueden citar la preparación (que hoy en día se alimenta y fortalece mediante la formación continua o formación a lo largo de la vida) y la creatividad. La unión de ambos factores facilitará en nuestra profesionalidad las dosis de innovación necesarias para el correcto desarrollo de nuestra carrera profesional.

Una de las formas más extendidas en las empresas de obtener estos factores de innovación son las encuestas o las tormentas de ideas. En ambos casos, se puede decir que la innovación se basa en un 1% de inspiración y un 99% de transpiración (o sea, de trabajo). Y este trabajo cotidiano debe basarse en:

- preparación,
- concentración,
- desarrollo,
- inspiración, y
- verificación y pruebas.

A continuación pasaré a hablar de **Liderazgo** como la capacidad de organizar los objetivos de un grupo de personas con unas necesidades o similitudes comunes. Así, en el liderazgo participan la motivación, la responsabilidad, la integridad, la lealtad y la toma de decisiones, esta última en base a la participación en el grupo y la realimentación de las acciones del mismo, [4], [5], [6].

Este liderazgo está muy ligado con el **Trabajo en grupo**. Así, hoy en día cualquier proceso (siendo las fases de éste la definición del mismo, el proceso de invención, análisis, decisión, desarrollo e iteración) es preciso trabajar en equipo, mostrando elevadas dosis de compañerismo, integración, responsabilidad y participación en las tareas del grupo. Esta es una de las características más valoradas en la cultura empresarial actual.

Así, el saber escuchar y actuar con sinergia, y proponer ideas y temas efectivos, evidentemente, podrán conducir a una gestión del tiempo más adecuada en las reuniones permitiendo cumplir las agendas y organizando éstas de una manera más efectiva.

Por último en este nivel, se comentará la necesidad de tener unas buenas dotes de Comunicación. Esta comunicación o el uso adecuado de la capacidad de hablar en público (desde pequeños grupos en reuniones de trabajo, hasta grandes grupos en reuniones empresariales o en presentaciones comerciales) es muy valorada por las empresas. Así, la capacidad de escribir buenos informes y presentaciones que permitan su posterior presentación con una elevada capacidad de convicción o de negociación, según el entorno deseado. Como idea básica, no se debe leer nunca un discurso o comunicación sino que debe interiorizarse su contenido para poderlo formular de forma más adecuada.

Como cierre de este artículo, y envolviendo todo lo anterior, se debe hablar de **ÉTICA**. La ética, [7], no debe integrar una serie larga e ilimitada de temas que se deben cumplir escrupulosamente sino que en general deben ser unos principios básicos de actuación, que al igual que en nuestra vida cotidiana y personal, nos deben marcar las pautas en nuestra vida profesional.

Así, se deben tener claros una serie de principios y valores (mantener la seguridad en un nivel lo más elevado posible, el cuidado del medio ambiente o la falta de discriminación) en el camino hacia la productividad y el ROI (retorno de la inversión) y la obtención de beneficios. Sin embargo se debe tener en cuenta que a veces estos valores dependen de hechos y aspectos culturales (como son los fines a usar para obtener un objetivo, o el uso de patentes y copyright en función de unos beneficios o márgenes o el mantenimiento de IPRs), donde a veces no existe ilegalidad sino noética (no debe confundirse con falta de ética).

CONCLUSIONES

Se han presentado en este artículo una serie de capacidades no técnicas que cada vez se van integrando (sobre todo en el extranjero) dentro de los conocimientos que debe tener un ingeniero cuando finaliza su preparación en las Escuelas de Ingeniería y en las que nuestros estudiantes habrán de dedicar parte de su tiempo dentro de los futuros programas de formación basados en los créditos europeos, ECTS.

REFERENCIAS BIBLIOGRÁFICAS

1. C. Selinger. *Stuff you don´t learn in Engineering School – Skills for success in the real world*. Ed. Wiley Interscience / IEEE Press, 2004.
2. J.H. Kemper y B.R. Sanders. *Engineers and their profession*. Ed. Oxford University Press, 2001.
3. H. Hart. *Introduction to Engineering communication*. Ed. Pearson/Prentice Hall, 2005.
4. Institute of Electrical and Electronics Engineers, Inc. – http://www.ieee.org/
5. Accreditation Board for Engineering and Technology, Inc. – http://www.abet.org/
6. Agencia Nacional de Evaluación de la Calidad y Acreditación – http://www.aneca.es/
7. IEEE Code of Ethics – http://www.ieee.org/portal/pages/about/whatis/code.html

BIOINGENIERÍA

SISTEMA DE MONITORIZACIÓN, ALMACENAMIENTO Y TRATAMIENTO DE DATOS

Aplicación en los servicios de reanimación de los hospitales

Por Fco. García Sevilla

El espectacular desarrollo de la electrónica en los últimos años y, como consecuencia de ello, de la informática, han hecho que el computador se convierta en una herramienta imprescindible en la mayor parte de las actividades humanas, y la medicina es una de ellas.

En el año 1979 Index Medicus recogía tres encabezamientos con cuatro páginas dedicadas a referencias sobre la utilización de computadores en medicina. En 1987 dedicó siete encabezamientos, diecinueve referencias cruzadas y doce páginas a artículos procedentes de más de diez revistas especializadas en computadores y ciencias de la salud. En la actualidad es difícil encontrar trabajos de investigación en el campo de la medicina en los que no se utilice el computador de alguna forma.

En medicina crítica es preciso utilizar equipos de electromedicina para el control de la evolución de los enfermos. Máquinas tales como respiradores, capnógrafos, pulsioxímetros, etc. son imprescindibles en los Servicios de Reanimación, ya que conectados a los pacientes a través de diferentes sensores y captadores, proporcionan periódicamente información sobre sus constantes fisiológicas y, en su caso, actúan con el fin de que aquellas se mantengan dentro de los márgenes establecidos.

Los equipos para el control de pacientes suelen funcionar de forma autónoma e independientes unos de otros, en tiempo real (presentan las magnitudes en cada instante) y no suelen almacenar dichas magnitudes.

Los sistemas centralizados para la monitorización y almacenamiento de datos comerciales que conocemos funcionan con equipos de la misma casa comercial y, por lo tanto, utilizan las mismas conexiones y protocolo para la comunicación de los equipos periféricos con la unidad central.

Cuando se dispone de equipos de diferentes características y fabricantes (lo cual es habitual en los Servicios de Reanimación) la centralización se complica notablemente al tener que realizar la comunicación de cada equipo periférico con la unidad central de una forma diferente y no contar, en muchos casos, con la información técnica de las máquinas.

El SMATD se ha desarrollado a partir de los equipos existentes en el Servicio de Reanimación del Hospital General Universitario de Albacete y ha sido adaptado a las necesidades de utilización del mismo en vigilancia e investigación.

Descripción del sistema

El sistema se ha desarrollado, en una primera fase, para la conexión de cuatro equipos de electromedicina con interface RS-232C, en cuatro de las cabinas del Servicio de Reanimación, es decir, un total de dieciséis equipos de forma simultánea. También puede conectarse al sistema una señal de tipo analógico procedente de cada una de las cabinas.

La Unidad Central está constituida por dos computadores Pentium III a 450 MHz, conectados entre sí mediante un cable de red ethernet cruzado. El computador principal (CP) se dedica a la adquisición, visualización y almacenamiento de datos de forma ininterrumpida (Figura 1), mientras que el computador secundario (CS) se utiliza como equipo auxiliar para la configuración del sistema y el tratamiento de los datos adquiridos. Se ha optado por esta forma de trabajo para que no se interrumpa la adquisición de datos cuando se realicen otras tareas en el sistemas tales como altas, bajas o modificación de datos de los enfermos, establecimiento de las magnitudes a medir o tratamiento de los datos adquiridos.

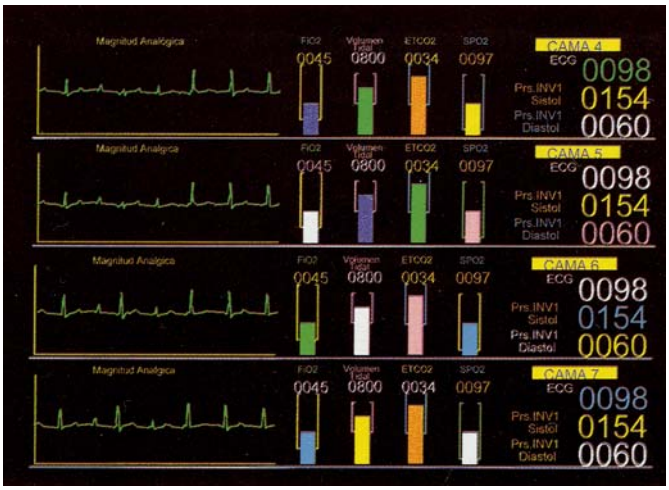


FIGURA 1. Pantalla principal.

En las ranuras de expansión del CP se han instalado dos tarjetas especiales para la comunicación con los equipos de electromedicina situados en las diferentes cabinas. Una de las tarjetas es una multiplexora de RS-232C con 16 entradas/salidas independientes. Cada una de ellas se puede configurar como dispositivo terminal de datos (DTE) o como dispositivo de comunicación de datos (DCE). Esta tarjeta dispone de 512 kbytes de memoria RAM que utiliza como buffer para los 16 canales; la velocidad de transmisión es configurable para cada canal y se tiene control sobre las señales TX, RX, RTS, CTS, DTR y DSR. A través de esta tarjeta se conectan, simultáneamente hasta 16 equipos de electromedicina con interface RS-232C.

La otra tarjeta utilizada en el CP es una placa de entrada/salida conocidas habitualmente como tarjetas de adquisición de datos. Esta permite la conexión de 16 entradas analógicas simples u ocho diferenciales con una frecuencia de muestreo variable hasta 100.000 muestras por segundo, con una resolución de

12 bits y ganancia independiente para cada canal. Dispone asimismo, de 16 entradas digitales, 16 salidas digitales y 2 salidas analógicas. Al computador secundario únicamente se le ha instalado, a partir de su configuración inicial, una unidad de almacenamiento en cinta magnética (DAT).

El enlace entra la unidad central CP y los equipos periféricos se realiza a través de cables coaxiales que alcanzan los 20 metros en algunos casos, y a los que se accede mediante una caja de conexiones en cada cabina.

La transmisión de datos digitales (interface RS-232C) se realiza de forma directa equipo-tarjeta multiplexora a través de los cables de comunicación sin ningún tipo de acondicionamiento. La transmisión de las señales analógicas se realiza en lazos de corriente (para minimizar los efectos del ruido eléctrico en la transmisión) conectando convertidores V-I en las salidas analógicas de los equipos (situados en las cabinas) y convertidores I-V en las entradas de la tarjeta de adquisición de datos donde se realiza también un filtrado.

La comunicación entre los computadores se realiza a través de una tarjeta de red ethernet, lo cual permite alcanzar altas velocidades en la transmisión de datos. Los equipos con interface RS-232C que en esta primera fase se conectan con el SMATD se indican en la Tabla 1. Como señales analógicas (una por cabina) se utilizan las que proporcionan el Athena Monitor en sus canales analógicos 1 y 2, que son configurables desde la propia máquina.

Nombre	Fabricante	Descripción
Advent	Ohmeda	Respirador
Biox 3740	Ohmeda	Pulsioxímetro
Adult Star	Intro Sonic Inc.	Respirador
Oxicap 4700	Ohmeda	Capnógrafo
Spiegelberg	Geprüfte	Medidor de presión intracraneal epidural
Athena Monitor	Sicherheit	Monitor

TABLA 1. Equipos de biomedicina.

