Estructuras Complejas de Reutilización: Definición de Mecano Estático

Francisco José García Peñalvo*
fgarcia@ubu.es

José Manuel Marqués Corral†
jmmc@infor.uva.es

Miguel Ángel Laguna†
mlaguna@infor.uva.es

Jesús Manuel Maudes Raedo*
jmaudes@ubu.es

Resumen

El ámbito de la reutilización del software se ha extendido hacia las fases iniciales del ciclo vital de los desarrollos software, apareciendo nuevos criterios de clasificación para los nuevos productos software reutilizables. Pero estas clasificaciones no contemplan una reutilización que afecte a diferentes niveles de forma simultánea, característica que favorecería el concepto de reutilización sistemática desde la especificación de requisitos a la implementación. El presente trabajo presenta una estructura de reutilización compleja que soporta un elemento software reutilizable definido en diferentes niveles, que permita tanto al desarrollador para reutilización como al desarrollador con reutilización tener una visión más cercana a la reutilización de subsistemas a la hora de ponerla en práctica

Palabras Clave

Reutilización de software, estructuras complejas de reutilización, assets

*Área de Lenguajes y Sistemas Informáticos. Escuela Universitaria Politécnica. Universidad de Burgos

†Departamento de Informática. Universidad de Valladolid

1 Introducción

La reutilización del software es una técnica que se está practicando desde hace varias décadas, la primera propuesta al respecto data del año 1968 [McIlroy, 1976]. Sin embargo, es un tema que últimamente está suscitando un creciente interés, siendo numerosos los proyectos de investigación y desarrollo que se dedican a la reutilización.

Inicialmente, la reutilización del software se concibió como la combinación de componentes de código almacenados en bibliotecas de funciones mediante herramientas automatizadas, caracterizándose por llevarse a cabo con una total carencia de tintes metodológicos. Trabajos posteriores dieron lugar a la aparición de diferentes tecnologías, enfoques y extensiones del concepto de producto software reutilizable, existiendo diferentes clasificaciones de los mismos, [Jones, 1984], [Biggerstaff, 1992], [Krueger, 1992], [Mili et al., 1995], [Edwards et al., 1997]. En una primera aproximación, y de forma muy esquemática, todos estos desarrollos relacionados con la reutilización del software se pueden clasificar basándose en el objeto y en el método de reutilización usado.

Atendiendo al método de reutilización, se puede distinguir entre tecnologías de composición y tecnologías de generación. Las primeras presentan un enfoque de bloques de construcción, mientras que las segundas abordan la reutilización desde un enfoque generador o de procesador reutilizable.

Si se centra la atención en el punto de vista del objeto reutilizable, se debe destacar la ampliación del concepto de elemento reutilizable y el incremento del nivel de abstracción en el que se produce la reutilización.

Según esto, en la actualidad se considera que todo el conocimiento y productos derivados de la producción del software, son susceptibles de ser reutilizados en la construcción de nuevos sistemas software [Freeman, 1987].

El aumento del nivel de abstracción al que debe producirse la reutilización está justificada, en primer lugar, por el hecho de que el aumento de la potencia de reutilización pasa por la elevación del nivel de abstracción [Parnas et al., 1989], y, en segundo lugar, porque los elementos software reutilizables de mayor nivel de abstracción son los potencialmente más beneficiosos para la reutilización del software. Esto último es debido a que las primeras fases de los desarrollos software constituyen el mayor esfuerzo de todo el desarrollo, por tanto, los elementos software generados en estas fases deben suponer un importante aumento de la productividad y un ahorro significativo.

No obstante, aunque el aumento del ámbito de la reutilización es positivo, la totalidad de las taxonomías anteriormente mencionadas presentan unos elementos software reutilizables definidos en un solo nivel de abstracción o correspondiéndose a una sola fase del ciclo de vida de desarrollo del software. Como consecuencia de esto, ni los desarrolladores para reutilización, ni los desarrolladores con reutilización tienen concepción de la reutilización sistemática que puede afectar a varios niveles de abstracción al mismo tiempo.

Una primera conclusión que se puede extraer de todo lo anterior, es que la reutilización tiene que enfocarse hacia la reutilización de requisitos y diseños de alto nivel. Además, los beneficios de la reutilización de estos elementos software se verían aumentados si se reutilizasen los productos software subsiguientes, derivados de éstos [Cybulski et al., 1997]. Contar con un elemento software reutilizable complejo, definido

en diferentes niveles de abstracción simultáneamente, es un factor que potenciará la reutilización del software y en consecuencia los beneficios derivados de la reutilización.

En consecuencia, se propone como elemento base para la reutilización un elemento compuesto cuya estructura reúna assets catalogados en distintos niveles. Este componente software, de aquí en adelante recibirá el nombre "Mecano".

En la siguiente sección se van a presentar los conceptos sobre los cuales se van a construir los mecanos. En la sección tercera se incidirá en los requisitos que deben cumplir los mecanos, presentándose en la cuarta sección el modelo de mecano representado mediante un modelo objeto. Finalmente, se concluye este trabajo con unas conclusiones y las líneas de trabajo futuro a realizar.

