

Evaluación de herramientas CASE para UML

Gonzalo Génova, José M. Fuentes, Juan Llorens

Departamento de Informática, Universidad Carlos III de Madrid
Avda. Universidad 30, 28911 Leganés (Madrid), España
{ggenova, fuentes, llorens}@inf.uc3m.es
<http://www.ie.inf.uc3m.es/>

Resumen. Seleccionar una herramienta CASE para UML se ha convertido en una tarea ardua, dado el enorme número de productos que han aparecido en el mercado en los últimos años. En este artículo presentamos el trabajo realizado por un grupo de profesores y alumnos de la Universidad Carlos III de Madrid, que han desarrollado una metodología para evaluar comparativamente el grado de fidelidad al estándar de UML que presentan una serie de herramientas comerciales, especialmente en lo que se refiere a los aspectos notacionales. El resultado se presenta en forma de evaluación cuantitativa y cualitativa.

Palabras clave. UML, Herramienta CASE, Evaluación, Calidad.

Introducción

Desde que UML se ha establecido como un estándar *de facto* para el modelado orientado a objetos de sistemas informáticos, multitud de empresas se han lanzado a la carrera de conquistar el mercado de herramientas de apoyo a la ingeniería software, de tal manera que el solo número de productos existentes ya hace difícil la tarea de seleccionar el más conveniente para un uso concreto. La “guerra de los métodos” se ha transformado en una “guerra de herramientas”. La búsqueda en internet mediante palabras clave como “UML Tool” proporciona cientos o miles de enlaces a páginas que supuestamente proporcionan “la mejor herramienta del mercado, compatible con la última versión del estándar de UML”. Esta búsqueda se facilita acudiendo a catálogos específicos que enumeran las principales herramientas existentes, junto con las características principales que ofrece el fabricante. Destaca entre ellos el catálogo proporcionado por *Objects by Design* [7], que actualmente contiene algo más de un centenar de herramientas y detalla las siguientes características de cada una, proporcionando también enlaces a los sitios web respectivos:

- Empresa
- Producto (con breve descripción del mismo en un par de líneas)
- Versión y fecha
- Plataforma (Windows, Java VM, Linux, Unix, etc.)
- Precio (desde \$0 hasta \$10.000)

No obstante, el simple examen de estas características se revela insuficiente para fundamentar una elección acertada, ya que demasiado a menudo las promesas del fabricante resultan incumplidas en el producto real, incluso por parte de las empresas

que realizan una campaña de publicidad más agresiva. Por tanto, se hace necesaria una evaluación propia de las herramientas que a primera vista puedan parecer más convenientes para el cliente final. En otras ocasiones encontramos informes cualitativos (que siempre son necesarios) referidos a un conjunto de herramientas, pero que por su heterogeneidad dificultan la labor comparativa [6]. Nuestro propósito es proporcionar una metodología que ofrezca valoraciones cuantificadas.

¿Qué podemos esperar de una herramienta CASE para UML? UML es un lenguaje visual de modelado para “visualizar, especificar, construir y documentar los artefactos de un sistema software” [4, 10], es decir, UML es ante todo un lenguaje gráfico que estandariza la forma de crear diagramas, el significado preciso de los mismos, y las relaciones existentes entre ellos. Por tanto, lo primero que podemos esperar de una herramienta es que facilite la tarea de dibujar diagramas, su corrección sintáctica, y la coherencia entre los distintos diagramas. Esta tarea se ve obstaculizada desde el primer momento por los problemas que presenta el mismo estándar de UML, que no habiendo aún alcanzado su plena madurez, presenta todavía inconsistencias y lagunas importantes, que las herramientas deben resolver a menudo apartándose de “la letra” del estándar. Las distintas versiones del lenguaje que han aparecido en rápida sucesión a lo largo de la última década [8, 9], y a veces con cambios notables, también dificultan en gran medida que las herramientas puedan mantenerse al día en lo que respecta a la notación de UML.