El SMATD permite la gestión y almacenamiento de hasta 64 magnitudes para cada cabina (paciente) cada 20 segundos, es decir, se pueden almacenar 64x4 datos tres veces por minuto. Actualmente se almacenan 25 magnitudes que, para cada máquina, son:

- Advent: Volumen Corriente (Volumen Nidal), Peep-Cpap, Volumen minuto, Presión en vía aérea, Frecuencia respiratoria, FiO2.
- Biox 3740: Saturación de oxígeno (SpO2), Pulso.
- Adult Star: FiO2 (OxyCon), Volumen Corriente (TidVol), MacRat, Presión en vía aérea, PIP (Peep-Cpap), Volumen minuto (MinVol).
- OxiCap 4700: Saturación de oxígeno (SpO2), Pulso, Fracción expirada de carbónico (EtCO2).

- Spiegelberg: Presión intracraneal media (PIC media).
- Athena Monitor: Electrocardiograma (ECG), Presión arterial no invasiva sistólica (NIPBS) y diastólica (NIBPD), Presiones arteriales invasivas 1 y 2 sistólicas (IBP1S e IBP2S) y diastólicas (IBP1D e IBP2D), Temperatura.

Principio de funcionamiento

El computador principal gestiona la comunicación con las máquinas, la visualización de magnitudes (monitorización) y el almacenamiento en disco magnético de los datos adquiridos. El computador secundario se utiliza con distintos fines, bien para la configuración del CP, bien para la salvaguarda de datos (backup) entre computadores o para el tratamiento y análisis de los datos adquiridos. Con esta configuración se realiza la adquisición de datos de los pacientes de forma ininterrumpida en un computador y la gestión del sistema (datos de los pacientes, máquinas conectadas, magnitudes seleccionadas, límites de alarmas, colores en pantalla, etc.) que puede suponer un tiempo considerable en el que se perderían datos de los enfermos, en otro diferente.

El CP funciona de forma ininterrumpida, por el contrario, el CS sólo se conecta cuando sea necesario un cambio de configuración, una salvaguarda de datos o el tratamiento de los mismos, lo que posibilita la utilización de este computador para otras tareas.

Cuando se conecta el SMATD, el CP carga la configuración establecida en el CS y comprueba que existe comunicación con las máquinas seleccionadas, desconectándolas en caso de error. A continuación se inicializa la tarjeta convertidora de datos analógicos/digitales (A/D) y pasa al bucle principal del programa. En este, se leen las máquinas cada 20 segundos y se comprueban los límites de cada magnitud verificando si éstas quedan dentro de ellos, activando los mensajes de alarma en caso contrario. En los intervalos entre lecturas de las máquinas, el CP comprueba si hay petición de transmisión desde el CS.

La visualización de las magnitudes se realiza en un monitor dividido verticalmente en cuatro zonas, una para cada cabina (en la Figura 1 se muestra la pantalla).

Para cada paciente se puede representar en forma de gráfica en el tiempo (parte izquierda de la pantalla) una señal analógica (en este caso la ECG) proveniente del Athena Monitor. En la parte central se pueden representar, en forma de barras, cualquiera de las magnitudes, existiendo unos “corchetes” en los que se representan los valores máximos y mínimos entre los que puede variar la magnitud sin que se activen las alarmas. Por último, en la parte derecha de la pantalla existen tres indicadores numéricos en los que se puede representar también el valor de cualquiera de las magnitudes de las diferentes máquinas conectadas.

En las opciones de configuración el operador puede decidir las magnitudes que se representan en cada uno de los indicadores, sus valores máximos y mínimos e incluso los colores con los que se distinguirán en pantalla.

Las alarmas en el SMATD, que se activan cuando cualquiera de las magnitudes cae fuera de los márgenes establecidos, producen señales acústicas de aviso (si así se desea) y siempre mensajes de error en la línea de estado del sistema que se sitúa en la parte inferior de la pantalla.

El software, desarrollado específicamente para esta aplicación, se ha realizado con el lenguaje de programación Visual C++ de Microsoft. Con él se dispone, por un lado, de las características del lenguaje C para la programación a bajo nivel y, por otro, de las enormes posibilidades de un lenguaje orientado a objetos, que además proporciona las librerías necesarias para implementar el entorno de utilización buscado. La orientación a objetos supone, entre otras ventajas, el desarrollo modular y estructurado de los programas, la facilidad de su depuración y puesta a punto y la simplificación de su crecimiento (nuevas opciones o funciones), etc.

Existen dos programas principales para la gestión y explotación del sistema, que se denominan SCDA y REA. Se han programado en bajo nivel y con orientación a objetos. El programa SCDA gestiona la adquisición, visualización y almacenamiento de datos y el programa REA se ocupa de la gestión de pacientes (altas, datos personales, datos médicos, historial, etc.), de la configuración del CP y de las comunicaciones entre el CP y el CS.

El programa SCDA, instalado en el CP, controla la tarjeta multiplexora de RS-232C y la tarjeta de adquisición de datos. Las máquinas dotadas de interface RS-232C, se han configurado para que produzcan la salida de magnitudes tras la recepción de las correspondientes órdenes de volcado de datos que envía el CP cada 20 segundos a cada una de las máquinas.

La lectura de las señales analógicas se realiza por interrupciones del programa que provoca el contador/temporizador de la tarjeta de adquisición de datos a razón de 1000 lecturas/segundo para los cuatro canales analógicos conectados (250 para cada uno de ellos que es el número de muestras que proporcionan las salidas analógicas del Athena Monitor).

En cada interrupción se guarda el estado de vídeo, se calcula el punto de la curva que hay que representar (borrando el anterior cuando es necesario) y se vuelve a restaurar el modo de vídeo para permitir la escritura de las demás magnitudes.

El programa REA, instalado en el CS, se ocupa de la gestión de datos de los pacientes, configuración del sistema y comunicación con el CP. Por tanto, el operador del sistema lo controla con este programa, por lo que se ha diseñado teniendo en cuenta las siguientes especificaciones principales:

- De fácil utilización por cualquier usuario sin conocimientos de informática, interface agradable y con un alto grado de seguridad, de manera que se minimice la posibilidad de error.
- Comunicación entre equipos y/o computadores de forma rápida y segura.
- Versatilidad en la utilización de los recursos del sistema.
- Facilidad de crecimiento y/o adaptación a las nuevas necesidades del sistema.

El programa se ha organizado como un conjunto de menús desplegables y ventanas de entrada de datos, a las que se puede acceder mediante el teclado o utilizando el ratón. Esta parte del programa se ha realizado utilizando las clases que se incluyen en la librería MSFC de Microsoft Visual C++. En muchas ocasiones se han transformado estas clases mediante un mecanismo de herencia para solucionar aspectos muy específicos que se han planteado en esta aplicación. Por tanto, se puede decir que la gestión más compleja del interface programa-usuario la realiza una segunda generación de clases derivadas.

En el programa existen otras estructuras de entre las que destacan por su importancia las clases “Cama”, “Paciente” o “Equipo”, sobre todo si se tiene en cuenta que en cada cama (cabina) se pueden conectar varias máquinas, cada una de las cuales puede medir varias magnitudes que, a su vez, tienen distinto tratamiento.

Tratamiento de los datos

Como ya se indicó, el SMATD se está utilizando con una doble finalidad: por un lado, como sistema de monitorización que permite, desde una sola pantalla, controlar la evolución de hasta cuatro pacientes, y por otro lado para el almacenamiento de las constantes fisiológicas de los pacientes para su posterior tratamiento con diferentes fines relacionados con la investigación.

Las posibilidades de explotación de los datos almacenados sobre la evolución de los enfermos con alta dependencia externa, son múltiples en función de los objetivos que se busquen en cada caso. En el nuestro nos hemos planteado, inicialmente, el tratamiento de dichos datos desde el punto de vista cronobiológico.

La cronobiología es la ciencia que cuantifica e investiga los mecanismos de la estructura temporal biológica, incluyendo las manifestaciones rítmicas de la vida.

Ritmos con diferentes frecuencias y fases están presentes en todos los niveles de integración biológica: ecosistemas, población, grupo, organismo, sistema orgánico, órgano, tejido, célula y estructura celular. Series temporales extensas y densas recogidas a lo largo de varias décadas, demuestran que las variables biológicas presentan algún grado de comportamiento más o menos periódico.

Se denomina cronoma a la suma de todos los ritmos y trenes que componen la estructura temporal biológica individual o poblacional.

Los métodos cronobiológicos definen la salud cuantificando la dinámica de los valores que permanecen dentro de los rangos fisiológicos y los estándares de variación especificados temporalmente.

El conocimiento de los ritmos que caracterizan al organismo humano en la salud y sus alteraciones o cambios en al enfermedad permiten un mejor conocimiento de éstas, la actuación sobre ellas en los momentos más adecuados y, sobre todo, su predicción.

Para la obtención y caracterización de los citados ritmos, se utilizan diferentes métodos matemáticos de tratamiento de la señal entre los que cabe destacar el cosinor simple, las transformada discretas (DFT) y rápida (FFT) de Fourier, el modelo autorregresivo (AR), la transformada wavelet (WT), etc.

Primeras medidas experimentales

Las Figuras 2 y 3 muestran algunas gráficas correspondientes a diferentes magnitudes obtenidas de los pacientes del Servicio de Reanimación del HGUA y que están tratadas utilizando los métodos anteriormente indicados. En las gráficas se han omitido los datos que podrían identificar a los pacientes.

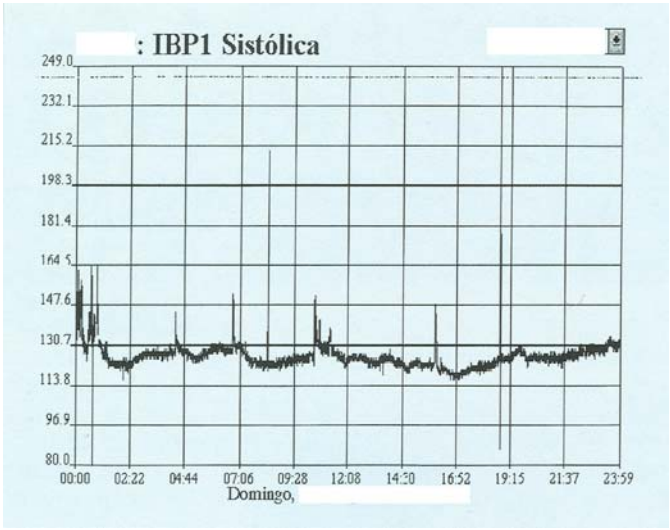


FIGURA 2. Presión invasiva sistólica.

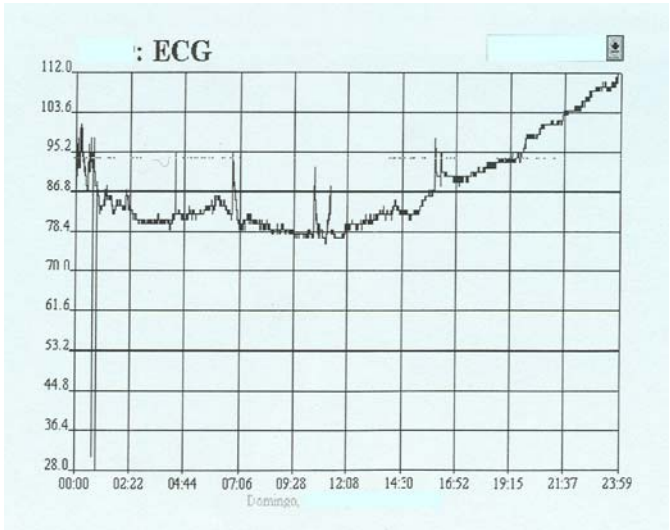


FIGURA 3. Electrocardiograma.

