

# Sistema ANACONDA para el análisis automático de la calidad del software

Enrique Martín<sup>a</sup>, Fernando Sáenz<sup>b</sup>

<sup>a</sup>Telefónica Móviles España, <sup>b</sup>Universidad Complutense de Madrid

## Resumen

ANACONDA (analizador automático de código Natural para el desarrollo de aplicaciones) es un sistema para la evaluación de la calidad del software de aplicaciones informáticas desarrolladas en el entorno Natural/ADABAS. Analiza la aplicación partiendo del código fuente, verificando que cumple los requisitos de calidad establecidos en la instalación para poder ser implantada y explotada, y proporciona un diagnóstico que refleja el nivel de calidad del software. Este diagnóstico lo realiza basándose en unas métricas y pesos definidos a partir del conocimiento de los expertos. Es capaz de analizar miles de líneas de código fuente por minuto generando un informe detallado con los errores y recomendaciones.

## 1. Introducción

La fuerte evolución tecnológica actual provoca que la sociedad demande de la informática la capacidad de desarrollar un gran número de aplicaciones informáticas cada vez más grandes y en unos plazos de tiempo cada vez más cortos. Esto tiene como consecuencia que el software desarrollado no siempre tenga la calidad necesaria. Esta falta de calidad se manifiesta en posteriores fallos y defectos que generan pérdidas importantes y altos costes de mantenimiento.

Una cuestión que se plantea es si esta demanda debe implicar necesariamente una reducción en la calidad. Sin embargo, cada vez se tiene más presente la necesidad de garantizar un determinado nivel de calidad del software. Los clientes exigen el aseguramiento de la calidad mediante la certificación<sup>1</sup> [1]. El principal obstáculo es la necesidad de dedicar una cantidad de recursos humanos y un tiempo de los que normalmente no se dispone; por ello se plantea la necesidad de desarrollar herramientas automáticas que ayuden en esta tarea. Teniendo en cuenta que garantizar la calidad total del software es actualmente un objetivo prácticamente inalcanzable, un objetivo viable es construir herramientas que ayuden a garantizar una calidad aceptable a un costo razonable.

---

<sup>1</sup> Organizaciones como AENOR son las encargadas de emitir las certificaciones de los productos bajo determinados estándares (e.g., la serie ISO 9000).

Considerando que existen una serie de atributos del software que se pueden medir de forma automática a partir del código fuente, como se presentará en apartados posteriores, se ha desarrollado el sistema ANACONDA que analiza automáticamente el código fuente y emite un diagnóstico fiable sobre la calidad del software. El trabajo desarrollado se centra por tanto en la medición de la calidad del software<sup>2</sup>. Se definen métricas de producto [2] exclusivamente en la fase de codificación; es decir, la etapa en el plan de calidad posterior a la especificación y diseño, y anterior a la verificación. Nos centramos en el *análisis sintáctico* del producto en lugar del *análisis semántico* de los programas y aplicaciones (equivalencia semántica entre la especificación y la implementación). No obstante, realizamos medidas para obtener datos de complejidad ciclomática simple (cfr. [3]). Se estudian indicadores de calidad de la fase de codificación tales como la autodocumentación de programas, el mantenimiento del código, los errores, la trazabilidad o el cumplimiento de la normativa, que se pueden encontrar en propuestas como la de McCall [4]. El sistema de medida propuesto se puede integrar en el estándar de aseguramiento de la calidad que la empresa contrate con el cliente o tenga como propio. Como en los modelos de evaluación de calidad de McCall [4] y de Boehm [5] (cfr. [6]), descomponemos los indicadores de calidad (generalmente externos) en características más sencillas hasta que puedan medirse objetiva y directamente (usando las medidas como signos de los indicadores de calidad).

En [7] se puede encontrar un trabajo relacionado en el que se realizan medidas de volumen y complejidad de código, aunque su orientación es claramente diferente. En nuestro trabajo se usan múltiples medidas para obtener indicadores de calidad como los mencionados en el párrafo anterior, y además aportamos datos del rendimiento del análisis, cuya importancia es clara cuando el tiempo es un factor importante.