Además de los aspectos puramente notacionales, habitualmente el usuario final espera encontrar otras muchas capacidades en la herramienta buscada, de modo que tenga utilidad real en su entorno de producción: integración con herramientas ofimáticas (como copiar y pegar los diagramas en documentos de texto), posibilidad de trabajo multiusuario (para que varios ingenieros puedan acceder simultáneamente a distintas partes de un modelo), exportación en formato XMI, y, especialmente, integración dentro del entero proceso de desarrollo de software, desde la obtención de requisitos de usuario hasta la generación automática de código, estimación de esfuerzo necesario para acometer la implementación de un modelo dado, planificación, mantenimiento, pruebas, etc. Señaladamente, la reutilización de todo tipo de artefactos software (no sólo código fuente o ejecutable, sino también diagramas de análisis o diseño, definición de pruebas, etc., y hasta requisitos) es una característica muy deseable de una herramienta completa de apoyo al proceso de desarrollo de software. El cliente final deberá definir bien sus necesidades y contrastarlas con el precio del producto ofrecido por el fabricante, pero no puede eludir la tarea de evaluar en concreto la herramienta para comprobar que efectivamente cumple las promesas del fabricante.

Trabajos realizados

Desde el entorno docente, en el *Information Engineering Group* del Departamento de Informática de la Universidad Carlos III de Madrid hemos acometido desde hace varios años el desarrollo de una metodología de evaluación de herramientas CASE para UML, centrándonos en primer lugar en los aspectos notacionales. La primera aproximación fue realizada en el año 2001 por Tom Boive y Peter Ordén [3], alumnos

del *Åland Institute of Technology* (Mariehamn, Finlandia), que en un periodo de intercambio realizaron un trabajo de investigación dirigido por los autores de este artículo, en el que se establecieron ya las bases de la metodología que hemos desarrollado en nuestro grupo, realizando una comparativa de tres herramientas comerciales: RATIONAL ROSE, MAGIC DRAW y VISUAL UML. El resultado, obtenido con la metodología que se explicará más adelante, fue favorable para MAGIC DRAW y bastante desfavorable para RATIONAL ROSE, ocupando un lugar intermedio la tercera herramienta evaluada, VISUAL UML.

Más adelante, en el año 2002, en el contexto de unas Prácticas Académicas Externas de nuestra Universidad desarrolladas en el seno de la empresa DTINF (Desarrollos para las Tecnologías de la Información), que actualmente se encuentra inmersa precisamente en el desarrollo de una herramienta de gestión del conocimiento y apoyo al proceso de desarrollo de software (CAKE - *Computer Aided Knowledge Environment*) [5], los alumnos Gaspar Blein Cuadrillero y Jesús Sancho Resino [2] extendieron la evaluación a un conjunto más amplio de herramientas, pero sin variar la metodología, e igualmente supervisados por los autores de este artículo. Las herramientas evaluadas fueron nueve, que se enumeran a continuación en orden de menor a mayor puntuación obtenida: RATIONAL ROSE, TOGETHER, POSEIDON, OBJECTEERING, MEGA SUITE, VISUAL UML, UMLCAKE, OBJECT DOMAIN y MAGIC DRAW. Nuevamente RATIONAL ROSE resultó ser la peor y MAGIC DRAW la mejor.