Conclusiones

Aunque el SMATD lleva instalado y funcionando sólo unos meses en el Servicio de Reanimación del HGUA, se han podido comprobar sus excelentes características y grandes prestaciones que lo convierten en una potente herramienta tanto desde el punto de vista de la atención a los enfermos con alta dependencia externa, como para su utilización en diferentes líneas de investigación. Una de estas líneas en curso es el estudio cronobiológico de los datos almacenados. Para este tratamiento se están aplicando diferentes transformadas y métodos matemáticos que permitan extraer la máxima información, de las diferentes señales, para nuestros fines. Algunos datos se han enviado, para su estudio, al profesor F. Halberg de la Universidad de Minesota (Estados Unidos).

En los próximos meses se realizará la ampliación del número de cabinas conectadas al sistema (y por tanto del número de equipos de electromedicina) de tal forma que ocho camas de enfermos de alta dependencia externa del servicio de reanimación puedan ser controladas por el SMATD.

Fco. García Sevilla
Coordinador del Comité de socios y bienvenida.
Ingeniero Industrial.
fgsevilla@ieee.org

DIFERENCIAS ENTRE JAVA, JAVASCRIPT, JSP Y ASP

Por Eugenio López Aldea

En la actualidad, las aplicaciones y el acceso a datos a través de Internet es algo muy común tanto en las empresas como en los organismos oficiales, así como cualquier aplicación que necesite una interrelación entre el usuario y una aplicación que esté alojada, por ejemplo, en un servidor.

Actualmente, ya hay varias soluciones en el mercado, que resuelven la problemática para realizar grandes aplicaciones de una forma relativamente sencilla. Sin lugar a dudas, Microsoft a través de la tecnología .NET ha ofrecido un gran avance. A través de sus lenguajes anteriores como Visual Basic, han dado vida a una serie de herramientas que el programador podrá usar de una forma sistemática y ordenada para llevar a cabo sus objetivos. Para cumplir éstos, es necesario el desembolso de un dinero inicial, tanto para la compra de elementos *hardware* como elementos de *software*, por ejemplo Visual Studio .NET como programa de desarrollo integrado (IDE – *Interface Developtment Environment*), Microsoft SQL Server como administrador de bases de datos, IIS (*Internet Information Server)* como Servidor de Páginas Web entre otros.

En Internet, fuente inagotable de recursos, también podemos realizar aplicaciones análogas y de gran alcance, mediante lenguajes de programación de libre acceso (*Open Source*) [URL1] [URL2] que no suponen un desembolso inicial. Este es el caso de Java de Sun Microsystems. Sin embargo, sin entrar en la polémica de la competencia, en este artículo describiremos las diferencias entre Java, JavaScript y JSP y mencionaremos alguna diferencia entre JSP y ASP para resolver diferentes planteamientos dentro de las aplicaciones *online* o lo que podríamos llamar entre nosotros: “*e-aplicattions*”.

A continuación vamos a ir recogiendo diferentes definiciones de Internet y de varios libros que serán de utilidad para conocer bien qué se puede hacer y cuales son las diferencias.

Comencemos con Java

Java es un lenguaje de programación con el que podemos realizar cualquier tipo de programa. En la actualidad es un lenguaje muy extendido y cada vez cobra más importancia tanto en el ámbito de Internet como en la informática en general. Está desarrollado por la compañía Sun Microsystems. [URL3]

Una de las ventajas de Java es la capacidad de ejecutarse a través de cualquier ordenador independientemente de la plataforma, es decir, podría correr bajo Linux o Windows. Esto es, obviamente una ventaja significativa que le confiere de una gran libertad y potencia.

Con Java podemos programar páginas Web dinámicas, con accesos a bases de datos, utilizando estándares (XML), a través de componentes ya preprogramados (como JDBC) con cualquier tipo de conexión de red entre

cualquier sistema. En general, cualquier aplicación que deseemos hacer con acceso a través Web se puede hacer utilizando Java.

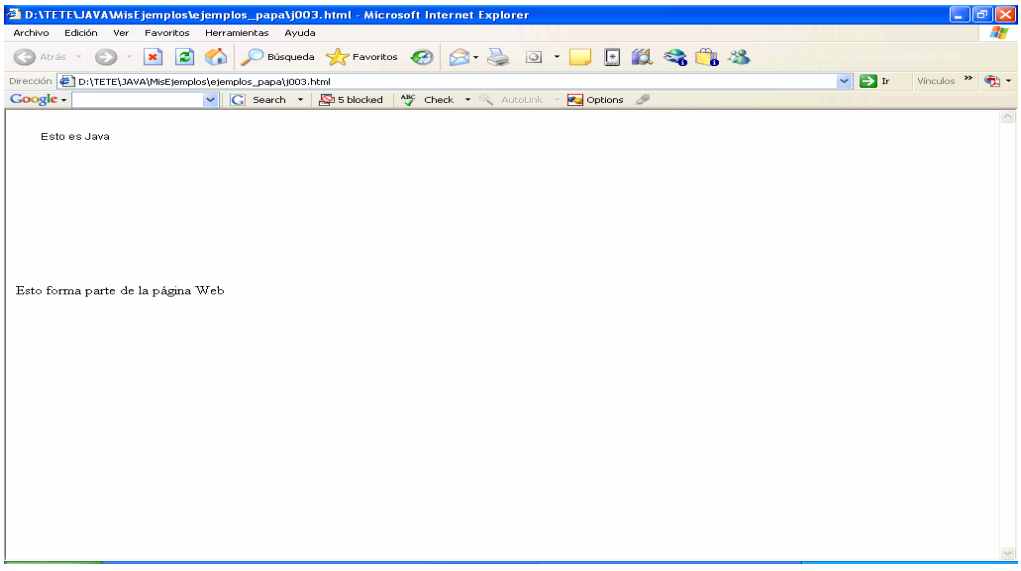
Se expone a continuación un ejemplo de una aplicación en Java que se ejecuta en Internet. Se trata de un *applet* de Java.

```
//Applet j002
import java.awt.Graphics;
import java.applet.Applet;
public class j002 extends Applet{
 public void paint(Graphics g){
 g.drawString("Esto es Java",25,25);
            }
   }
```

Este Applet puede ser llamado desde una página Web mediante el siguiente código HTML:

```
<html>
<applet code=j002.class width=100 heigt=100>
</applet>
<body>
<br><br>
Esto forma parte de la página Web
</body>
</html>
```

La aplicación simplemente diferencia entre la parte ejecutada de Java y la ejecutada con HTML. Véase la siguiente Figura.



De una forma análoga podríamos hacer las aplicaciones que deseáramos y ejecutarlas bajo un explorador como Explorer o Netscape y cualquier persona desde cualquier parte con conexión a Internet podría verlo (siempre que esta aplicación se subiera a algún servidor de páginas Web).

Sigamos con JavaScript

Del libro de la referencia [1] podemos sacar las siguientes conclusiones:

"JavaScript es el lenguaje de secuencia de comandos (o *scripts*) en cliente más utilizado actualmente en la Web. Su uso está muy extendido en tareas que van desde la validación de los datos de formularios a la creación de complejas interfaces de usuario.

Algunos de los usos típicos de JavaScript son los siguientes:

- Validación de formularios
- Decoración de páginas y efectos especiales.
- Sistemas de navegación (menús desplegables, etc.).
- Cálculos matemáticos básicos.
- Generación dinámica de documentos.

Históricamente, la función principal que JavaScript venía a realizar era la comprobación simple de datos de los formularios antes de su envío a una aplicación en el servidor, como por ejemplo, un programa CGI. Esto se conoce generalmente como validación de formularios. Está siendo muy usado para emplear sistemas de navegación como por ejemplo, los menús desplegables o las listas de consulta. También se ha empleado para cálculos pero básicos en cliente, como calculadoras para créditos y otras aplicaciones que pueden resultar útiles para un visitante Web. Quizá la utilización más novedosa para JavaScript es la manipulación dinámica de documentos de forma sencilla, como es la inclusión condicional de código o de marcado dependiendo de la versión del navegador, y al final se utilizará para modificar una página de forma significativa después de cargarla. Este concepto se llama a menudo *HTML dinámico o DHTML* de forma abreviada.

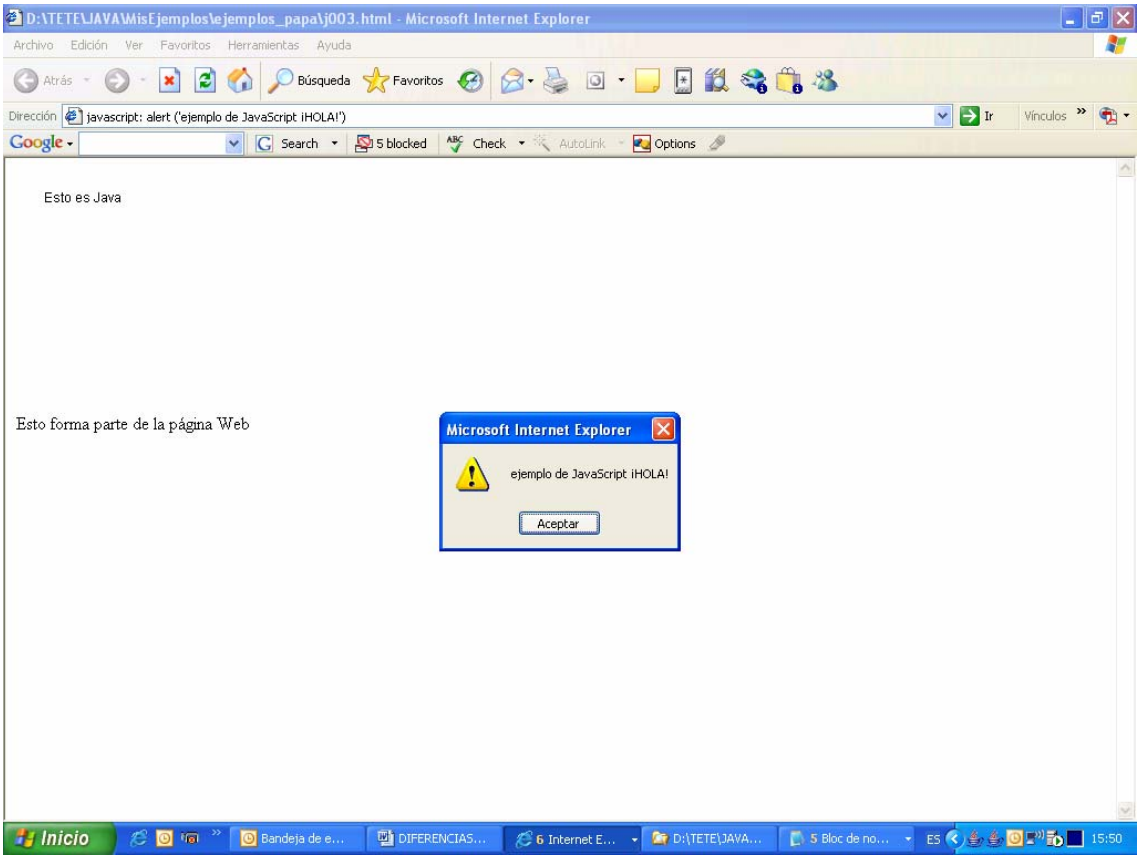
Aunque en JavaScript se pueden escribir auténticas aplicaciones (en toda la extensión de la palabra), la mayor parte de los programadores no lo encuentra adecuado para aplicaciones extensas debido a los errores, las diferencias entre navegadores y estabilidad de este lenguaje. Recuerde que es un lenguaje de secuencias de comandos (scripts) y como tal se supone que está enfocado a un uso específico.

