

Novática, revista fundada en 1975 y decana de la prensa informática española, es el órgano oficial de expresión y formación continua de **ATI** (Asociación de Técnicos de Informática). **Novática** edita también **Upgrade**, revista digital de **CEPIS** (Council of European Professional Informatics Societies), en lengua inglesa, y es miembro fundador de **UPENET** (UPGRADE European Network)

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

ATI es miembro fundador 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**, **AIZ** y **ASTIC**.

CONSEJO EDITORIAL

Antoni Carbonell Noguera, Francisco López Crespo, Julián Marcelo Cocho, Celestino Martín Alonso, Josep Molas i Bertrán, Roberto Moya Quiles, César Pérez Chirinos, Mario Piattini Velthuis, Fernando Plera Gómez (Presidente del Consejo), Miquel Sarries Griño, Asunción Yturbe Herranz

Coordinación Editorial

Rafael Fernández Calvo <rfcalvo@ati.es>

Composición y autoedición

Jorge Llácer

Traducciones

Grupo de Lengua e Informática de ATI <<http://www.ati.es/gt/lengua-informatica/>>

Administración

Tomás Brunete, María José Fernández, Enric Camarero, Felicidad López

SECCIONES TÉCNICAS: COORDINADORES

Administración Pública electrónica

Gumersindo García Arribas, Francisco López Crespo (MAP)

<gumersindo.garcia@map.es> <lloca@ati.es>

Arquitecturas

Jordi Tubella (DAC-UPC) <jordit@ac.upc.es>

Victor Vilata Yibera (Univ. de Zaragoza) <vvilata@unizar.es>

Auditoría SITIC

Marina Touriño, Manuel Palao (ASIA)

<marinaburnino@marinatourino.com>, <manuel@palao.com>

Base de datos

Coral Calero Muñoz, Mario G. Piattini Velthuis

(Escuela Superior de Informática, UCLM)

<Coral.Calero@uclm.es>, <mpiattini@inf-cr.uclm.es>

Derecho e tecnologías

Isabel Hernando Collazos (Fac. Derecho de Donostia, UPV) <ihernando@legalek.net>

Isabel Davara Fernández de Marcos (Davara & Davara) <ldavara@davara.com>

Essenciaza Universitaria de la Informática

Joaquín Ezpeleta Mateo (CPS-UZAR) <ezpeleta@posta.unizar.es>

Cristóbal Pareja Flores (DSIP-UCM) <cpareja@sip.ucm.es>

Gestión del Conocimiento

Jean Baiget Solé (Cap Gemini Ernst & Young) <joan.baiget@ati.es>

Informática y Filosofía

Josep Corco (UIC) <jcorco@unica.edu>

Esperanza Maros (ESOCET-URJC) <cuaa@escet.urjc.es>

Informática Gráfica

Miguel Chover Selles (Universitat Jaume I de Castellón) <chover@lsi.uji.es>

Roberto Vivio (Eurographics, sección española) <rvivio@dsic.upv.es>

Ingeniería del Software

Javier Dolado Cosin (DLSI-UPV) <dolado@lsi.uji.es>

Luis Fernández (PRIS-UIEM) <lulem@pris.esi.uem.es>

Inteligencia Artificial

Federico Barber, Vicente Botti (DSIC-UPV)

<fvbotti.fbarber@dsic.upv.es>

Información Personal-Computador

Julio Abascal González (FI-UPV) <julio@si.uhu.es>

Jesús Lorés Vidal (Univ. de Lleida) <jesus@eup.udl.es>

Internet

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

Llorenç Pagès Casas (Indra) <pages@ati.es>

Lengua e Informática

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

Lenguajes Informáticos

Andrés Marín López (Univ. Carlos III) <amarin@ti.uc3m.es>

J. Angel Velázquez (ESOCET-URJC) <a.velazquez@escet.urjc.es>

Librerías e Informática

Alfonso Escolano (FIR-Univ. de La Laguna) <aescolano@ull.es>

Lingüística computacional

Xavier Gómez Guinovart (Univ. de Vigo) <xgg@wigo.es>

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

Mundo estudiantil

Adolfo Vázquez Rodríguez (Rama de Estudiantes del IEEE-UCM)