2 Marco conceptual de los mecanos

El aumento del ámbito de acción de la reutilización hacia niveles de mayor abstracción lleva asociado la aparición de un nuevo concepto para reflejar el alcance actual de la reutilización, el asset. Un asset se define como *cualquier producto del ciclo de vida del software que pueda ser potencialmente reutilizado. Esto incluye: modelo de dominio, arquitectura de dominio, requisitos, diseño, código, bases de datos, esquemas de bases de datos, documentación, manuales de usuario, casos de prueba ...* [DoD, 1992].

Según lo expresado anteriormente, la reutilización se puede definir como "*cualquier procedimiento que produce o ayuda a producir un sistema mediante el nuevo uso de algún elemento precedente de un esfuerzo de desarrollo anterior*"[Freeman, 1987].

Dentro de este contexto, el objetivo de este trabajo es la definición de una estructura compleja de reutilización que defina un elemento software reutilizable, de forma que cumpla los siguientes principios:

- ***Aumento del nivel de abstracción de la reutilización:*** Debe aumentarse el alcance de la reutilización del software, dirigiéndolo hacia niveles de mayor abstracción que los conseguidos tradicionalemente mediante la reutilización de elementos software del nivel de implementación.
- ***Estructura multinivel de abstracción:*** La estructura debe acoger varios elementos software reutilizables relacionados entre sí y definidos en diferentes niveles de abstracción.
- ***Definición de un soporte para el desarrollo para reutilización:*** Esta estructura debe dar soporte a los elementos reutilizables que se hayan generado en un esfuerzo de desarrollo.
- ***Base para el desarrollo con reutilización:*** La estructura debe ofrecer una serie de operaciones que ofrezcan la flexibilidad necesaria para su reutilización en futuros desarrollos.

La estructura definida bajo las perspectivas anteriormente expuestas va a recibir el nombre de mecano. Una primera propuesta de definición de mecano se encuentra en [García et al., 1997], aunque de una forma más rigurosa se puede definir un mecano como: "*un conjunto de elementos software reutilizables, clasificados en diferentes niveles*

y relacionados entre sí, ya sea en el mismo nivel (relaciones intranivel) o en niveles diferentes (relaciones internivel), cumpliéndose la restricción de que debe existir al menos una r

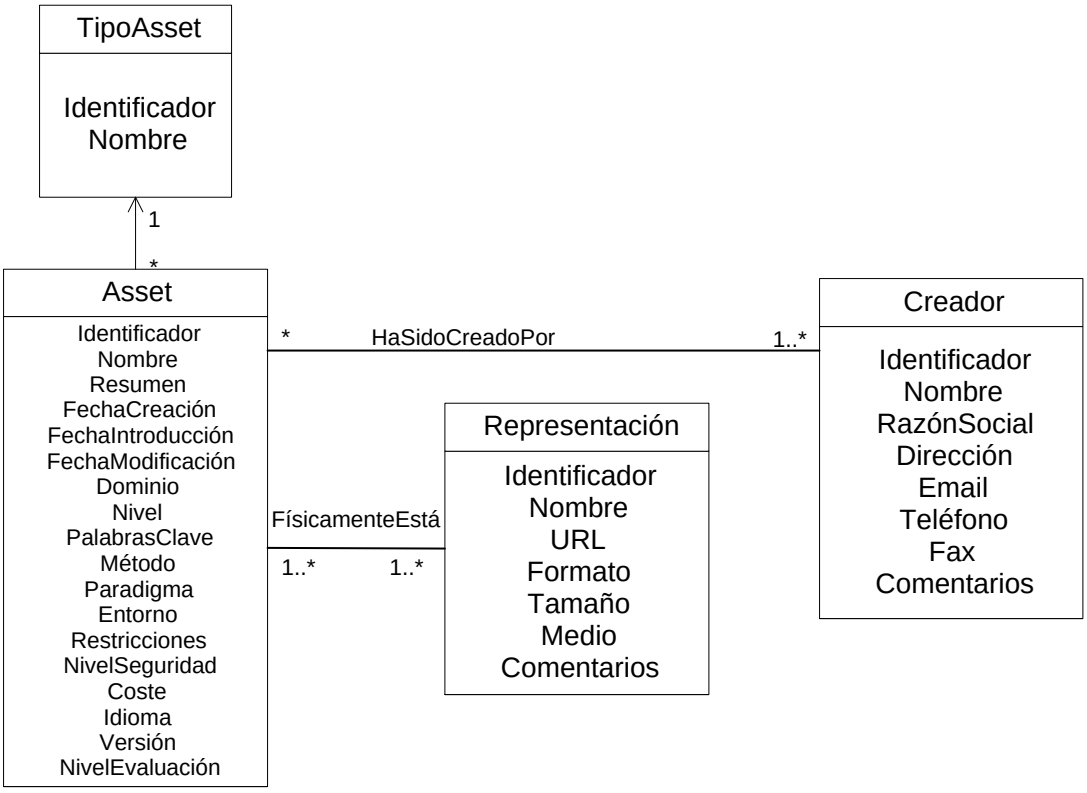

Figura 1: Estructura del asset.