Como conclusión, actualmente para realizar grandes aplicaciones que requieran más potencia, se recomendaría usar Java como lenguaje e intercalar para la Interface JavaScript en el lenguaje HTML de las páginas Web. Esto mismo se puede hacer con ASP y JSP, intercalar JavaScript para elementos concretos y darle mayor "vida" a las páginas Web.

Véase un ejemplo [URL4] de JavaScript en el buscador de páginas. Escribiendo:

javascript: alert ('ejemplo de JavaScript ¡HOLA!')

en el buscador, aparecerá el siguiente letrero que se ve en la mitad de la pantalla:



El letrero en cuestión es:

A continuación se expone un ejemplo de un reloj digital en JavaScript:

<script language="javascript">

// Reloj digital con DHTML, by Vanessa Jaen

//

// Este script y otros muchos pueden

// descarse on-line de forma gratuita

// en El Código: www.elcodigo.net

```
function muestraReloj() {
    // Compruebo si se puede ejecutar el script en el navegador del usuario
    if (!document.layers && !document.all && !document.getElementById)
            return

    // Obtengo la hora actual y la divido en sus partes
    var fechacompleta = new Date()
    var horas = fechacompleta.getHours()
    var minutos = fechacompleta.getMinutes()
    var segundos = fechacompleta.getSeconds()
    var mt = "AM"

    // Pongo el formato 12 horas
    if (horas > 12) {
            mt = "PM"
            horas = horas - 12
    }
    if (horas == 0)
            horas = 12

    // Pongo minutos y segundos con dos dígitos
    if (minutos <= 9)
            minutos = "0" + minutos;
    if (segundos <= 9)
            segundos = "0" + segundos;

    // En la variable 'cadenareloj' puedes cambiar los colores y el tipo de fuente
    document.title = horas + ':' + minutos + ':' + segundos
    cadenareloj = "<p>" + horas + ":" + minutos + ":" + segundos + " " + mt + "</p>"

    // Escribo el reloj de una manera u otra, según el navegador del usuario
    if (document.layers) {
            document.layers.spanreloj.document.write(cadenareloj)
            document.layers.spanreloj.document.close()
    } else if (document.all) {
            document.all["spanreloj"].innerHTML = cadenareloj
```

```
    } else if (document.getElementById) {
            document.getElementById("spanreloj").innerHTML = cadenareloj
    }

    // Ejecuto la función con un intervalo de un segundo
    setTimeout("muestraReloj()", 1000)
}

</script>
```

Se recoge a continuación algunos comentarios para aclarar algunos conceptos [2]:

- **Java no es un lenguaje para Internet.** Java es, como ya se ha comentado anteriormente, un lenguaje de propósito general que, al ser independiente de la plataforma, es una herramienta excepcional para Internet.
- **Java no es un lenguaje interpretado.** Aunque para su ejecución definitiva en la máquina es necesario un intérprete –la Máquina Virtual Java- tiene mucho más en común con los lenguajes compilados, como C o Pascal, que con los lenguajes interpretados.
- **Java y JavaScript no son lenguajes parecidos.** De hecho, tienen en común poco más que el nombre. JavaScript es un lenguaje interpretado licenciado por Nestcape que nos recuerda a Java (tanto como C, por ejemplo). El código de JavaScript como es interpretado, no tiene sentido hablar de fuente o ejecutable, va incluido en la propia página Web.

JSP

Esta parte está recogida de la página Web [URL5] y para más información consultar ésta referencia.

Los *servlets* y *Java Server Pages (JSPs)* son dos métodos de creación de páginas Web dinámicas en servidor usando el lenguaje Java. En ese sentido son similares a otros métodos o lenguajes tales como el PHP, los CGIs (common gateway interface)**,** programas que generan páginas Web en el servidor, o los ASP (Active Server Pages), un método específico de Microsoft. Sin embargo, se diferencian de ellos en otras cosas.

Para empezar, los JSPs y servlets se ejecutan en una máquina virtual Java, lo cual permite que, en principio, se puedan usar en cualquier tipo de ordenador, siempre que exista una máquina virtual Java para él. Cada servlet (o JSP, a partir de ahora lo usaremos de forma indistinta) se ejecuta en su propia hebra, es decir, en su propio contexto; pero no se comienza a ejecutar cada vez que recibe una petición, sino que persiste de una petición a la

siguiente, de forma que no se pierde tiempo en invocarlo (cargar programa + intérprete). Su persistencia le permite también hacer una serie de cosas de forma más eficiente: conexión a bases de datos y manejo de sesiones, por ejemplo.

Los JSPs son en realidad servlets: un JSP se compila a un programa en Java la primera vez que se invoca, y del programa en Java se crea una clase que se empieza a ejecutar en el servidor como un servlet. La principal diferencia entre los servlets y los JSPs es el enfoque de la programación: un JSP es una página Web con etiquetas especiales y código Java incrustado, mientras que un servlet es un programa que recibe peticiones y genera a partir de ellas una página Web

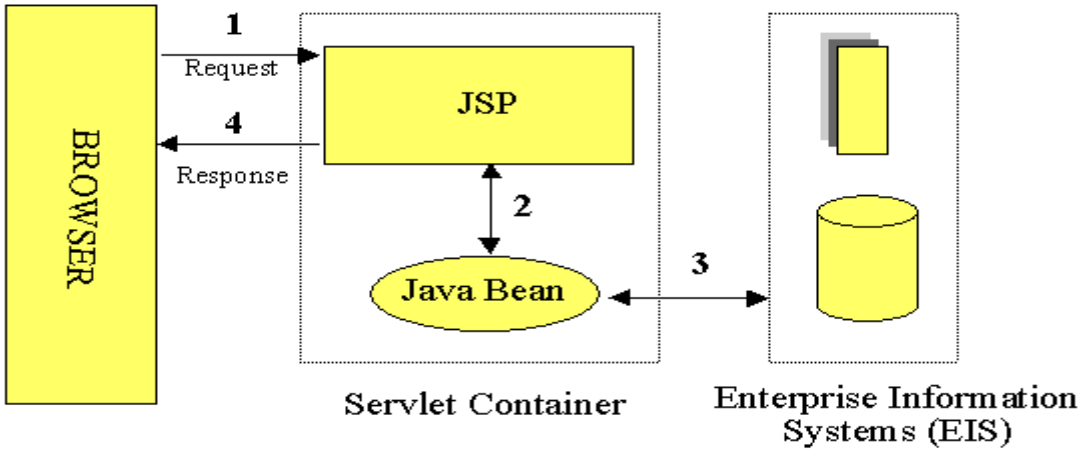


Figura recogida de [URL5]

Ambos necesitan un programa que los contenga, y sea el que envíe efectivamente páginas Web al servidor, y reciba las peticiones, las distribuya entre los servlets, y lleve a cabo todas las tareas de gestión propias de un servidor Web. Mientras que servidores como el Apache están especialmente pensados para páginas Web estáticas CGIs, y programas ejecutados por el servidor, tales como el PHP, hay otros servidores específicos para servlets y JSPs llamados *contenedores de servlets (servlet containers)* o *servlet engines*.

No se hace mención aquí sobre las herramientas o IDEs creados para facilitar la programación con Java (como JBuilder, NetBeans, Ultraedit, etc.) pues hay creadas muchas y no es objetivo del mismo, sin embargo se puede recurrir a la página Web http://meteo.ieec.uned.es/www_Usumeteo8/ para mayor información.

Algunas diferencias entre ASP y JSP

Esta parte está recogida de la referencia [URL3]: "JSP y ASP sirven para hacer, más o menos, el mismo tipo de aplicaciones Web. Sin embargo, en el fondo tienen bastantes diferencias.

Con ASP (*Active Sever Page*) y JSP (Java Server Page) se pueden resolver mucho problemas de Inteface de usuario y comunicar éstos con bases de datos relacionales como Oracle, MySQL, etc.

Plataforma e independencia del servidor, JSP sigue la filosofía de la arquitectura JAVA de "escribe una vez ejecuta donde quieras". La implantación de ASP está limitada para arquitecturas basadas en tecnología Microsoft.

Así, JSP se puede ejecutar en los sistemas operativos y servidores Web más populares, como por ejemplo Apache, Netscape o Microsoft IIS. Mientras que ASP sólo tiene soporte nativo para los servidores IIS y Personal Web Server, que son los dos servidores Web para sistemas Microsoft, el primero con tecnología NT y el segundo para sistemas Windows 98 y similares.

Proceso de desarrollo abierto (*Open Source*). El API JSP se beneficia de la extendida comunidad JAVA existente, por el contrario la tecnología ASP es específica de Microsoft que desarrolla sus procesos internamente.

Reusabilidad entre plataformas. Los componentes JSP son reusables en distintas plataformas (UNIX, Windows). La tecnología JSP usa Java como lenguaje de Script mientras que ASP usa VBScript o Jscript. Java es un lenguaje más potente y escalable que los lenguajes de Script. Las páginas JSP son compilados en Servlets por lo que actúan como una puerta a todos los servicios Java de Servidor y librerías Java para aplicaciones HTTP. Java hace el trabajo del desarrollador más fácil p. e. ayuda a proteger el sistema contra las "caídas" mientras que las aplicaciones ASP sobre sistemas NT son más susceptibles a sufrirlas, también ayuda en el manejo de la memoria protegiendo contra fallos de memoria y el duro trabajo de buscar los fallos de perdida de punteros de memoria que pueden hacer mas lento el funcionamiento de una aplicación.

Una de las conclusiones a las que se puede llegar para usar JSP o ASP es que Java es más potente y libre, sin embargo, ASP es más sencillo de utilizar, sobre todo, si no se tiene experiencia en programación.

CONCLUSIÓN

Actualmente y según se ha visto, hay muchas soluciones para ofrecer posibilidades de creación de interfaces de diferentes *e-aplicattions*: Java, .NET, JavaScript, JSP, y un montón más como MySQL o MSSQL Sever para administrar bases de datos, Apache, Tomcat, IIS como servidores de páginas Web, Flash dinámico para presentaciones e Interfaces así como aplicaciones Web más dinámicas y potentes mediante ActionScript, y muchos más como PHP, Perl, XML, XSL, VBScript, ASP, etc. Cada vez más nos inundan éstas palabras de tres letras y que de una forma u otra dan vida a la gran red mundial, Internet.

A la hora de utilizar esta tecnología para resolver alguna aplicación *online*, es necesario un buen análisis inicial y conocer cuál de estas tecnologías podría usarse según nos convenga, teniendo en cuenta que hay tecnología de libre distribución y otras de más fácil manejo pero con el coste inicial a aportar.

Adquirir el conocimiento de todos los lenguajes si uno no está dedicándose continuamente a ello, es más complicado. No obstante, si se puede disponer de los conocimientos generales que le indiquen a uno hacia donde caminar a la hora de hacer el diseño. El objetivo de este artículo es simplemente dar algunas referencias para no confundir entre Java y JavaScript así como, haber generado la curiosidad del lector para que se interese por este mundo tan simple y complejo al mismo tiempo. En este artículo, se mezclan algunos textos de interés de las referencias con la experiencia del autor y que le pueden servir al que se inicia en estos temas.