<a.vazquez@ieee.org>

Profesión Informática

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

Miquel Sarries Griño (Avto. de Barcelona) <msarries@ati.es>

Redes y servicios telemáticos

Luis Guisjarro Coloma (DCOM-UPV) <lguisjar@dcom.upv.es>

José Solé Pareta (DAC-UPC) <pareta@ac.upc.es>

Seguridad

Javier Areitio Bertolin (Univ. de Deusto) <jareitio@eside.deusto.es>

Javier López Muñoz (ETSI Informática-UMA) <jljm@icc.uma.es>

Sistemas de Tiempo Real

Alejandro Alonso, Juan Antonio de la Puente

(DIT-UPM) <[@dit.upm.es">aalonso.jpunte @dit.upm.es](mailto:aalonso.jpunte)>

Software Libre

Jesús M. González Barahona, Pedro de las Heras Quirós

(GSYC-URJC) <[@gsyc.escet.urjc.es">jph.gheras @gsyc.escet.urjc.es](mailto:jph.gheras)>

Tecnología de Objetos

Jesús García Molina (DIS-UM) <jmolina@correo.um.es>

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

Tecnologías para la Educación

Juan Manuel Doderio Berardo (UC3M) <jdoderio@inf.uc3m.es>

Francisco Riviere (Palomati) <friviere@waridoo.es>

Tecnologías y Empresa

Pablo Hernández Medrano (Bluematt) <pablohm@bluematt.biz>

TIC para la Salud

Valentín Masero Vargas (DI-UNEX) <vmasero@unex.es>

TIC y Turismo

Andrés Aguayo Maldonado, Antonio Guevara Plaza (Univ. de Málaga)

<aguayo.guevara@icc.uma.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, Redacción Central y Redacción ATI Madrid

Padilla 66, 3º dcha., 28006 Madrid

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

Composición, Edición y Redacción ATI Valencia

Av. del Reino de Valencia 23, 46005 Valencia

Tel. fax 963330282 <secretaria@ati.es>

Administración y Redacción ATI Cataluña

Via Laietana 41, 1º, 1º, 08003 Barcelona

Tel. 934125236; fax 934127713 <secretgen@ati.es>

Redacción ATI Andalucía

Isaac Newton, s/n, Ed. Sadiel

Isla Cartuja 41092 Sevilla, Tel./fax 954460779 <secretand@ati.es>

Redacción ATI Aragón

Lagascia 9, 3º B, 50006 Zaragoza

Tel. fax 976235181 <secretara@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 <secretgal@ati.es>

Suscripción y Ventas

<<http://www.ati.es/novatica/interres.html>>, o en ATI Cataluña o ATI Madrid

Publicidad

Padilla 66, 3º dcha., 28006 Madrid

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

Imprenta

9 Impresión S.A., Juan de Austria 66, 08005 Barcelona.

Deposito legal: B 15.154-1975 -- ISSN: 0211-2124; CODEN NOVACE

Periodico: Antonio Crespo Foix / © ATI 2004

Diseño: Fernando Agresta / © ATI 2004

editorial

Grupos de Trabajo y trabajo voluntario: a propósito de las IX JICS en resumen

> 02

Nos identifican (digitalmente) luego existimos

> 02

monografía

Firma electrónica e identidad digital

(En colaboración con *Upgrade*)

Editores invitados: *Javier López Muñoz, Apol·lònia Martínez Nadal, Ahmed Patel*

Presentación. La firma electrónica, clave para la seguridad en la Sociedad de la Información