La tercera fase de este proyecto ha sido realizada por once alumnos de Cuarto Curso de la titulación Ingeniería Informática, como práctica voluntaria de la asignatura Ingeniería del Software I en el curso académico 2003-2004 [1]: Paloma Bello Millán, Gaspar Blein Cuadrillero, Javier García Cuesta, Raquel Gómez Fuentes, Francisco León Nieto, Francisco Javier Martínez Sanz, Mónica Revenga Becedas, Roberto Rosingana Merlo, Víctor Santos Morales, Ana Sanz Esteban y Francisco del Valle Hernández. Las herramientas evaluadas, nuevamente en orden ascendente de puntuación, han sido: PROXY DESIGNER, UML DIAGRAMMER, UMBRELLO UML MODELLER, VISUAL PARADIGM, VISIBLE ANALYST, MICROSOFT VISIO, ARGO UML, VISUAL UML, OBJECT DOMAIN, OBJECTEERING y MAGIC DRAW, a las que posteriormente se añadieron la evaluaciones de SECAKE (sucesora de UMLCAKE) y RATIONAL ROSE. En este caso, además de ampliar una vez más el conjunto de herramientas evaluadas, la metodología de evaluación ha sido revisada. En este artículo exponemos en detalle dicha metodología, así como los resultados obtenidos, junto con el necesario análisis de los mismos.

Teniendo en cuenta el enorme número de productos existentes en el mercado, ha sido necesario realizar una labor de selección previa a las evaluaciones. En las tres fases del proyecto, las herramientas han sido seleccionadas a partir del catálogo proporcionado por *Objects by Design* [7], atendiendo a factores como disponibilidad de versiones gratuitas de evaluación, facilidad de instalación, etc., así como la información proporcionada por el propio fabricante, descartando directamente, por ejemplo, aquellas herramientas que ni siquiera pretenden cubrir todos los tipos de diagramas de UML. Esta selección es necesariamente parcial, de modo que no podemos descartar que alguna herramienta no seleccionada ni evaluada pueda ser una buena herramienta CASE para UML.

Metodología de evaluación

Como ya hemos comentado, hasta el momento nos hemos centrado en los aspectos notacionales para evaluar las herramientas seleccionadas, ya que encontrar una herramienta que utilice fielmente la notación de UML es nuestro primer objetivo para la docencia. El método consiste básicamente en formular una serie de cuestiones sobre la fidelidad de las herramientas a la notación de UML, y contabilizar el número de respuestas afirmativas y negativas obtenidas por cada herramienta. Ahora bien, es muy difícil cuantificar absolutamente hasta qué punto una herramienta es capaz de representar correctamente los distintos elementos que pueden aparecer en los diagramas de UML: no sería práctico enumerar *a priori* todas las características buscadas, y a continuación contrastarlas en las herramientas sometidas a evaluación, ya que la enumeración podría extenderse hasta el infinito, y posiblemente la mayoría de las cuestiones serían respondidas afirmativamente por todas las herramientas. Por ejemplo, si planteamos la pregunta “¿es posible representar una clase como un rectángulo con tres compartimentos, reservados al nombre de la clase, los atributos y las operaciones respectivamente?”, probablemente en la totalidad de las herramientas la respuesta será afirmativa, de modo que la pregunta no resulta relevante para discriminar la calidad de las distintas herramientas.

En la primera fase de este proyecto [3] el método seguido para encontrar un conjunto de preguntas relevantes fue el siguiente:

1. Se selecciona un conjunto de diagramas significativos, ordenados por tipo de diagrama, que reflejen la totalidad de la notación de UML, extraídos de los primeros libros publicados por los autores originales de UML, la Guía del Usuario [4] y el Manual de Referencia [10].
2. Se intenta representar estos diagramas utilizando las herramientas seleccionadas.
3. Cuando se detecta algún problema con una herramienta concreta, se formula de modo sintético la pregunta correspondiente, y se anota la respuesta negativa proporcionada por esta herramienta. Por ejemplo: “¿es posible representar una dependencia entre dos asociaciones?”
4. La misma pregunta se plantea a las demás herramientas, de modo que se pueda establecer la comparación entre ellas.
5. Finalmente, se recopila en una tabla la lista de preguntas con las correspondientes respuestas afirmativas o negativas proporcionadas por todas las herramientas.