REFERENCIAS

URLs

[URL1] http://www.google.es/search?hl=es&lr=lang_es&oi=defmore&defl=es&q=define:Open+Source

[URL2] http://www.webtaller.com/maletin/articulos/que-significa-open-source.php

[URL3] http://www.desarrolloweb.com/articulos/497.php?manual=15

[URL4] http://www.elcodigo.com/

[URL5] http://geneura.ugr.es/~jmerelo/JSP/

Libros:

[1] Manual de JavaScript. Thomas Powell y Fritz Schneider. Mc. Graw Hill, 2002

[2] Manual Imprescindible de Java. José Manuel Framiñán Torres. Anaya, 1999

Eugenio López
Presidente de la Rama de Estudiantes del IEEE-UNED.
Estudiante de Doctorado en el DIEEC.
Ingeniero Industrial.
eugenio@ieee.org

DISEÑO TÉCNICO CON UML

Por Ignacio García-Caro García

Introducción

Los sistemas o aplicaciones, toman forma cuando una o varias personas tienen la visión de cómo la tecnología puede mejorar las cosas. Los desarrolladores deben entender la idea mientras crean el sistema, para ello debe existir un enlace entre quien tiene la idea y el desarrollador.
El UML (Lenguaje unificado de modelado) es una herramienta que cumple con esta función, se basa en símbolos y diagramas que permiten a los creadores generar diseños que capturen la idea de un sistema para comunicárselo de una forma fácil de comprender a quien realice el proceso de desarrollo.
Existen diversas herramientas que permiten la realización de diagramas UML e integrarlos en un modelo de diseño. Los más notables son Select Enterprise, Visual UML y Rational Rose que es la utilizada a lo largo del presente artículo.

Tipos de Diagramas

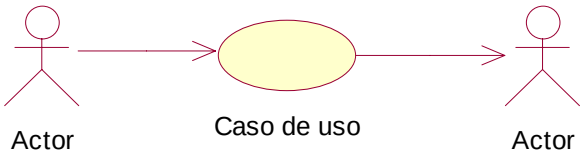
UML (lenguaje unificado de modelado) está compuesto de diversos elementos gráficos que se combinan para conformar diagramas. El objetivo de estos diagramas es simplificar un sistema presentando diversas perspectivas del mismo, a lo que se conoce como modelo.
El modelo visual creado permite simplificar la complejidad de los sistemas a analizar o diseñar de modo que sea comprendido fácilmente por todas las personas que intervienen en el proceso de desarrollo.

- **Diagrama de Casos de Uso**

Un caso de uso es una descripción de las acciones de un sistema desde el punto de vista de un usuario. Se representa la interacción con el sistema a desarrollar donde se muestran los requisitos funcionales.

Cada caso de uso, que representa el sistema, es una colección de situaciones y cada una de estas una secuencia de pasos. A las entidades que inician las secuencias se las conoce como **actores.**
Hay un actor que inicia un caso de uso y otro (o él mismo) que recibirá la acción que se ha generado. En la representación gráfica, el actor que inicia se encuentra a la derecha del caso de uso, indicado mediante una elipse, y el que

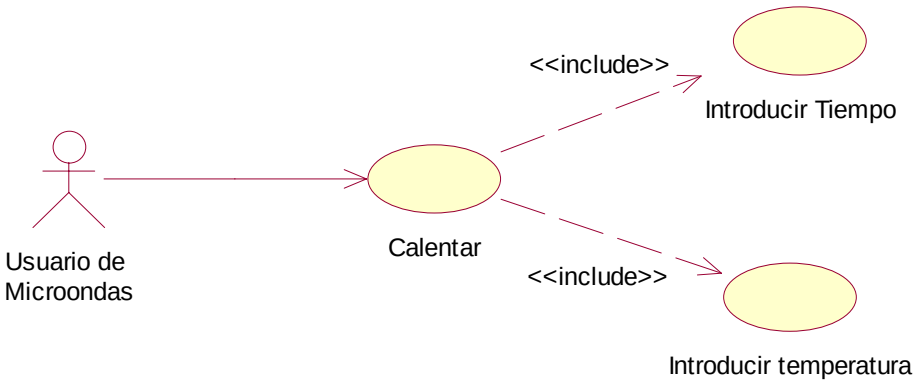
recibe, a la derecha. Tanto el nombre del actor como el del caso de uso, deberán mostrarse en el diagrama. Una línea de trazo continuo, conecta el caso de uso con los actores.



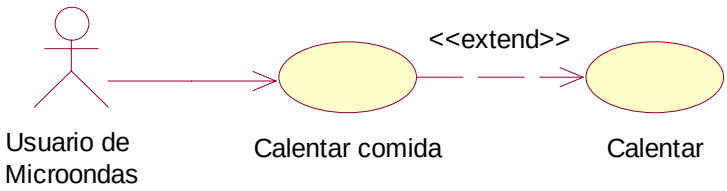
Relaciones entre casos de uso:

Los casos de uso se pueden relacionar entre si. Una relación que aparece es la de **inclusión**, en la que un caso de uso utiliza los pasos de otro.

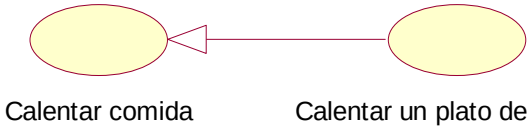
Para representar la inclusión, se utilizará una línea discontinua terminada en punta de flecha en un extremo, y una etiqueta <<include>>.



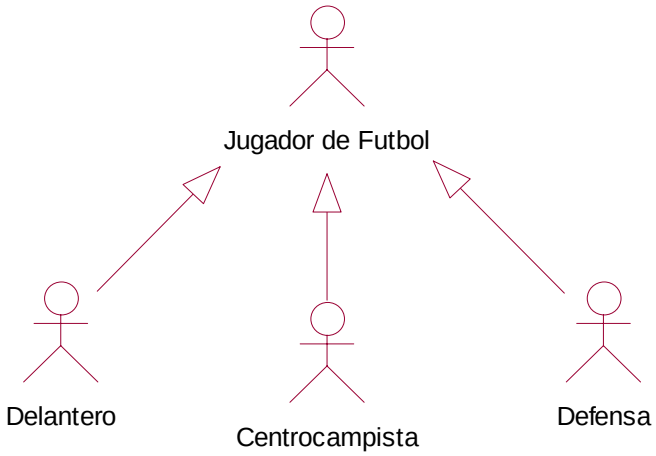
Otra relación entre los casos de uso es la **extensión** que permite crear un caso de uso mediante la adición de pasos a uno existente, denominado caso de uso base. La representación es análoga a la dada en la relación de inclusión, dónde la etiqueta es la <<extends>>.



Si un caso de uso hereda de otro, se dice que tienen una relación de **generalización**. El caso de uso secundario hereda las acciones y significado del primario, y además agrega sus propias acciones. Se representa con una línea continua, que parte del caso de uso secundario, terminado en una punta de flecha sin relleno apuntando al primario.



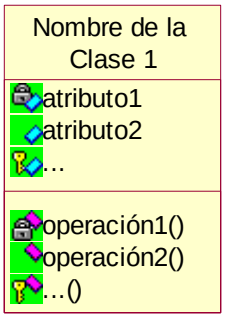
La relación de generalización también puede darse entre actores.



- **Diagrama de clases**

Una clase es una categoría o grupo de cosas que tienen atributos (propiedades) y que realizan determinadas acciones.
Un diagrama de clases representa el sistema a través de un conjunto de clases y sus relaciones.

En UML, una **Clase** es representada por un rectángulo que posee tres áreas:



Área Superior: Contiene el **nombre** de la Clase
Área Intermedia: Contiene los **atributos** (o propiedades) que caracterizan a la Clase, pueden ser de tres tipos, según el grado de comunicación y visibilidad de ellos con el entorno:
public (): Indica que el atributo será visible tanto dentro como fuera de la clase, es decir, es accesible desde otras clases.
private (): Indica que el atributo sólo será accesible desde dentro de la clase (sólo sus métodos lo pueden usar).
protected (): Indica que el atributo no será accesible desde fuera de la clase, pero si podrá ser usado por métodos de la clase además de las subclases que se deriven.

Área Inferior: Contiene los **métodos** (u operaciones), los cuales definen la forma de interactuar de la Clase con su entorno. Dependiendo de la visibilidad, los métodos pueden ser:

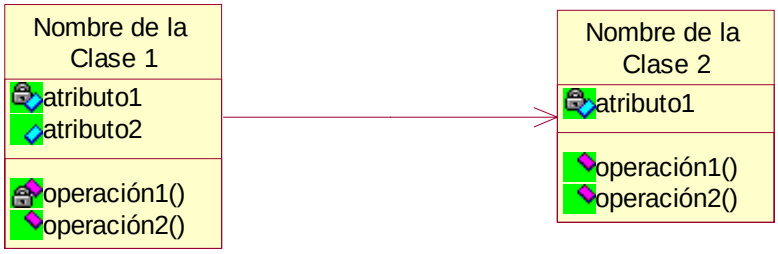
public (): Indica que el método será visible tanto dentro como fuera de la clase, es decir, es usado por otras clases.

private (): Indica que el método sólo será accesible desde dentro de la clase (sólo otros métodos de la clase lo pueden usar).

protected (): Indica que el método no será accesible desde fuera de la clase, pero si podrá ser usado por métodos de la clase además de métodos de las subclases que se deriven (ver herencia).

Relaciones entre Clases:

Cuando las clases se conectan entre si, esta conexión se conoce como **asociación**. En el diagrama de clases, la asociación se indica con una flecha de una clase a otra.



Dentro de una relación de asociación, cada clase juega un papel, que se indica en la parte superior de la línea que conecta a dichas clases.

Una asociación puede funcionar de forma inversa, en el ejemplo, vemos que un trabajador se encuentra empleado en una empresa. Pero también es cierto, que esa empresa, emplea a dicho trabajador.

Las asociaciones pueden ser más complejas, interviniendo más de una clase.

La asociación trazada entre el trabajador y la empresa, sugiere que las dos clases tienen una relación de uno a uno. No obstante, no es lo normal, una empresa cuenta con más trabajadores. Es decir que un trabajador pertenece sólo a una empresa, mientras que una empresa contiene varios trabajadores. Estas especificaciones son ejemplos de multiplicidad, que indica la cantidad de objetos de una clase que se relacionan con un objeto de la clase asociada.

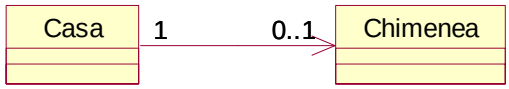
Hay varios tipos de multiplicidades:

uno a uno:

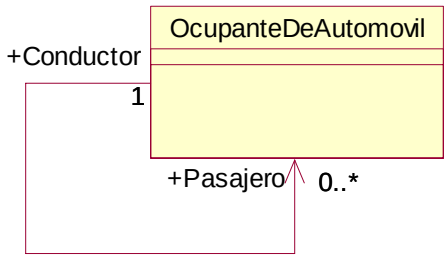
uno a muchos:

uno a uno o más:

uno a ninguno o uno:

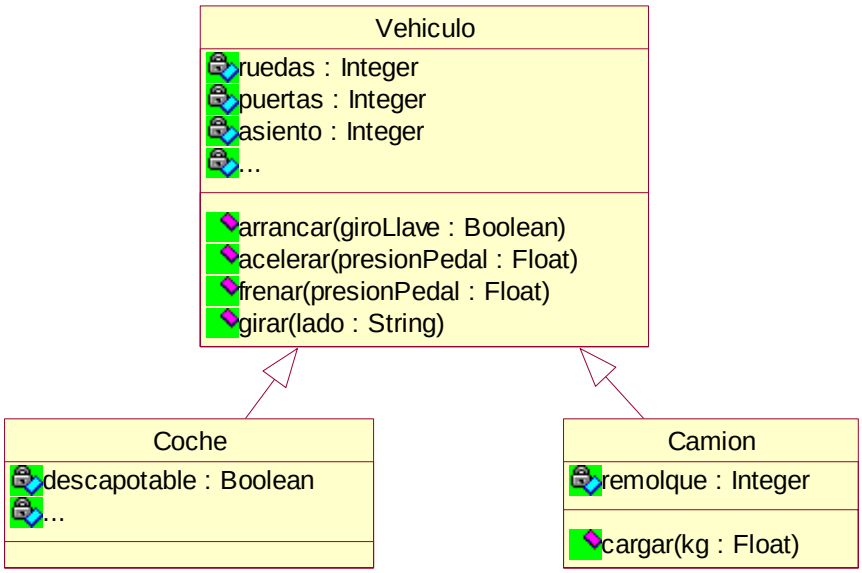


En ocasiones, una clase es una asociación consigo misma. Se denominan *asociaciones reflexivas* y se dan cuando una clase tiene objetos que pueden jugar diversos papeles. Un ocupanteDeAutomovil puede ser un conductor o un pasajero. En el papel de conductor, un ocupanteDeAutomovil puede llevar ninguno o más ocupanteDeAutomovil.



