

Novática, revista fundada en 1975, es el órgano oficial de expresión y formación continua de ATI (Asociación de Técnicos de Informática)

ATI es miembro de CEPIS (*Council of European Professional Informatics Societies*) y tiene un acuerdo de colaboración con ACM (*Association for Computing Machinery*). Tiene asimismo acuerdos de vinculación o colaboración con AdaSpain, AI² y ASTIC

<http://www.ati.es/novatica/>

CONSEJO ASESOR DE MEDIOS DE COMUNICACION

Pere Lluís Barbarà, Rafael Fernández Calvo, José Gómez, Manuel Orti Mezquita, Nacho Navarro, Fernando Sanjuán de la Rocha (Presidente), Miquel Sarries, Carlos Sobrino Sánchez, Manuel Solans

Coordinación Editorial
Rafael Fernández Calvo <rfcalvo@ati.es>

Composición y autoedición
Jorge Llácer

Administración
Tomás Brunete, Joan Aguiar, María José Fernández

SECCIONES TÉCNICAS: COORDINADORES

Arquitecturas
Antonio González Colás (DAC-UPC) <antonio@ac.upc.es>

Bases de Datos
Mario G. Plattini Velthuis (EUI-UCLM) <mpiattini@inf-cr.uclm.es>

Calidad del Software
Juan Carlos Granja (Universidad de Granada) <jcgranja@goliat.ugr.es>

Derecho y Tecnologías
Isabel Hernández Collazos (Fac. Derecho de Donostia, UPV) <ihernando@legaltek.net>

Enseñanza Universitaria de la Informática
Cristóbal Pareja Flores (Dep. Sistemas Informáticos y Programación-UCM) <cpareja@sis.ucm.es>

Euro/Efecto 2000
Joaquín Ríos Boutin <jrios@ati.es>

Informática Gráfica
Roberto Vivó (Eurographics, sección española) <rvivo@dsic.upv.es>

Informática Médica
Valentín Masero Vargas (DI-UNEX) <vmasero@umex.es>

Ingeniería del Software
Luís Fernández (PRIS-EI/UEM) <lufern@dpri.esi.uem.es>

Inteligencia Artificial
Federico Barber, Vicente Botti (DSIC-UPV) <fjbotti_fbarber@dsic.upv.es>

Interacción Persona-Computador
Julio Abascal González (FI-UPV) <julio@si.ehu.es>

Internet
Alonso Álvarez García (TID) <alonso@ati.es>

Llorenç Pagés Casas (Atlante) <pages@ati.es>

Lengua e Informática
M. del Carmen Ugarte (IBM) <cugarte@ati.es>

Lenguajes informáticos
Andrés Marín López (Univ. Carlos III) <amarin@it.uc3m.es>

J. Ángel Velázquez (ESCET-URJC) <a.velazquez@escet.urjc.es>

Libertades e Informática
Alfonso Escolano (FIR-Univ. de La Laguna) <aescolan@ull.es>

Lingüística computacional
Xavier Gómez Guinovart (Univ. de Vigo) <gomez@uvigo.es>

Manuel Palomar (Univ. de Alicante) <mpalomar@dsi.ua.es>

Profesión informática
Rafael Fernández Calvo (ATI) <rfcalvo@ati.es>

Miquel Sarries Grinyó (Ayto. de Barcelona) <msarries@ati.es>

Seguridad
Javier Areitio (Redes y Sistemas, Bilbao) <jareitio@orion.deusto.es>

Sistemas de Tiempo Real
Alejandro Alonso, Juan Antonio de la Puente (DIT-UPM) <jaalonso.jpuede@dit.upm.es>

Software Libre
Jesús M. González Barahona, Pedro de las Heras Quirós (GSYC, URJC) <jgibpheras@gsyc.escet.urjc.es>

Tecnología de Objetos
Esperanza Marcos (URJC) <e.marcos@escet.urjc.es>

Gustavo Rossi (LIFIA-UNLP, Argentina) <gustavo@sol.info.unpl.edu.ar>

Tecnologías para la Educación
Benita Compostela (F. CC. PP.-UCM) <benita@diad.umet.es>

Josep Sales Rufí (ESPIRAL) <jsales@pie.xtec.es>

Tecnologías y Empresa
Pablo Hernández Medrano <phmedrano@terra.es>

TIC para la Sanidad
Valentín Masero Vargas (DI-UNEX) <vmasero@umex.es>

Las opiniones expresadas por los autores son responsabilidad exclusiva de los mismos. Novática permite la reproducción de todos los artículos, salvo los marcados con © o *copyright*, debiéndose en todo caso citar su procedencia y enviar a Novática un ejemplar de la publicación.