- ***Las relaciones del dominio que sean unidireccionales deben reflejar el sentido de la relación.***
- ***Las relaciones del dominio entre los assets se derivan del dominio del desarrollo del software.***
- ***Las relaciones deben contar con las restricciones necesarias para evitar la introducción de incongruencias a la hora de relacionar los assets que componen un mecano.***
- ***Un mecano estático puede estar formado por un agregado de otros mecanos estáticos existentes.***
- ***Un mecano estático está ligado a un dominio y a un contexto de reutilización.***
- ***Un mecano estático debe soportar la evolución de sus assets componentes.***

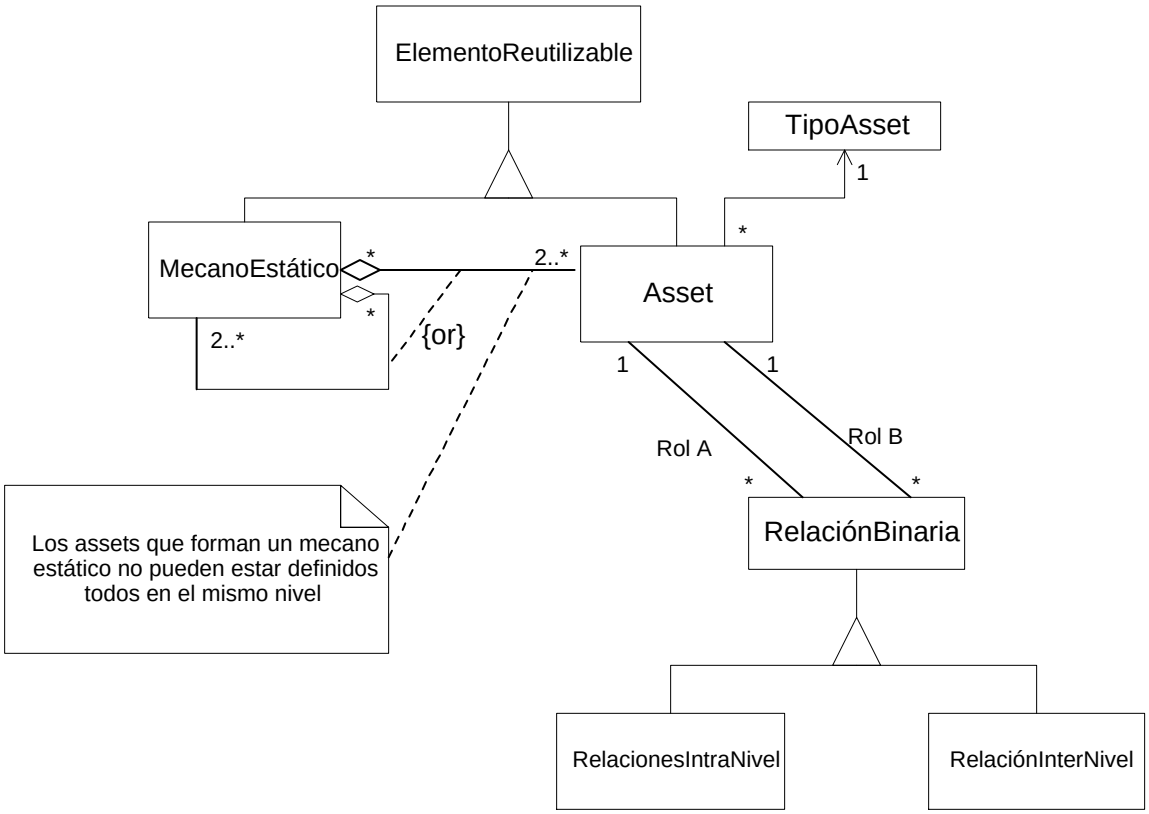

Figura 2: Estructura del mecano estático.

4 Modelo de mecano

Utilizando la notación UML 1.1, se va a presentar el modelo objeto que representa la estructura del asset (ver **Figura 1**) y la estructura del mecano estático (ver **Figura 2**).

5 Conclusiones y trabajo futuro

Con la definición de mecano, la distinción entre mecano estático y dinámico, y la creación de los modelos que representan la base de la estructura del asset componente y del mecano, se han sentado las bases para continuar el trabajo en pro de la definición de un entorno de reutilización sistemática que complemente los entornos de generación automática.

El trabajo inmediato derivado de esta línea de trabajo pasa por completar el modelo de mecano estático con las relaciones semánticas entre assets, además de modelar los diferentes escenarios formados por los posibles tipos de assets que presentan los distintos métodos de desarrollo de software.

Para la generación de los modelos de asset y de mecano se ha realizado utilizando UML 1.1, pero en su versión definitiva debería ajustarse al metamodelo que ofrece OASIS.

Otro aspecto relevante en el que ha de incidirse es en el tema del versionado de assets y su repercusión en los mecanos que los contienen.

Todos los avances que se van haciendo en el campo teórico dentro del grupo de investigación, deben tener su correspondiente estudio en el terreno práctico. Las primeras pruebas realizadas en este sentido se han llevado a cabo utilizando el repositorio EURO