Una de las características fundamentales de la orientación a objetos es la **herencia** (en UML también denominada **generalización),** indica que una clase (clase secundaria o *subclase*) hereda los atributos y métodos de otra (clase principal o *superclase*). La superclase es más genérica y por tanto, la subclase además de poseer sus propios métodos y atributos, poseerá las características y atributos visibles de la superclase (public y protected).

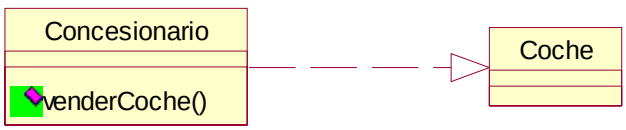
En UML, se representa la herencia con una línea que conecta la clase principal con la clase secundaria. En el extremo de la línea próximo a la clase principal, se encuentra una flecha sin relleno, apuntando a dicha clase principal.



En la figura se especifica que un Coche y un Camión heredan de Vehículo, es decir, coche posee las Características de Vehículo (precio, velMax, etc.) además posee algo particular que puede ser descapotable, en cambio Camión también hereda las características de Vehículo (precio, velMax, etc.) pero posee como particularidad propia, tara y carga.

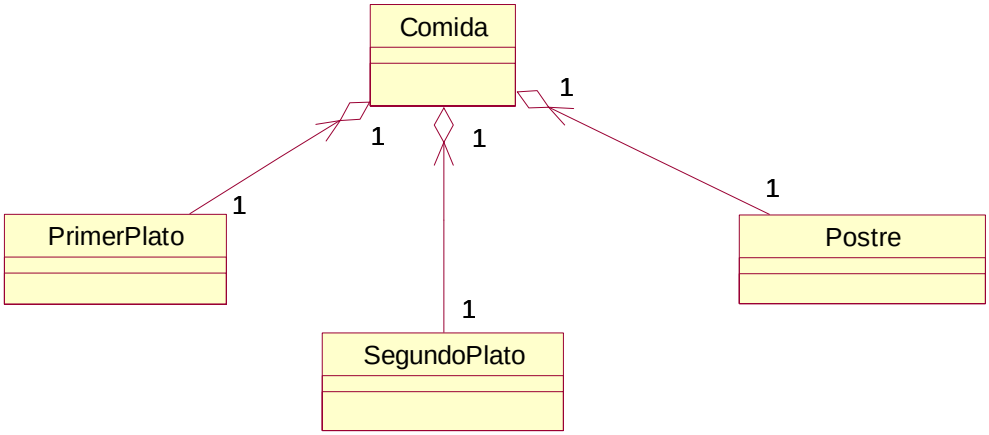
Una clase puede no provenir de una clase principal, en cuyo caso será una *clase base* o *clase raíz*. Así mismo, una clase podría no tener clases secundarias, denominándose *clase final* o *clase hoja.* Si una clase tiene una única clase principal, se dice que tiene herencia simple, mientras que tendrá herencia múltiple, si proviene de varias clases principales.

En otro tipo de relación, una clase utiliza a otra. A esta relación se le denomina **dependencia.** Se representa por una línea discontinua, que parte de una clase, y termina en flecha sin relleno en el extremo de la clase de la que depende.



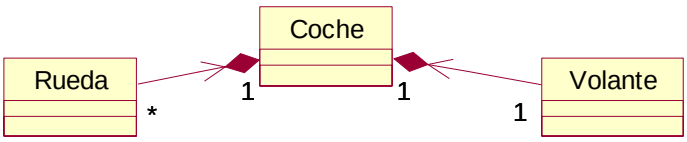
Existe un tipo especial de relación denominada **agregación** o **acumulación** que se da cuando una clase consta de otras clases.

Veamos como ejemplo, varias clases: PrimerPlato, SegundoPlato y Postre. Todas ellas se relacionan con una clase comida como una relación de agregación, es decir, que la clase Comida se compone de las clases mencionadas

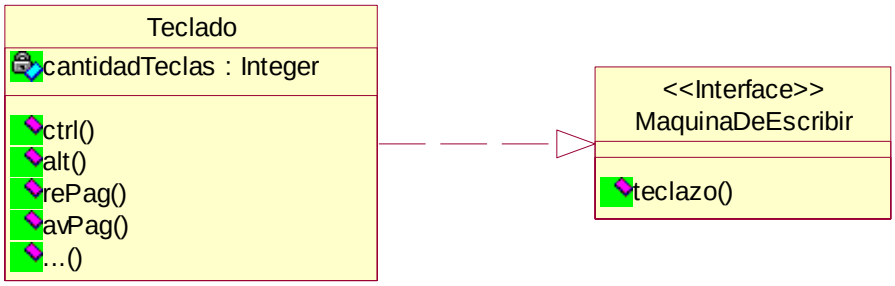


Una relación de agregación se representa por una línea que parte de los componentes y finaliza con un rombo sin relleno en el extremo más cercano al todo.

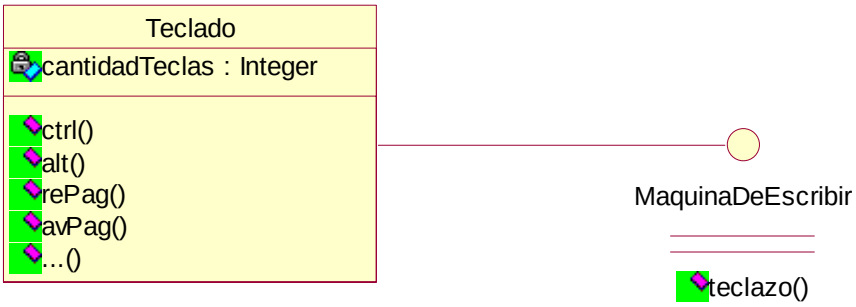
Una **composición** es un tipo muy representativo de una agregación. En dicha relación, cada componente pertenece solamente a un todo.



Una **interfaz** es un conjunto de operaciones que especifica cierto aspecto de la funcionalidad de una clase, y es un conjunto de operaciones que una clase presenta a otras.
Un interfaz no tiene atributos, sus métodos son abstractos, es decir, que no se implementan dentro de la interfaz, sino dentro de la clase que utiliza dicha operación. La utilidad de esto es que varias clases pueden utilizar este método creado por una clase e introducida en el interfaz.
Se representa como una clase pero indicando la palabra <<interfaz>> antes del nombre. En los diagramas UML, de la clase que utiliza el interfaz parte una línea discontinua acabada en flecha sin relleno, en el extremo más próximo al interfaz.

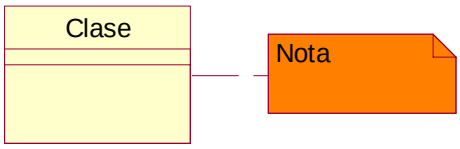


o



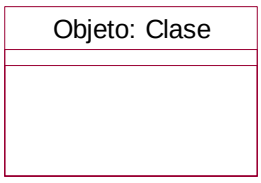
Destacar que una clase puede realizar más de un interfaz, y éste puede ser realizado por más de una clase.

Dentro del lenguaje unificado de modelado, y con el objetivo de aclarar aquellos puntos que el diseñador considera oportunos, aparecen las *notas*, que cuelgan de las clases y que pueden relacionar diagramas. Es posible asociar a una nota otro diagrama, de modo que se simplifica la compresión de diagramas de gran tamaño, pudiéndose dividir en varios que se relacionen mediante notas.



- **Diagrama de objetos**

Un objeto es una instancia de la clase, dicho de otra manera, una entidad con valores específicos de sus atributos y operaciones. (Nosotros somos instancias de una clase persona)



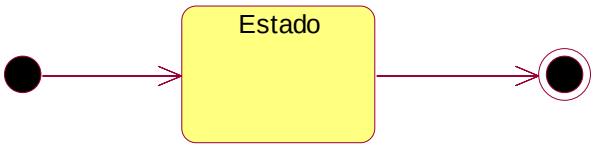
En UML, los objetos de representan mediante un rectángulo indicando el nombre del objeto seguido de dos puntos (:) y el nombre de la clase de la que se instancia.

- **Diagrama de estados**

Un objeto, en un momento en particular, se encuentra en un estado definido, al igual que una persona es recién nacida, infante, adolescente, joven o adulta.
Un diagrama de estados representa los estados en que puede encontrarse un objeto, junto con las transiciones entre los estados.

Es necesario contar con diagramas de estados, ya que permiten a los analistas, diseñadores y programadores comprender el comportamiento de los objetos del sistema. Mientras que los diagramas de clases y de objetos muestran los aspectos estáticos, los diagramas de estados representan el comportamiento dinámico del sistema.
Fundamentalmente los desarrolladores deben saber como los objetos se comportarán, debido a que serán ellos quienes tendrán que reflejar tales comportamientos en el software.

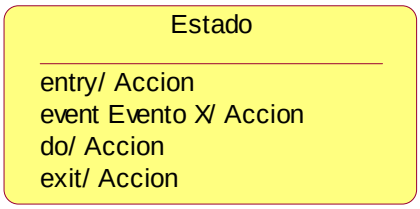
La representación es la siguiente:



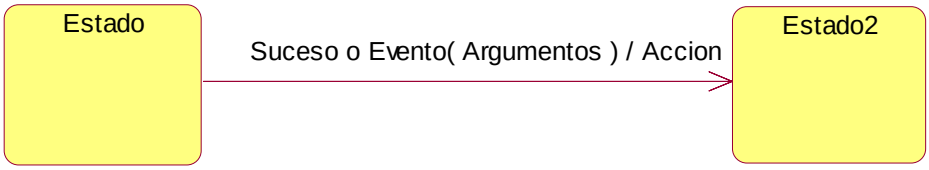
El punto relleno representa el estado inicial, de todos los que presentará el objeto. De este parte una flecha que indica transición hacia otro simbolizado mediante un rectángulo de esquinas redondeadas. Para finalizar se encuentra una diana que representa el estado final del objeto.

En UML al icono de estado (rectángulo redondeado), es posible agregarle más información con el objetivo de tener un diagrama más completo. Se puede dividir el símbolo en tres áreas, (nombre, atributos y operaciones). El área superior mostrará el nombre del estado, el área central las variables de estado, mientras que el área inferior contendrá las actividades u operaciones.

Las operaciones que actúan sobre el estado, deben servir de aclaración sobre la transición que sufre el objeto, por lo que se pueden dividir en tres fundamentales: *entry* (que sucede cuando el sistema entra al estado), *do* (que ocurre cuando el sistema está en el estado), *event* (acción que aparece al darse un determinado evento dentro del estado) y *exit* (que sucede cuando el sistema sale del estado).