> 03

Javier López Muñoz, Apol·lònia Martínez Nadal, Ahmed Patel

La firma digital como soporte de confianza de la Sociedad de la Información

> 05

Arturo Ribagorda Garnacho

La Declaración de Prácticas de Certificación de la FNMT-RCM

> 11

Josep Lluís Ferrer Gomila, Magdalena Payeras Capellà

Requisitos de funcionalidad y seguridad en firma electrónica

> 14

Gemma Déler Castro, Juan Carlos Cruellas Ibarz

La firma electrónica hoy: visión de un fabricante

> 18

Francisco Jordan Fernández, Jordi Buch i Tarrats

Desarrollo de un Sistema Integrado de Gestión Documental con servicio de firma electrónica avanzada

> 22

Iñaki Echevarria Larrinaga, Oscar García Jimeno- Juan Antonio Martín Zubiaur,

Víctor Llorente Gómez, Javier Areitio Bertolin

La legislación española sobre firma electrónica y DNI en el contexto europeo

> 27

Apol·lònia Martínez Nadal

La Ley Modelo de la CNUDMI/UNCITRAL sobre las Firmas Electrónicas

> 31

Rafael Illescas Ortiz

Iniciativas legales sobre firma electrónica en Latinoamérica

> 35

Mariñana Rico Carrillo

/ docs /

Tal como somos ("Encuesta ATI")

> 39

PAFET 2003

secciones técnicas

Enseñanza Universitaria de la Informática

Tendencias actuales en las herramientas de ayuda para la enseñanza y el aprendizaje de la programación

> 45

Mercedes Gómez Albarrán

Ingeniería del Software

Ingeniería concurrente y evaluación en el desarrollo del software: el caso del proyecto Top Fit

> 49

Andrés Muñoz Machado, Miguel Ángel Pérez Costero

Redes y servicios telemáticos

> 53

Servicio VoIP para Redes Móviles

Ai-Chun Pang, Yi-Bing Lin

Tecnología de Objetos

Naturaleza de las relaciones entre actores y casos de uso

> 56

Gonzalo Génova Fuster, Juan Llorens Morillo

Referencias autorizadas

> 62

sociedad de la información

Personal y transferible

El Braille y el placer de la lectura: los ciegos queremos seguir leyendo con los dedos

> 67

Carmen Bonet Borrás

Programar es crear

Por otra ruta, por favor (CUPCAM 2003, problema E, enunciado)

> 73

Ángel Herranz Nieve

Un refactorizador simple (CUPCAM 2003, problema D, solución)

> 74

José A. Leiva Izquierdo, Ángel Herranz Nieve

asuntos interiores

Coordinación editorial / Programación de Novática

> 76

Normas de publicación para autores / Socios Institucionales

> 77

Monografía del próximo número: "Agentes Software"

José A. Leiva Izquierdo¹, Ángel Herranz Nieva²
¹ Plettac Seguridad y Sistemas; ² Facultad de Informática, Universidad Politécnica de Madrid

<jleiva@ya.com>, <aherranz@fi.upm.es>

1. Resumen del problema

El problema consistía en programar una *refactorizador* de código que básicamente moviera operaciones de un módulo a otro con el objetivo de disminuir el acoplamiento entre todos los módulos que constituyen el sistema. Es obvio que el problema es una abstracción trivial del concepto de acoplamiento y que se ignoran infinidad de dificultades reales. El acoplamiento total de sistema se mide mediante la suma de apariciones estáticas de llamadas entre operaciones de distintos módulos. Como información de entrada al problema se recibían módulos, métodos, pertenecía de métodos a módulos y llamadas de métodos a métodos. La salida que se espera es el valor del acoplamiento mínimo tras aplicar la refactorización.

2. Grafos y colores

Es posible aplicar alguna variante de coloración de grafos. Cada método es un nodo (de un color determinado por pertenecer a un módulo concreto). Una arista (dirigida) existe entre dos nodos si hay una llamada de un método a otro. El objetivo final es minimizar el número de aristas que parten de un nodo y llegan a otro de diferente color.

Mediante un algoritmo iterativo sería posible estabilizar el grafo llevando a cada método al módulo óptimo en su caso. En cada paso del algoritmo se seleccionaría un método cuyo acoplamiento fuera positivo, existiendo un módulo tal que al pertenecer se disminuyera el acoplamiento general del programa.