De esta manera se obtuvo una lista de 109 preguntas, con un número de respuestas afirmativas entre 19 y 60. Como puede observarse, esta lista de preguntas contiene sólo aquellas que han sido respondidas negativamente al menos por una de las herramientas, de modo que el resultado final no constituye una medida absoluta de la fidelidad de cada herramienta a la notación de UML, pero sí una útil medida comparativa entre ellas. Si una herramienta responde afirmativamente a 35 preguntas de 100, esto no significa que sea capaz de representar el 35% de la notación de UML, sino el 35% de las preguntas que al menos una de ellas ha respondido negativamente. Si, en un intento por obtener una medida absoluta, a esta lista de 100 preguntas añadiéramos otras 900, respondidas afirmativamente por todas las herramientas, el resultado sería un 93,5% de fidelidad, frente a otros porcentajes todos ellos por

encima de 90%. Con este método de selección de preguntas (que hemos denominado “criterios negativos”) ahorramos el trabajo de formular y contabilizar esas 900 preguntas que al fin y al cabo no son significativas porque no sirven para discriminar unas herramientas de otras, centrándonos en el tramo que sí es relevante.

En la segunda fase de este proyecto [2] se aplicó la misma metodología, aplicándola a seis nuevas herramientas, además de las tres anteriores, para las cuales se repitieron las pruebas. La lista de preguntas se incrementó hasta 124, gracias a las dificultades planteadas por las nuevas herramientas incorporadas a la evaluación, con resultados desde 44 hasta 80 respuestas afirmativas.

El problema más importante detectado en estas dos fases consistió en que el punto de referencia adoptado para realizar la evaluación resultó ser inadecuado. Efectivamente, tanto la Guía del Usuario [4] como el Manual de Referencia [10] se corresponden aproximadamente con la versión 1.3 del Estándar de UML [8], que fue publicada al año siguiente que estos dos libros. En no pocas ocasiones se detectó que los diagramas extraídos de estos libros y utilizados como prueba no eran compatibles con las definiciones del Estándar: a veces porque exigían más, a veces porque exigían menos, y a veces porque eran contradictorios con la notación “oficial”. Además, la versión 1.3 fue pronto sustituida por la versión 1.4, que introdujo cambios significativos en la notación, y posteriormente por la versión 1.5, aunque esta última tiene menos relevancia desde el punto de vista notacional, probablemente porque estando ya muy avanzados los trabajos para la nueva versión 2.0 no valía la pena invertir demasiado esfuerzo en ella, ya que rápidamente quedaría obsoleta. De modo que era necesario repetir de nuevo las evaluaciones, tomando como referencia la propia definición estándar del lenguaje, tarea que acometimos en la tercera fase de este proyecto. En el momento de realizarse las evaluaciones la versión 2.0 todavía no había sido definitivamente aprobada, y además ninguna de las herramienta pretendía ser compatible con ella, de modo que se adoptó la versión 1.4 como referencia común [8]. Así, en la tercera fase de este proyecto [1], realizada a partir de los diagramas de la Sección 3 del Estándar (“UML Notation Guide”) sobre once herramientas, la lista de preguntas respondidas negativamente por alguna de ellas ascendió a 282, con resultados afirmativos entre 71 y 218 (una de las herramientas obtuvo una puntuación de 250, pero este valor no debe considerarse significativo, tal como se explicará más adelante).

Resultados numéricos

A continuación presentamos de modo sintético los resultados numéricos obtenidos en las tres fases de este proyecto.