También es posible agregar detalles a las líneas de transición, tales como un *suceso* que provoque una transición, y la *acción* que se ejecuta y haga que se lleve a cabo la modificación del estado. Los sucesos y acciones se representarán junto a la línea que indica la transición y separados de una barra diagonal

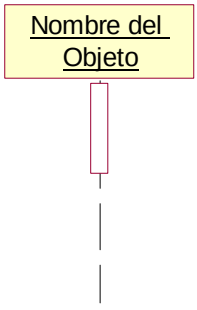


En ocasiones un evento causará una transición sin acción asociada y algunas veces una transición se dará sin ser causada por un suceso sino porque finalizará una actividad. A estos tipos de relaciones se les conoce como *transición no desencadenada*.

- **Diagrama de secuencia**

Los diagramas de objetos y de clases, muestran el sistema desde un punto de vista estático, sin embargo en un sistema funcional los objetos interactúan entre si a lo largo del tiempo.
Los diagramas de estados se refieren a las transiciones que sufre un objeto, mientras que en UML los diagramas de secuencias muestra la forma en que un objeto interactúa con otros en base a los tiempos.

En los diagramas de secuencia, los **objetos** se representan con rectángulos con nombre subrayado. Se sitúan en la parte superior del diagrama, de izquierda a derecha. Debajo del rectángulo, de forma descendente, parte una línea discontinua que representa *la línea de vida* del objeto. Junto a la línea de vida se encuentra un rectángulo en posición vertical, que representa la ejecución de una operación por parte del objeto, conocido como a*ctivación*. La longitud del rectángulo indica la duración de la activación del objeto.



Los **mensajes** parten de un objeto a otro, o a él mismo, indicando que objeto entra en ejecución en el instante de tiempo. Pueden ser simples, sincrónico, o asincrónico.
Un mensaje simple es la transferencia de control de un objeto a otro. Si el mensaje es sincrónico, el objeto que lo envía esperará respuesta antes de continuar con su ejecución. Si, por el contrario, el objeto envía un mensaje asincrónico, no esperará respuesta antes de continuar.
En el diagrama de secuencias, se representan por una línea terminada en punta de flecha que parte de la línea de vida de un objeto apuntando a la de otro objeto (o la suya propia).

El diagrama representa el **tiempo** en dirección vertical, comenzando en la parte superior y avanzando hacia la parte inferior. Por tanto, un mensaje que esté más cerca de la parte superior ocurrirá antes que otro que esté cerca de la parte inferior.

Así, el diagrama de secuencias tiene dos dimensiones, la vertical que muestra el paso del tiempo, y la horizontal que indica la disposición de los objetos.

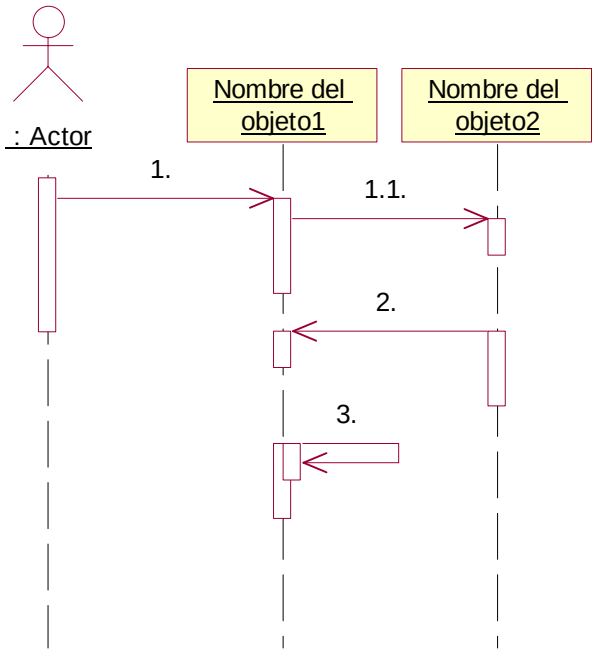
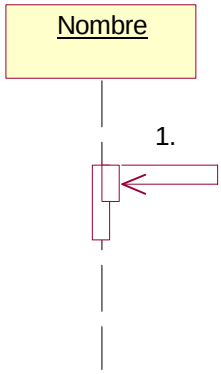


Figura: los objetos, dentro del diagrama de secuencias, se colocan de izquierda a derecha en la parte superior. Cada línea de vida de un objeto es una línea discontinua que se desplaza hacia abajo del objeto. Una línea continua terminada en punta de flecha, que representa un mensaje entre objetos, conecta una línea de vida con otra. El tiempo se inicia en la parte superior y continúa hacia abajo.

En ocasiones un objeto cuenta con una operación que se invoca a sí misma. A esto se le conoce como **recursividad**. En UML se representa mediante una flecha que parte de la activación, indicando la operación, que apunta a un rectángulo pequeño sobrepuesto en la activación.

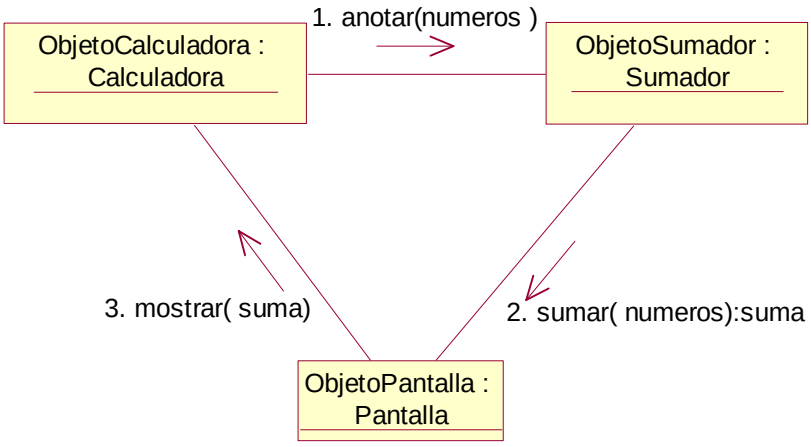


- **Diagrama de colaboraciones**

Los diagramas de colaboraciones muestran la forma en que los objetos colaboran entre si, tal y como sucede en el diagrama de secuencias. Ambos diagramas son semánticamente equivalentes, esto significa que proporcionan la misma información y se podría convertir uno en el otro.

Los diagramas de secuencias destacan la sucesión de las interacciones, constituyendo una organización con respecto al tiempo. Los diagramas de colaboraciones destacan el contexto y la organización general de los objetos que interactúan., organizándose de con respecto al espacio. Esta diferencia que presentan, hace útil contar con ambas representaciones del sistema a la hora de afrontar un proceso de desarrollo.

Un diagrama de colaboraciones muestra los objetos como tales y sus relaciones entre sí. Así mismo, indica los mensajes que se envían los objetos.



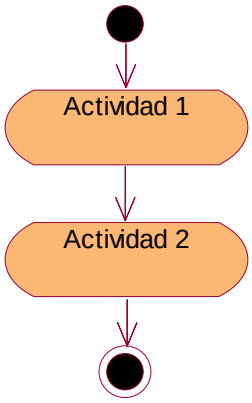
Los mensajes se representan con una flecha cerca de la línea que asocia los objetos, esta flecha apunta al objeto receptor de la operación. El mensaje se etiqueta, indicando al objeto receptor la operación que debe realizar. El nombre de la operación se representa seguido de paréntesis en donde aparecerán los parámetros (en caso de haber alguno) con los que funcionará el método.

- **Diagrama de actividades**

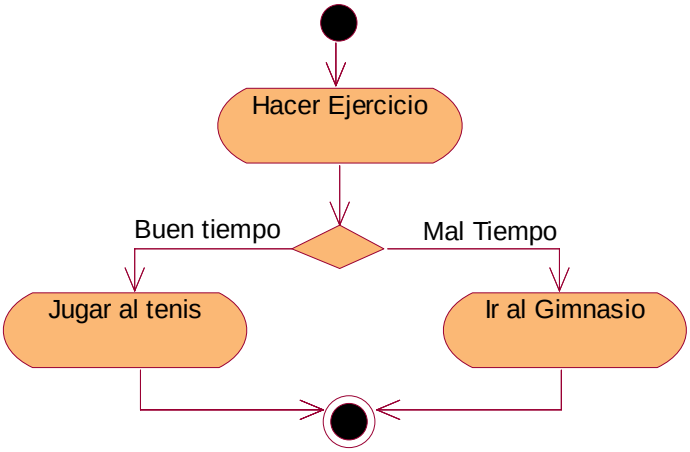
Uno de los primeros métodos visuales aplicados a la programación son los diagrama de flujo, muestra una secuencia de pasos, procesos, puntos de decisión y bifurcaciones. Estos pretenden facilitar la comprensión del sistema, entendiendo todos y cada uno de los pasos y caminos que se pueden dar. Éste diagrama es convertido posteriormente en código.
El diagrama de actividades de UML, es muy parecido a los diagramas de flujo, muestra los pasos (conocidos como actividades) en una operación o proceso, así como los puntos de decisión y bifurcaciones.

Si volvemos un momento atrás, podemos ver que los diagramas de estados muestran los estados de un objeto y las actividades representadas mediante flechas entre los estados. El diagrama de actividades extiende el diagrama de estados, refinando lo que ocurre dentro de cada actividad.

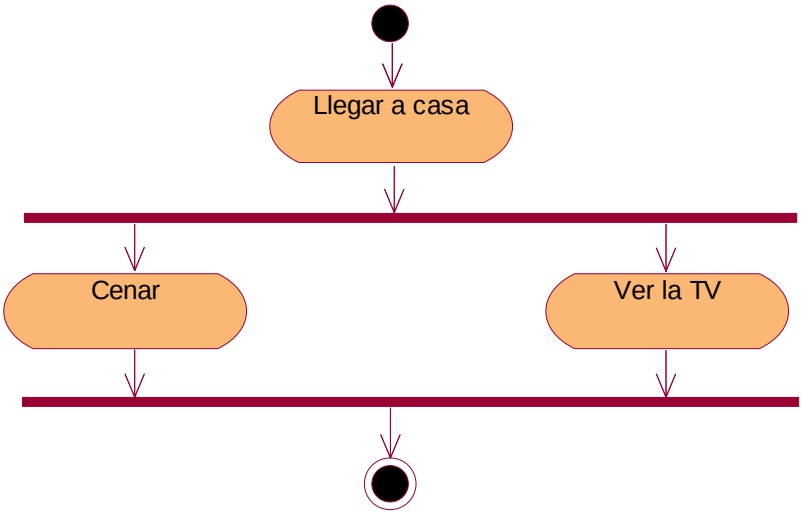
Cada actividad se representa mediante un rectángulo de esquinas redondeadas (más alargado y ovalado que la representación de un estado). La transición a otra actividad viene indicada con una flecha. Un punto inicial (círculo relleno) y un punto final (diana), completan los símbolos utilizados en el diagrama de actividades.



Dentro de una secuencia de actividades, frecuentemente se llegará a un punto donde se realizará alguna decisión. Un **punto de decisión** se representa mediante un rombo que indica decisión. Entre corchetes se indicará la condición junto a la ruta correspondiente.



Los **eventos** indican el suceso que ocurre en la transición de una actividad a otra. En el diagrama, se indican encima de las conexiones entre actividades (Buen tiempo y Mal tiempo en el ejemplo).
En ocasiones, dos actividades se realizan al mismo tiempo y se resumen en el mismo punto, denominándose **concurrentes.** Se representan mediante una línea gruesa perpendicular a la transición de donde parten ambas rutas, que apuntan a una nueva línea gruesa en donde finalizan.



- **Diagramas de componentes**

Los diagramas UML anteriores, representan el sistema desde el punto de vista conceptual, es decir, la abstracción de elementos. Existe un diagrama que representa a una entidad real, un componente de software.
Se define un componente de software como una parte física de un sistema, y se encuentra en la computadora, no en la mente del analista.