El resto del artículo se organiza como sigue. En el apartado 2 se introducen las características de ANACONDA. En el apartado 3 se introduce la terminología y el procedimiento de cálculo de la calidad, mostrando un caso de estudio. En el apartado 4 se resumen la arquitectura y el funcionamiento de ANACONDA. Finalmente, en el apartado 5 se extraen algunas conclusiones del trabajo.

## **2. Características de ANACONDA**

ANACONDA se ha desarrollado para analizar código fuente del lenguaje de cuarta generación Natural. Su implementación se ha realizado en este mismo lenguaje para quedar integrado dentro del entorno Natural/ADABAS como un componente más y así poder interac-

---

<sup>2</sup> Se puede encontrar una revisión general de la medición de la calidad del software en [6].

tuar con el resto de componentes (editores, compiladores, bibliotecas, diccionario de datos, etc.)

Se ha diseñado con una filosofía incremental con una doble finalidad. Por un lado, para obtener resultados desde el primer ciclo de desarrollo y, por otro, no poner límites al sistema, permitiendo que pueda crecer en la dirección que marquen las necesidades de cada momento.

Con este sistema se permite analizar el código fuente desde un solo módulo Natural (programa, subprograma, mapa, área de datos, etc.) hasta una o varias aplicaciones en conjunto.

Para realizar el análisis no es necesario trasladar módulos o parar las actividades normales como la edición, el almacenamiento, la compilación y la ejecución. El análisis se hace de forma transparente para los usuarios, ya sean técnicos de desarrollo o usuarios finales. Por tanto, en cualquier momento se pueden realizar análisis para conocer el estado de la calidad del software en aplicaciones, independientemente de que éstas estén en fase de desarrollo, preexplotación o explotación.

El análisis se puede realizar en modo interactivo o en modo de procesamiento por lotes. En ejecución interactiva se puede seguir en tiempo real la evolución del análisis con la información que se presenta por pantalla y, al término del proceso, se presenta un resumen con la información más relevante. En ejecución por lotes se genera un extenso informe en el que además del diagnóstico y de un resumen se recogen una a una las incidencias detectadas.

Tiene una interfaz de usuario amigable que permite de forma sencilla introducir la información necesaria para la ejecución del análisis (módulos a analizar, ejecución interactiva o por lotes, el tipo de módulos a analizar, etc.)

El departamento de Normalización y Calidad de la instalación es el encargado de la supervisión de ANACONDA, y su aplicación debe ser un paso obligado para las aplicaciones una vez terminado su desarrollo y toda vez que el código se modifique durante el mantenimiento.

### **3. Atributos, medidas, métricas, indicadores, pesos y diagnóstico**

En este apartado se introduce la terminología usada en el resto del artículo y el procedimiento general seguido para la evaluación de la calidad<sup>3</sup>, como se esquematiza en la figura 1.

---

<sup>3</sup> Siguiendo algunos enfoques resumidos en [8].