Herramienta	Puntuación	Porcentaje
Rational Rose	19	17
Visual UML	44	40
Magic Draw	60	55

Tabla 1. Resultados numéricos de la primera fase (total: 109 preguntas)

Herramienta	Puntuación	Porcentaje
Rational Rose	44	35
Together	51	41
Poseidon	54	44
Objecteering	59	48
Mega Suite	60	48
Visual UML	65	52
umICAKE	71	57
Object Domain	75	60
Magic Draw	80	65

Tabla 2. Resultados numéricos de la segunda fase (total: 124 preguntas)

Herramienta	Puntuación	Porcentaje	Categoría
Proxy Designer	185	66	Gráfica
UML Diagrammer	250	89	
Umbrello UML Modeller	84	30	Sintáctica
Visual Paradigm	109	39	
Visible Analyst	163	58	
Microsoft Visio	179	63	
Argo UML	71	25	Semántica
Rational Rose	110	39	
Visual UML	152	54	
Object Domain	155	55	
seCAKE	158	56	
Objecteering	160	57	
Magic Draw	218	77	

Tabla 3. Resultados numéricos de la tercera fase (total: 282 preguntas)

Como la tercera fase tiene más interés, detallamos a continuación el porcentaje de respuestas afirmativas en cada tipo de diagrama.

Herramienta	Diagrama	Estruct.	C. Uso	Inter.	Estad.	Activ.	Implem.
	Preguntas						
Proxy Designer	77	86	90	0	0	80	
UML Diagrammer	82	86	97	89	100	93	
Umbrello UML Modeller	39	86	23	14	20	0	
Visual Paradigm	31	100	29	60	55	60	
Visible Analyst	55	86	63	46	50	80	
Microsoft Visio	60	86	75	49	70	53	
Argo UML	28	29	22	11	30	40	
Rational Rose	42	43	22	52	75	27	
Visual UML	56	86	45	34	80	73	
Object Domain	58	71	53	49	53	50	
seCAKE	53	71	48	83	65	47	
Objecteering	55	43	53	62	90	47	
Magic Draw	70	100	70	77	89	80	

Tabla 4. Porcentaje de respuestas afirmativas por tipo de diagrama en la tercera fase

Análisis de los resultados

Como puede observarse en la Tabla 3, las herramientas no aparecen estrictamente en orden de puntuación menor a mayor, sino divididas en tres grupos remarcados con un borde más grueso, correspondientes a tres categorías que hemos denominado Gráfica, Sintáctica y Semántica.

Las herramientas meramente *gráficas* son aquellas que proporcionan algún tipo de ayuda para dibujar diagramas UML (como plantillas de formas predefinidas, etc.), de modo que resultan algo más convenientes que otras herramientas genéricas de dibujo (como podría ser MICROSOFT POWERPOINT). El problema es que no imponen prácticamente ningún tipo de restricción en lo que el usuario puede dibujar, de modo que se facilita la construcción de diagramas UML, pero no se garantiza en absoluto su corrección sintáctica. Por ejemplo, podemos subrayar un atributo para indicar que es estático, pero igualmente podemos utilizar un subrayado doble, que carece de significado en UML. Precisamente por su ausencia de restricciones es posible lograr una puntuación muy elevada con este tipo de herramientas (como es el caso de UML DIAGRAMMER), pero no debemos dejarnos engañar por este resultado: una buena herramienta CASE no sólo debe ayudar a dibujar diagramas, sino diagramas *correctos*. Por tanto, no pensamos que éstas sean buenas herramientas CASE para UML: es más, no alcanzan ni siquiera el mínimo exigible.

Las herramientas *sintácticas* son aquellas que, en general, sólo permiten dibujar diagramas correctos según las reglas notacionales de UML (o al menos lo intentan). Por tanto, suponen una ayuda mayor para el usuario respecto a las herramientas meramente gráficas. No obstante, en general estas herramientas no construyen internamente un modelo a la vez que los diagramas que lo expresan, de modo que los diagramas quedan desconectados unos de otros: de alguna manera, son diagramas correctos pero sin *significado*, sin un referente común. Por ejemplo, no es posible exigir que, para representar un mensaje de invocación de operación en un diagrama de interacción, exista la correspondiente operación en un diagrama de clases. Esto constituye un inconveniente importante, porque impide comprobar la coherencia entre los distintos diagramas que expresan un mismo modelo desde distintos puntos de vista. Por tanto, pensamos que éstas tampoco deben considerarse herramientas CASE satisfactorias para UML.