3. Algoritmo

Se propone una sencilla implementación en Java para modelar el dominio descrito y aplicar el sencillo algoritmo iterativo propuesto. El primer paso es modelar el dominio donde deben coexistir métodos, módulos (también llamadas clases) y el programa como tal. Se han omitido del código los métodos de acceso a atributos y aquellos encargados de analizar la entrada.

Todos los objetos del dominio implementan un método encargado de llevar a cabo la tarea de optimización. El verdadero paso de iteración se da en la clase encargada de definir un método. Los objetos módulo y programa gestionan el trabajo realizado por la clase método y se detienen en el momento en que un paso no ha implicado mejoría.

Bien es cierto que esta solución puede encontrarse el problema de detener la ejecución creyendo haber llegado a una solución óptima cuando en realidad no es así. Sin embargo en pro de simplificar la solución el conjunto de datos de partida fue reducido voluntariamente asumiendo el grado de error.

```
class Method
{
    private int id;
    private Module module;
    private Vector called;

    public Method(int id, Module module)
    {
        called = new Vector();
        this.setModule(module);
        this.setId(id);
    }

    public boolean optimise()
    {
        boolean result = false;
        int highest =
            getMostFrecuentCalledModule();
```

Un refactorizador simple

El enunciado de este problema apareció en el número 168 de *Novática* (marzo-abril 2004, p. 71). Es el problema D de los planteados en el I Concurso Universitario de la Comunidad Autónoma de Madrid (CUPCAM 2003)

```
    if (highest != module.getId()) {
        module.getProgram().moveMethod(
            module.getId(),
            getId(),
            highest);
        result = true;
    }
    return result;
}

private int
getMostFrecuentCalledModule()
{
    int highest = this.module.getId();
    int[] sources =
        new int[module.getProgram().
            getModulesAmount()];

    for (int i = 0;
        i < called.size();
        i++)
        sources[
            ((Method) called.elementAt(i)).
                getModule().getId()
        ]++;

    for (int i = 0;
        i < sources.length;
        i++)
        if (sources[i]
            > sources[highest])
            highest = i;
    return highest;
}

public int getForeignCalls()
{
    int result = 0;

    for (int i = 0;
        i < called.size();
        i++)
        if (((Method) called.
            elementAt(i)).
                getModule().getId()
            != module.getId())
            result++;
    return result;
}
```

El objeto módulo, definido como clase en la demostración posee un conjunto de métodos sobre los que aplica el algoritmo de optimización.

```
class Module
{
    private int id;
    private Vector methods;
    private Program program;
    private int nextMethodId;

    public boolean optimise()
    {
        boolean improved = false;
        for (int i = 0;
            i < methods.size();
```

```

        i++)
        improved =
        improved ||
        ((Method)methods.
        elementAt(i)).
        optimise();
    return improved;
}

public void addMethod(Method method)
{
    method.setModule(this);
    method.setId(nextMethodId);
    methods.add(method);
    nextMethodId++;
}

public void delMethod(Method method)
{
    methods.remove(method);
}

public void reIndexMethods()
{
    for (int i = 0;
        i < methods.size();
        i++)
        ((Method)methods.elementAt(i)).
        setId(i);
}

public int getForeignCalls()
{
    int result = 0;

    for (int i = 0;
        i < methods.size();
        i++)
        result +=
        ((Method)methods.
        elementAt(i)).
        getForeignCalls();
    return result;
}

```

Un programa está compuesto por un conjunto de módulos. Se muestran el método que desencadena el proceso de optimización y el implicado en la aplicación del patrón implicado en el desplazamiento de código de un módulo a otro.

```

public class Program
{
    private Vector modules;

    public Program()
    {
        modules = new Vector();
    }

    public void optimise()
    {
        boolean improved = true;
        while (improved) {
            improved = false;
            for (int i = 0;
                i < modules.size();
                i++)
                improved =
                improved ||
                ((Module)modules.
                elementAt(i)).
                optimise();
        }
        reIndexMethods();
    }
}