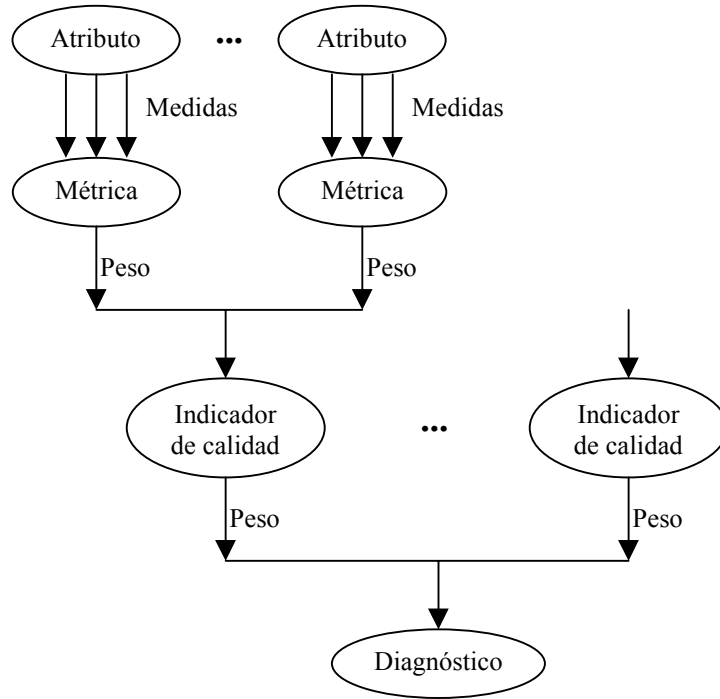
Se denominan atributos a los factores que pueden influir en la calidad del software. Al no ser posible medir de forma automática todos los atributos, debido al alto grado de subjetividad de algunos de ellos, en ANACONDA se consideran sólo aquéllos que sí pueden de alguna manera ser cuantificados automáticamente para poder medirlos. La medida de un atributo es una cantidad que en ocasiones indica el número de veces que algo ocurre y en otras simplemente indica si algo ocurre o no. Cada atributo puede tener varias medidas asociadas.

Para cada atributo se ha establecido una métrica. A partir de un atributo, la métrica es una función que, aplicada sobre sus medidas, determina una cuantificación de la calidad del atributo. El rango de las métricas se ha normalizado de cero a cien de forma que a cualquier medida le corresponderá siempre una métrica dentro de este rango. El valor cero representa el valor más bajo en el nivel de calidad y el valor cien el más alto. En cualquier métrica se consideran al menos estos dos valores, y se pueden considerar tantos otros valores intermedios como sea necesario. El conjunto de métricas normalizado permite realizar una evaluación de la calidad.

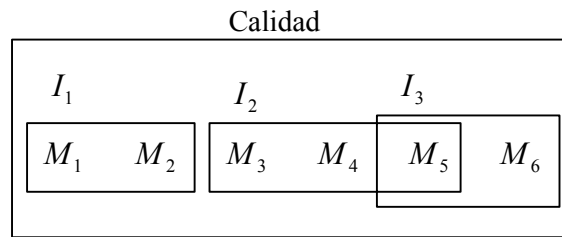
De la evaluación de una o más métricas se obtiene un indicador de calidad que se calcula en función de las métricas que intervienen en el indicador, asignándoles un peso a cada una de ellas. Una métrica puede intervenir en uno o más indicadores.

Por último, para realizar el diagnóstico del nivel de calidad del software alcanzado se ha definido el peso de cada indicador en relación al resto de indicadores. La figura 2 resume el procedimiento de cálculo de la calidad.

Para definir los atributos, su cuantificación, las métricas, los indicadores, el peso de cada métrica en los indicadores en que interviene y el peso de cada indicador en el diagnóstico se ha recurrido al conocimiento de los expertos humanos en la materia. Todas las métricas y pesos son parámetros de ANACONDA y pueden cambiarse para ajustar su funcionamiento.



**Figura 1.** Representación de los elementos que intervienen en el análisis de calidad.



$$C = \sum_i P_i \cdot I_i$$

$$I_i = \sum_j P_{ij} \cdot M_j$$

$$M_j = f_j(m_j^1, \dots, m_j^{k_j})$$

$C$  : Cuantificador de la calidad.

$I_i$  : Indicador de calidad  $i$ -ésimo.

$P_i$  : Peso del indicador de calidad  $I_i$  en la evaluación de la calidad  $C$ .

$M_j$  : Métrica del atributo  $j$ .

$P_{ij}$  : Peso de la métrica  $M_j$  en la evaluación del indicador de calidad  $I_i$ .

$f_j$  : Función de cuantificación de calidad del atributo  $j$ .