Un componente sería, por ejemplo, la representación dentro del software (código) de una clase. La clase se trata de una definición de atributos y operaciones, mientras que el componente es la realización física de estos. Un componente puede estar formado por varias clases.

Uno de los puntos fundamentales de la programación orientada a objetos es la reutilización. Los componentes se crean con el objetivo de poder ser reutilizados en diversos sistemas, de modo que se agilice la realización de nuevo software.

Componentes e interfaces

Definimos interfaz como un conjunto de operaciones de una clase que son utilizadas por otras. De igual modo, un componente (implementación de software de una clase) utilizará un interfaz, y sólo a través de éste se podrán ejecutar sus operaciones.
La relación entre un componente y su interfaz se conoce como *realización*.

Un componente podrá hacer disponible su interfaz (*interfaz de exportación*) para que otros componentes puedan utilizar las operaciones que contiene. Dicho de otra manera, un componente puede acceder a los servicios de otro componente (*interfaz de importación*).

Sustitución y reutilización

Las interfaces son fundamentales dentro del modelado, ya que son primordiales en la reutilización y sustitución.
Un componente se puede sustituir por otro, si el nuevo contiene las mismas interfaces que el antiguo. Esto es posible debido a que los demás componentes no notan el cambio, pues se comunican entre si a través de su interfaz.
La reutilización de un componente en otro sistema es posible, si éste puede acceder al componente reutilizado mediante sus interfaces. Es por tanto, muy importante un diseño orientado a la reutilización, de modo que pequeños cambios en sus interfaces hagan reutilizables los componentes, con el considerable ahorro de tiempo para el desarrollador (ahorro en implementación de código que debe ser integrado y probado).

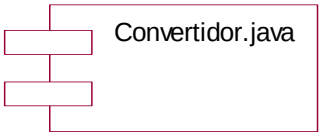
Tipos de componentes

1. *Componentes de distribución*, que conforman el fundamento de los sistemas ejecutables (dll, ejecutables, controles Active X, Java Beans, etc.).
2. *Componentes para trabajar en el producto* (archivos de Base de datos y de código)
3. *Componentes de ejecución*, creados como consecuencia de un sistema de ejecución.

Representación en UML

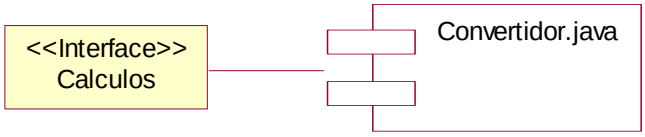
Los diagramas de componentes UML contienen, obviamente, componentes, interfaces y sus relaciones.

Un **componente** se representa mediante un rectángulo que tiene otros dos sobrepuestos en su lado izquierdo.

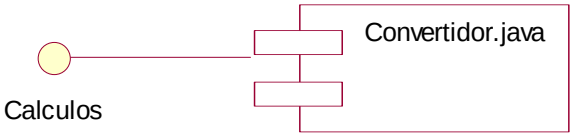


Se puede agregar al símbolo detalles sobre el componentes, como el nombre del paquete al que pertenece, las clases que contiene, así como el tipo de componente (Applet, Active X, …etc.).

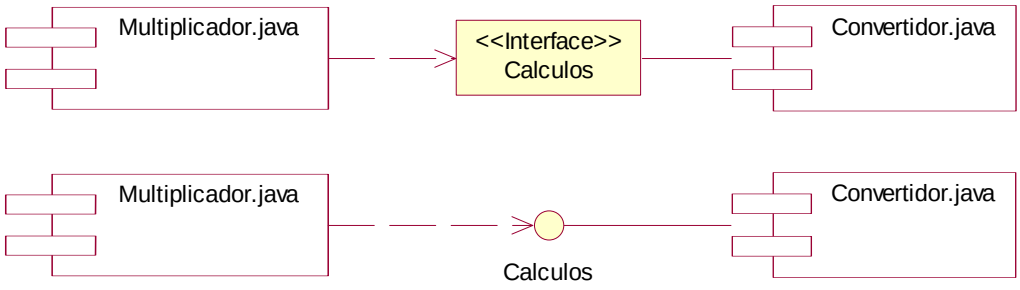
Las **interfaces** se representan con rectángulos conectados al componente mediante línea continua.



Otra representación de la interfaz es un círculo.



Bajo ambas representaciones, se puede incluir la relación con otros componentes, a través de la interfaz de importación.

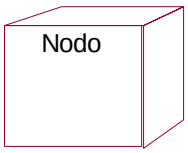


- **Diagramas de distribución**

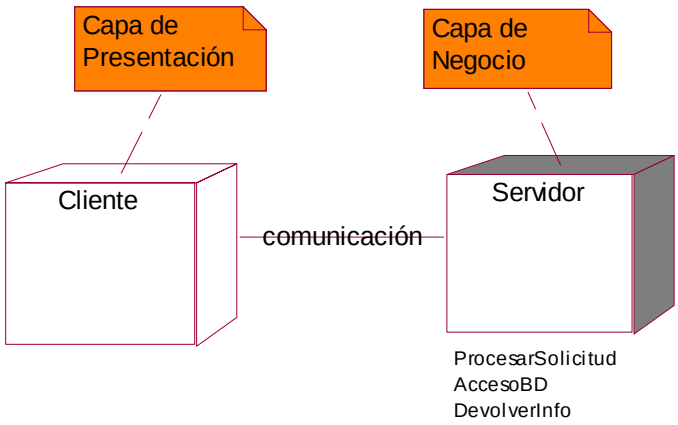
Todo sistema está compuesto de hardware, siendo básico para un buen diseño del sistema, un diseño sólido de distribución del hardware.
A los elementos de hardware se les denomina de forma genérica **nodos**. Dentro del hardware, se pueden usar dos tipos de nodos:

- Un procesador, el cual puede ejecutar un componente.
- Un dispositivo (impresora, monitor, etc.), que no ejecuta componentes, pero que tiene la función de interacción con el mundo exterior.

En UML, un nodo viene representado por un cubo su nombre.



Así mismo, puede contener más información, tal como el nombre del paquete al que pertenece, componentes situados en dicho nodo, etc.
La relación entre los elementos de hardware o nodos, se representa mediante una línea que une los cubos en donde se puede utilizar el estereotipo de conexión que se da.

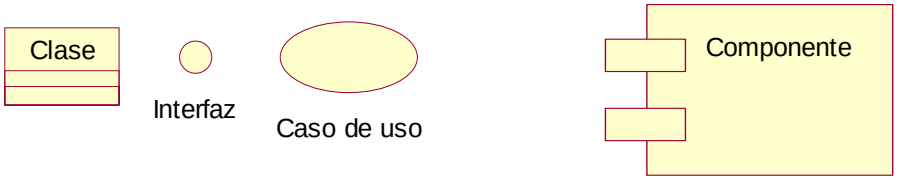


Resumen

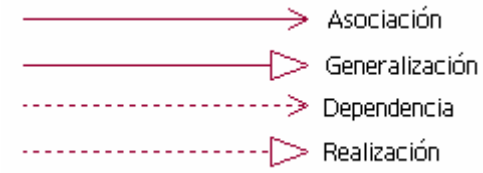
El objetivo de este artículo es introducir a los lectores al diseño de sistemas basado en UML, puesto que es el futuro del diseño orientado a objetos y por tanto es de imprescindible conocimiento no sólo para poder realizar un diseño robusto de un sistema, sino para poder entenderlo y a partir de él afrontar un proceso de desarrollo (programación).

REPASEMOS…

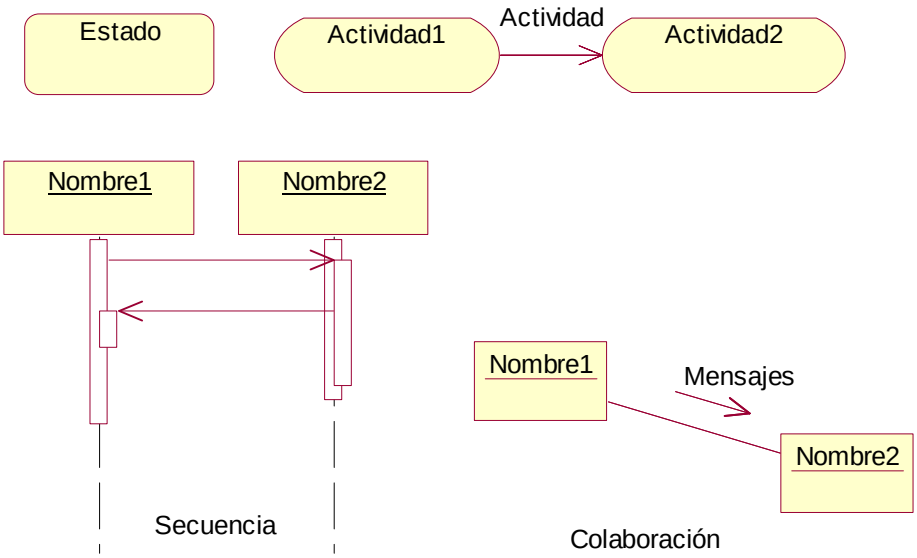
Elementos Estructurales



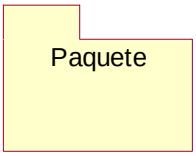
Relaciones



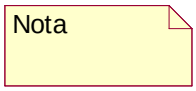
Elementos de comportamiento



Agrupaciones



Anotación



Bibliografía

[1] ***El lenguaje Unificado de Modelado***. G. Booch, J. Rumbaught, I. Jacobson. (Addison Wesley Iberoamericana, 1999).

[2] ***UML y Patrones***. C. Larman. (Prentice Hall, 1999).

[3] ***Aprendiendo UML en 24 horas***. Josep Schumer (Prentice Hall, 2000).

[4] ***UML Resource Center***. Rational Software. (http://www.rational.com/uml).

Ignacio García-Caro
Vicepresidente de la Rama de Estudiantes del IEEE-UNED.
Estudiante de Ingeniería Industrial.
nachogcg@hotmail.com

INFORMACIÓN GENERAL RESUMIDA

La Rama de Estudiantes creada en la Universidad Nacional de Educación a Distancia (UNED) tiene por objetivo principal **la difusión de la ciencia y la tecnología**.

Se consolidó inicialmente con 37 miembros en noviembre del año 2004.

La información general sobre sus actividades e información de cómo hacerse miembro se puede ver en la página Web:

http://www.ieec.uned.es/IEEE/

dentro del enlace de la Rama de Estudiantes.

Las actividades principales que las Ramas de España realizan son: charlas, cursos, congresos, concursos, actividades educativas, visitas a empresas y organizaciones, interrelación cultural y multidisciplinar y cualquier actividad que quiera desarrollar cada uno de sus miembros.

Actualmente puede participar cualquier estudiante de las carreras de Informática y de Industriales de la UNED. Para conocer más información sobre el IEEE, las Ramas de España y sus posibilidades se recomienda leer los primeros artículos de éste Boletín y visitar la página Web para ver lso boletines previos. De todas formas cualquier información o consulta puede dirigirse a Eugenio López:

elopez@ieec.uned.es.

Esperamos que os haya gustado a todos éste cuarto Boletín y agradecer una vez más a todos los autores el haber participado en el mismo haciéndolo posible.

UN SALUDO

Eugenio López
Presidente de la Rama de Estudiantes IEEE-UNED

Hazte socio
De la Rama de Estudiantes
del IEEE en la UNED

Web IEEE-UNED

http://www.ieec.uned.es/IEEE/

Charlas, conferencias,
cursos, visitas, empresa,
Boletín Electrónico, etc.

RAMA DE ESTUDIANTES IEEE-UNED
31-ENERO-2006 (BOLETIN Nº4)