Coordinación Editorial y Redacción Central (ATI Madrid)
Padilla 66, 3º, dcha., 28006 Madrid

Tel: 914029391; fax: 913093685 <novatica@ati.es>

Composición, Edición y Redacción ATI Valencia

Palomino 14, 2º, 46003 Valencia

Tel./fax: 963918531 <secreval@ati.es>

Administración, Suscripciones y Redacción ATI Cataluña

Via Laietana 41, 1º, 08003 Barcelona

Tel: 934125235; fax: 934127713 <secregen@ati.es>

Redacción ATI Andalucía

Isaac Newton, s/n, Ed. Sadiel, Isla Cartuja 41092 Sevilla

Tel./fax: 954460779 <secreand@ati.es>

Redacción ATI Aragón

Lagasca 9, 3-B, 50006 Zaragoza

Tel./fax: 976235181 <secreara@ati.es>

Redacción ATI Asturias-Cantabria <gp-astucant@ati.es>

Redacción ATI Castilla-La Mancha <gp-clmancha@ati.es>

Redacción ATI Galicia

Recinto Ferial s/n, 36540 Silleda (Pontevedra)

Tel: 986581413; fax: 986580162 <secregal@ati.es>

Publicidad: Padilla 66, 3º, dcha., 28006 Madrid

Tel: 914029391; fax: 913093685 <novatica.publicidad@ati.es>

Imprenta: Gráficas Sierra S.L., Atenas, 3, int. bajos, 08006 Barcelona.

Depósito Legal: B 15.154-1975

ISBN: 0211-2124; CODEN NOVATEC

Portada: Antonio Crespo Foix / © ATI 2001

SUMARIO

En resumen: **Libertad y madurez** 2

NOVIEMBRE - DICIEMBRE 2001

154

Monografía: «Software Libre/Fuente Abierta: hacia la madurez»
(En colaboración con *Informatik/Informatique* y *Upgrade*)
Coordinada por *Joe Ammann, Jesús M. González-Barahona y Pedro de las Heras Quirós*

Presentación: hacia la madurez 3
Joe Ammann, Jesús M. González-Barahona, Pedro de las Heras Quirós

Actualidad del software libre 5
Pedro de las Heras Quirós, Jesús M. González-Barahona

Eldaño viene de La Haya 14
Richard Stallman

Iniciativas europeas sobre el uso de software libre en el Sector Público 17
Juan Jesús Muñoz Esteban

Open Source en un gran banco suizo 22
Klaus Bucka-Lassen, Jan Sorensen

El proyecto GNU Enterprise: software de aplicación para la empresa 25
Neil Tiffin, Reinhard Müller

Contando patatas: el tamaño de Debian 2.2 30
Jesús M. González-Barahona, Miguel A. Ortuño, Pedro de las Heras, José Centeno, Vicente Matellán

La crisis del software libre científico 38
David Santo Orcero

El proyecto Debian GNU/Linux 41
Javier Fernández-Sanguino Peña

Sistemas de ficheros con Journaling en Linux 45
Ricardo Galli

Secciones técnicas

Ingeniería del Software

Un nuevo modelo de evaluación de procesos de software para PYMES a partir de SPICE (ISO/IEC TR-15504-5) 52
Antonia Mas Pichaco, Ángel Igelmo Ganzo, Esperança Amengual Alcover, Gabriel Fontanet Nadal

Profesión informática

El futuro de la Ingeniería del Software 57
Karol Frühauf

Seguridad

De mí misma libremente Dios, que del Sircam ya me libro yo (y II) 59
Mª del Carmen Ugarte García

Tecnología de Objetos

¿Es conveniente la Orientación a Objetos en un primer curso de programación? 64
Jesús J. García Molina

Referencias autorizadas 69

Sociedad de la Información

Programar es crear

Ancho de banda en Internet 72
Concurso de Programación ACM 2000: programa E

«Fila y asociados»: solución 73
Álvaro Martínez Echevarría

Asuntos Interiores

Coordinación editorial / Programación de Novática 76

Normas de publicación para autores / Socios Institucionales 77

Monografía del próximo número: «Gestión del Conocimiento y TIC»

Programar es crear

Álvaro Martínez Echevarría

«Fila y asociados»: solución

<ame@acm.org>

El enunciado de este problema apareció en el número 153 de Novática (septiembre-octubre 2001, p. 70). Es el programa G de los planteados en el 24º Concurso Internacional de Programación de la ACM (2000)