$m_j^i$  : Medidas del atributo  $j$  ( $i \in \{1, \dots, k_j\}$ ).

**Figura 2.** Cálculo de la calidad.

### 3.1. Caso de estudio

El objetivo de este apartado es ilustrar con un ejemplo sencillo los elementos descritos hasta el momento. Para ello se ha escogido el indicador de calidad de la documentación en el código (autodocumentación), y se describe cómo se determinan sus atributos, medidas, métricas y pesos.

#### 3.1.1. *Determinación de los atributos*

En este primer paso se determinan los atributos relacionados con la autodocumentación que se pueden cuantificar. Se consideran los siguientes:

- *Módulos sin cabecera (MSC)*. Determina si un módulo tiene una cabecera que lo identifica y describe.
- *Número de comentarios frente al número de líneas de código (NCNL)*. Determina si un módulo está suficientemente autodocumentado para facilitar su comprensión.
- *Módulos no documentados en el diccionario Predict (MND)*. Determina si un módulo tiene su diseño técnico documentado en el diccionario Predict.
- *Módulos con la cabecera incompleta (MCI)*. Determina si la cabecera está documentada completamente según la normativa. La estructura de la cabecera está recogida en la normativa de la instalación y determina los elementos que debe contener y en qué orden.
- *Módulos que llaman a módulos no documentados en la cabecera (MLMND)*. Se trata de asegurar que todos los módulos que se llaman desde el que se esté analizando están documentados en la cabecera del módulo.
- *Módulos documentados en la cabecera y que no son llamados (MDNL)*. Se trata de asegurar que los módulos documentados en la cabecera como llamados lo son realmente.

#### 3.1.2. *Determinación de las medidas*

En este segundo apartado se determina la manera de medir los atributos. Se definen las siguientes medidas para cada atributo:

- *Módulos sin cabecera (MSC)*. Este atributo sólo se considera en los módulos tipo programa, subprograma, subrutina y copycode. Se comprueba si en el módulo existe una cabecera (EC); si es así, la medida es 1, y 0 en caso contrario.
- *Número de comentarios frente al número de líneas de código (NCNL)*. Se aplica a todo tipo de módulos. Las medidas son el número de líneas de código (L) y el número de líneas que son o que contienen comentarios (C).
- *Módulos no documentados en el diccionario Predict (MND)*. Se aplica a todos los tipos de módulos. Se accede al diccionario y se comprueba si el módulo está documentado en Predict (DP); si es así, la medida es 1, y 0 en caso contrario.

- *Módulos con la cabecera incompleta (MCI)*. Se aplica a programas, subprogramas, subrutinas y copycodes. Se determinan los conceptos que aparecen documentados en la cabecera buscando las palabras que los identifican (autor, descripción, fecha, etc.) y se comprueba que están acompañados de contenido. Las medidas son el número de conceptos buscados (Co) y el número de conceptos documentados (CoD).
- *Módulos que llaman a módulos no documentados en la cabecera (MLMND)*. Se aplica a programas, subprogramas, subrutinas y copycodes. Las medidas son el número de módulos documentados en la cabecera que son llamados (MDLL) y el número de módulos realmente llamados (MLL).
- *Módulos documentados en la cabecera y que no son llamados (MDNL)*. Se aplica a programas, subprogramas, subrutinas y copycodes. Las medidas son el número de módulos documentados en la cabecera que no son llamados (MDnLL) y el número de módulos documentados en la cabecera (MD).