```

```

    }

    private void reIndexMethods()
    {
        for (int i = 0;
            i < modules.size();
            i++)
            ((Module)modules.elementAt(i)).
            reIndexMethods();
    }

    public void addModule(Module module)
    {
        modules.add(module.getId(),
            module);
    }

    public void
        moveMethod(int fromModule,
            int idMethod,
            int toModule)
    {
        Method method =
            getModule(fromModule).
            getMethod(idMethod);
        getModule(fromModule).
        delMethod(method);
        getModule(toModule).
        addMethod(method);
    }

    public int getForeignCalls()
    {
        int result = 0;
        for (int i = 0;
            i < modules.size();
            i++)
            result +=
            ((Module)modules.
            elementAt(i)).
            getForeignCalls();
        return result;
    }
}

```

Por último la clase Refactorer es la encargada de tomar la entrada estándar, construir el dominio y ejecutar el algoritmo de optimización.

```

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class Refactorer {

    private Program program;

    public void ioRun()
    {
        try {
            BufferedReader br =
                new BufferedReader(
                    new InputStreamReader(
                        System.in));
            program = new Program();
            while (!program.parse(br)) {
                program.optimise();
                System.out.println(
                    program.getForeignCalls());
            }
        } catch (IOException ioe) {
            System.out.println(
                "IOException: "
                + ioe.getLocalizedMessage());
        }
    }
}

```

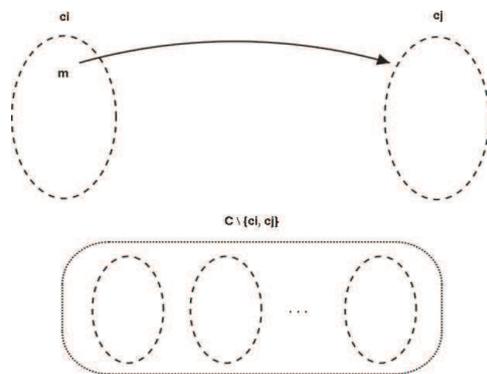


Figura 1. Prueba.

4. Terminación

Nos asalta una duda: ¿el algoritmo propuesto termina? Para ello nuestro objetivo será demostrar como el acoplamiento general del programa disminuye en cada iteración. Se muestran unas definiciones previas para indicar a continuación la condición que se debe cumplir para desplazar un método entre módulos. Dado un programa, utilizaremos la siguiente notación:

Conjunto de clases $C = c_0, c_1, c_2, \dots, c_i$

Conjunto de métodos $M = m_0, m_1, m_2, \dots, m_i$ tal que $\forall m \in M. \exists c \in C. m \in c$

Grado de salida de un método $g_s(m_i)$ Número de llamadas a procedimientos que realiza

Grado de entrada de un método $g_e(m_i)$ Número de métodos que le llaman

Grado de salida a una clase $g_s(m_i, c_j)$ Número de llamadas a procedimientos de la clase c_j

Grado de entrada de una clase $g_e(m_i, c_j)$ Número de procedimientos de la clase c_j que llaman

Acoplamiento de un método Diferencia entre las llamadas a métodos de otras clases con respecto a métodos de su propia clase. Sea el método m_i y la clase c_j con $m_i < c_j$ se define el acoplamiento de m_i como

Acoplamiento de programa

$$A_G = \sum_c^{m_i} A(m_i)$$

Cuando un método se desplaza de clase tal y como muestra la figura 1 el acoplamiento de los métodos de la clase c_i que llamaban a m aumenta mientras que el acoplamiento de m y el de los métodos de c_j que llaman a m disminuyen.