La solución es muy sencilla: simplemente una simulación. Puesto que los parámetros propuestos son razonablemente pequeños (un cuanto de tiempo de un minuto durante un día, con un máximo de 20 temas y 5 personas), podemos permitirnos la fuerza bruta y avanzar de minuto en minuto. El código es algo aparatoso, como en toda simulación, pero el concepto es muy sencillo: describir el estado del sistema en cada momento e ir incrementando el tiempo en minutos.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <vector>
#include <hash_map>

struct topic {
    /* Peticiones que aún quedan por llegar */
    int remaining;
    /* Peticiones a la espera de atenderse */
    int waiting;
    /* Tiempo de llegada de la próxima petición */
    int next;
    /* Tiempo de servicio */
    int duration;
    /* Tiempo entre peticiones */
    int interval;
};

struct person {
    /* Identificación */
    int id;
    /* Temas bajo su responsabilidad */
    vector<int> topics;
    /* Tema del que se ocupa en este momento */
    int now;
    /* Tiempo de la última asignación */
    int last;
    /* Tiempo a partir del cual queda disponible */
    int available;
};

struct state {
    int t;
    hash_map<int,topic> requests;
    vector<person> personnel;

    /* Peticiones totales que quedan */
    int pending;
    /* Número de personas ocupadas */
    int busy;
};

void
simulate(state *st) {
    vector<int> available;

    /*
     * Inicialmente todos están disponibles.
     */
}
```

```

*/
for (unsigned int i=0;i<st->personnel.size();i++) {
    available.push_back(i);
}

while (1) {

    /*
    * Recibir las peticiones.
    */
    if (st->pending>0) {
        for (hash_map<int,topic>::iterator it=st->requests.begin();it!=st->requests.end();it++) {
            if (it->second.remaining>0 && st->t==it->second.next) {
                it->second.remaining--;
                it->second.waiting++;
                it->second.next+=it->second.interval;
            }
        }
    }

    /*
    * Liberar y ordenar el personal.
    */
    for (unsigned int i=0;i<st->personnel.size();i++) {
        if (st->personnel[ i ].now>=0 && st->t==st->personnel[ i ].available) {
            st->personnel[ i ].now=-1;
            st->busy--;

            /*
            * Insertamos 10*(1+última_asignación)+(orden), para que
            * al ordenar el vector el resultado sea el que queremos;
            * es decir, que el personal quede ordenado por:
            * (1) petición anterior menos reciente
            * (2) identificador
            */
            available.push_back(i+10*(st->personnel[ i ].last+1));
        }
    }
    sort(available.begin(),available.end());

    /*
    * Asignar las peticiones entre las personas disponibles.
    */
    if (st->pending>0) {
        hash_map<int,topic>::iterator topic;
        for (vector<int>::iterator it=available.begin();it<available.end();it++) {
            int who=(*it)%10;
            for (unsigned int i=0;i<st->personnel[ who ].topics.size();i++) {
                /*
                * ¿Existe ese tema? ¿Hay alguna pregunta esperando?
                * Si es así, asignarla.
                */
                if ((topic=st->requests.find(st->personnel[ who ].topics[ i ]))==
                    st->requests.end() || topic->second.waiting==0) {
                    continue;
                }
                st->personnel[ who ].last=st->t;
                st->personnel[ who ].available=st->t+topic->second.duration;
                st->personnel[ who ].now=topic->first;
                st->busy++;
                st->pending--;
                topic->second.waiting--;
                available.erase(it);
                break;
            }
        }
    }

    if (st->pending==0 && st->busy==0) {
        break;
    }
}

```

```

    }

    st->t++;

}

}

int
main(int argc, char *argv[]) {
    int scenario=1,a,b;
    state *st=new state();

    while (1) {
        if (scanf("%d",&a)!=1 || a==0) {
            break;
        } else if (a<0) {
            abort();
        }

        st->t=0;
        st->busy=0;
        st->pending=0;

        st->requests.clear();
        for (int i=0;i<a;i++) {
            if (scanf("%d",&b)!=1) {
                abort();
            }
            st->requests[ b ].waiting=0;
            if (scanf("%d %d %d %d",
                &st->requests[ b ].remaining,
                &st->requests[ b ].next,
                &st->requests[ b ].duration,
                &st->requests[ b ].interval)!=4) {
                abort();
            }
            st->pending+=st->requests[ b ].remaining;
        }

        if (scanf("%d",&a)!=1 || a<=0) {
            abort();
        }

        st->personnel.resize(a);
        for (int i=0;i<a;i++) {
            if (scanf("%d %d",&st->personnel[ i ].id,&b)!=2) {
                abort();
            }
            st->personnel[ i ].topics.resize(b);
            for (int j=0;j<b;j++) {
                if (scanf("%d",&st->personnel[ i ].topics[ j ]) !=1) {
                    abort();
                }
            }
            st->personnel[ i ].last=0;
            st->personnel[ i ].available=0;
            st->personnel[ i ].now=-1;
        }

        simulate(st);
        printf("Scenario %d: All requests are serviced within %d minutes.\n",scenario++,st->t);
    }

    exit(0);
}

```