### 3.1.3. Determinación de las métricas

En este tercer paso se determinan las métricas y se normalizan en el rango de 0 a 100. Se establece el siguiente criterio:

- *Módulos sin cabecera ( $M_{MSC}$ )*. Se toma como métrica 100 si la medida fue 1 y 0 si la medida fue 0.  $M_{MSC}=100*EC$ .
- *Número de comentarios frente al número de líneas de código ( $M_{NCNL}$ )*. Para determinar la métrica se atiende al tipo de módulo en cuestión; si es un programa, subprograma, subrutina o copycode, la métrica se determina por proporcionalidad directa entre la relación de líneas de comentarios frente a las líneas de código. La relación que se considera óptima es un comentario por cada cinco líneas de código, sin penalizar el exceso; esto es,  $M_{NCNL}=\min(5*C/L*100, 100)$ . Si es un área de datos, el cálculo es similar pero considerando que la relación óptima es un comentario por cada línea de código que contenga la definición de un dato. Por último, si es un mapa, el cálculo también es similar pero considerando sólo las líneas de código de las reglas de proceso definidas en él.
- *Módulos no documentados en el diccionario Predict ( $M_{MND}$ )*. Se aplica a todos los tipos de módulos. Se toma como métrica 100 si la medida fue 1 y 0 si la medida fue 0.  $M_{MND}=100*DP$ .
- *Módulos con la cabecera incompleta ( $M_{MCI}$ )*. La métrica es el porcentaje de conceptos documentados frente al número de conceptos buscados.  $M_{MCI}=CoD/Co*100$ .
- *Módulos que llaman a módulos no documentados en la cabecera ( $M_{MLMND}$ )*. La métrica es el porcentaje del número de módulos llamados y documentados en la cabecera frente al

número total de módulos llamados, teniendo en cuenta que si el número de módulos llama-

$$\text{mados es 0 la métrica es 100. } M_{MLMND} = \begin{cases} 100, \text{ si } MLL = 0 \\ \frac{MDLL}{MLL} \cdot 100, \text{ e.o.c} \end{cases}$$

- *Módulos documentados en la cabecera y que no son llamados ( $M_{MDNL}$ )*. La métrica es el porcentaje del número de módulos documentados no llamados y el número de módulos documentados en la cabecera, salvo si el número de módulos documentados en la cabecera es 0 o si el número de módulos documentados y no llamados es 0, en cuyo caso la mé-

$$\text{trica es 100. } M_{MDNL} = \begin{cases} 100, \text{ si } MD = 0 \vee MDnLL = 0 \\ \frac{MDnLL}{MD} \cdot 100, \text{ e.o.c} \end{cases}$$

#### 3.1.4. Determinación del peso de cada métrica en el indicador de calidad

En este cuarto paso se determina el peso que debe aplicarse a cada métrica para calcular el indicador de calidad. Estos pesos, que se muestran en la tabla 1, se determinan en función de la importancia que tiene cada atributo en el indicador de calidad.

Indicador de calidad de la autodocumentación	
Métricas	Pesos
Módulos sin cabecera ( $M_{MSC}$ )	25%
Número de comentarios frente al número de líneas de código ( $M_{NCNL}$ )	20%
Módulos no documentados en el diccionario Predict ( $M_{MND}$ )	25%
Módulos con la cabecera incompleta ( $M_{MCI}$ )	10%
Módulos que llaman a módulos no documentados en la cabecera ( $M_{MLMND}$ )	10%
Módulos documentados en la cabecera y que no son llamados ( $M_{MDNL}$ )	10%

**Tabla 1.** Pesos de las métricas para el indicador de la calidad de la autodocumentación.

#### 3.1.5. Determinación del peso del indicador de calidad en el diagnóstico

En este último paso se determina el peso que debe aplicarse al indicador de la calidad de la autodocumentación en función de la importancia de este indicador frente al resto de indicadores en el diagnóstico del nivel de calidad del software. A este peso se le ha asignado el valor 10%.



#### **4. Arquitectura y funcionamiento de ANACONDA**

ANACONDA está formado por distintos componentes. El componente principal supervisa el análisis, localiza los módulos Natural en las distintas bibliotecas e identifica su tipo. Existen unos componentes secundarios especializados en el análisis de cada uno de los tipos de módulos Natural (programas, subprogramas, mapas, etc.). Por último existen unos componentes auxiliares que son llamados desde el componentes principal y desde los componentes secundarios para realizar tareas concretas tales como la determinación de la fecha de almacenamiento, la localización de comentarios en una línea, etc.