Dado que el objetivo es lograr que el acoplamiento disminuya en cada paso del algoritmo es necesario asegurarse de que el aumento que provocan los métodos de c_i sea inferior a la disminución lograda con m y los métodos de c_j . Las condiciones a tener en cuenta serían las siguientes:

$$g_e(m, c_i) < g_e(m, c_j) + g_s(m, c_j) \quad (1)$$

$$g_s(m, c_j) < g_s(m, c_i) \quad (2)$$

También se podrían sustituir las dos condiciones por una sola:

$$g_e(m, c_i) + g_s(m, m_i) < 2(g_e(m, c_j) + g_s(m, c_j)) \quad (3)$$

Novática disponible en la Intranet de ATI

Recordamos que en la Intranet de ATI, en <http://intranet.ati.es/novatica/>, se encuentran disponibles, en formato PDF, las versiones digitales completas de todos los números de nuestra revista a partir del 138 (feb.-mar. 1999), además de los números 0 y 1, incorporados recientemente.

El acceso está restringido a los socios de ATI únicamente, con la única condición de que estén registrados en ATInet. Para registrarse hay que ponerse en contacto con info@ati.es.

Noticias de UPGRADE y UPENET (UPGRADE European Network)

En el número de junio 2004 de **UPGRADE**, <http://www.upgrade-cepis.org>, se ha incorporado a **UPENET (UPGRADE European Network)** la revista digital **Mondo Digitale**, <http://www.monodigitale.net>, editada por AICA (Associazione Italiana per l'Informatica ed il Calcolo Automatico). En dicho número aparece un artículo de esta revista italiana, seleccionado por su director, **Franco Filippazzi**.

Tras esta nueva incorporación, **UPENET**, la recientemente creada red de publicaciones de sociedades miembro de CEPIS (Council of Professional Informatics Societies, <http://www.upgrade-cepis.org>), cuenta ya con tres miembros: **Novática**, **Mondo Digitale** y **Pro Dialog**, revista polaca coeditada por PTI-PIPS (Polskie Towarzystwo Informatyczne – Polish Information Processing Society) y por el Instituto de Informática de la Universidad Técnica de Poznan.

Tras el verano está prevista la incorporación de revistas de otras sociedades miembro de CEPIS, de lo cual podrán beneficiarse los lectores de **Novática**, pues las revistas que forman parte de esta red, entre ellas la nuestra, podrán a su vez publicar artículos aparecidos en **UPGRADE** y en las revistas participantes en la misma ... y también se beneficiarán los autores pues en **UPGRADE** podrán publicarse artículos seleccionados de las revistas que forman parte de **UPENET**.

Recordamos que **UPGRADE** tiene una lista cuyo objetivo es distribuir noticias sobre esta revista digital. En <http://www.upgrade-cepis.org/pages/editinfo.html#newslst> se encuentra información sobre sus características y sobre cómo darse de alta en ella.

programación de novática

Monografías para 2004

Por acuerdo de los Consejos Editoriales de **Novática** y **Upgrade** los temas de las restantes monografías del año 2004, salvo imprevistos o causas de fuerza mayor, serán los siguientes:

Nº 170 (julio-agosto): “Agentes Software”. Editores invitados: **Pedro Cuesta Morales**, pcuesta@uvigo.es y **Juan Carlos González**, jcmoreno@ei.uvigo.es (Universidad de Vigo), **Zahia Guessoum** (Université Paris VI, Francia), Zahia.Guessoum@lip6.fr y **Juan Pavón Mestras** (Universidad Complutense de Madrid), jpavon@sip.ucm.es.

Nº 171 (septiembre-octubre): “Tecnologías de Proceso Software”. Editores invitados: **Gerardo Cánfora** (Università di Sannio, Italia), gerardo.canfora@unisannio.it, y **Francisco Ruiz González** (Universidad de Castilla - La Mancha), francisco.ruizg@uclm.es.

Nº 172 (noviembre-diciembre): “Criptografía”. Editores invitados: **Javier Areitio Bertolín** (Universidad de Deusto), jareitio@orion.deusto.es, **Arturo Ribagorda Garnacho** (Universidad Carlos III de Madrid), arturo@inf.uc3m.es.