El componente principal localiza uno a uno los módulos a analizar. Para cada módulo, identifica su tipo y pasa el control al componente secundario especializado en el análisis de ese tipo de módulo. El componente secundario, con la intervención de los componentes auxiliares, mide cada uno de los atributos correspondientes al tipo de módulo que se está analizando y, al devolver el control al componente principal, calcula su métrica, acumulando el resultado al de los módulos ya analizados. Después realiza una evaluación de estas métricas recién actualizadas, aplicándoles los pesos correspondientes para deducir los indicadores de calidad. Por último, se aplica a estos indicadores su peso correspondiente y se determina el diagnóstico. El proceso se repite hasta que no quedan módulos por analizar.

Se han establecido dos niveles de análisis: el de aceptación y el de detalle. El de aceptación se dedica a la localización de errores o de circunstancias con probabilidad de generar fallos o defectos en las fases de implantación y explotación. El de detalle se centra en circunstancias que, sin ser errores en sí, pueden provocar problemas posteriores al complicar la fase de mantenimiento, ya sea éste correctivo o evolutivo. El nivel de análisis de aceptación debe ser superado para que la aplicación sea calificada como apta para su implantación y explotación.

De forma orientativa, en un host IBM-9221 en condiciones normales de carga, la velocidad de análisis es aproximadamente 25.000 líneas de código por minuto. Hay que considerar que los dos niveles de análisis, el de aceptación y el de detalle, también pueden ejecutarse por separado. El análisis de aceptación procesa 76.000 líneas de código por minuto y el de análisis detallado, 31.000.

#### **5. Conclusiones**

ANACONDA es una herramienta que mejora la implantación, disminuye considerablemente el número de errores de explotación y facilita el mantenimiento de aplicaciones. Como valor añadido, es un marco de referencia imparcial y objetivo (bajo la subjetividad de

la elección de los parámetros de decisión por parte de los expertos) para comparar o determinar el nivel de calidad del software de las aplicaciones.

En las instalaciones donde de forma puntual o permanente no se dispone de personal con una cualificación alta, la herramienta proporciona información suficiente para que los responsables de la toma de decisiones puedan estar asistidos. En instalaciones que sí cuentan con personal especializado, esta herramienta disminuye notablemente su trabajo, descargándoles de las tareas más rutinarias y permitiendo así que se dediquen a tareas de mayor complejidad, al tiempo que, como ya se ha mencionado anteriormente, se dota a la instalación de un marco de referencia independiente y objetivo para tratar todas las aplicaciones por igual, compensando en cierto modo la inevitable subjetividad del técnico humano.

## Referencias

- [1] Alarcón, M.I., “Certificación: ¿objetivo o consecuencia?”, *Novática*, nº 137, 1999.
- [2] Freixanet, J.R., Cañas, E. y Roca, E., “Plan de métricas del software: aproximación a su diseño”, *Novática*, nº 128, 1997.
- [3] McCabe, T.J., “A Complexity Measure”, *IEEE Transactions on Software Engineering*, Vol SE-2, nº 4, Diciembre, 1976, pp. 308-320.
- [4] McCall, J., Richards, P., y Walters, G., *Factors in Software Quality*, NTIS AD-A049-014, 015, 055, noviembre 1977.
- [5] Boehm, B. et al, *Characteristics of Software Quality*, North Holland, Nueva York, 1978.
- [6] Fernández, L., “Una revisión breve de la medición de software”, *Novática*, nº 137, 1999.
- [7] Tuya, J., Álvarez, M.A., Venero, A., Matilla, R. y Corrales, J.A., “Seguimiento y control del desarrollo de proyectos software en las PYMES”, *Novática*, nº 125, 1997.
- [8] Pressman, R., *Ingeniería del software. Un enfoque práctico*, McGraw-Hill, Madrid, 1997.