El tercer grupo lo constituyen las herramientas *semánticas*, es decir, aquellas que tratan de garantizar la construcción de un modelo que esté correctamente expresado en diagramas que además sean coherentes entre sí. Aunque en algunos casos la puntuación pueda ser inferior a la obtenida por herramientas de los otros grupos, es dentro de esta categoría donde encontramos las que con propiedad pueden llamarse herramientas CASE para UML. Notemos que, de las siete herramientas evaluadas, ARGO UML tiene una puntuación muy inferior a las demás (de hecho, es la que menos puntuación obtiene en todo el grupo, aunque otras características de esta herramienta la hacen interesante); a continuación, ocupando el penúltimo lugar dentro de esta categoría, aparece RATIONAL ROSE; las cuatro herramientas del grupo intermedio obtienen puntuaciones muy similares; y sólo MAGIC DRAW destaca por su elevada puntuación (sólo superada globalmente por una de las herramientas meramente gráficas).

Aunque las fronteras entre los tres grupos puedan ser hasta cierto punto difusas, lo cierto es que constituyen tres categorías heterogéneas, entre las que una comparación meramente numérica resultaría injusta. Las herramientas de la categoría semántica son mucho más ambiciosas que las de la categoría gráfica, y lógicamente se encuentran con mayores dificultades para resolver adecuadamente los problemas notacionales de UML; pero, cuando lo hacen, lo hacen mucho mejor.

Hasta ahora sólo hemos hecho referencia a los resultados cuantitativos (puntuación) y cualitativos (categoría) de la evaluación, que deberían orientar en la selección de una herramienta CASE entre las doce evaluadas. Pero este proyecto ha producido un resultado más importante si cabe, que es la lista de 282 preguntas concretas obtenidas en la tercera fase. En efecto, con esta lista en la mano podemos evaluar otras herramientas de modo mucho más metódico, y disponiendo además de unos valores previos que nos sirven de marco de referencia para valorar los resultados que obtengamos. De hecho, éste ha sido el procedimiento seguido con las herramientas SECAKE y RATIONAL ROSE que fueron evaluadas utilizando las preguntas obtenidas a partir de las otras once herramientas.

Trabajos futuros

Tenemos intención de continuar este proyecto en distintas direcciones. En primer lugar, nos gustaría ampliar el conjunto de herramientas evaluadas, descartando de entrada aquéllas que no entren en la categoría semántica, pero incluyendo otras que en la tercera fase no ha sido posible evaluar por diversos motivos, como la falta de una versión adecuada para la evaluación, dificultades en la instalación de la herramienta, etc. Entre éstas se encuentran productos conocidos como TOGETHER, MEGA SUITE, RATIONAL ROSE, RHAPSODY, POSEIDON, FUJABA, TAU, etc.

Además, junto a los criterios empleados, que se refieren casi exclusivamente a las capacidades notacionales de las distintas herramientas, queremos incluir también criterios de usabilidad en la edición de los diagramas y otros aspectos mencionados en la introducción (integración con herramientas ofimáticas, posibilidad de trabajo multiusuario, coordinación con otras fases del proceso de desarrollo de software, reutilización, generación automática de código, exportación en formato XMI, etc.). Naturalmente, cuando las herramientas comerciales empiecen a adaptarse a la nueva versión 2.0 del estándar de UML, habrá que adaptar la lista de criterios obtenidos y repetir las evaluaciones para las nuevas versiones de las herramientas ya estudiadas.

Por otra parte, la metodología empleada tiene algunas limitaciones que intentaremos superar en el futuro. El sistema de preguntas con respuesta Si/No puede ser inadecuado, ya que en muchos casos la respuesta no es tan sencilla. Por ejemplo, algunas herramientas permiten definir en el modelo subyacente determinadas propiedades de los elementos de un diagrama, aunque no sea posible mostrar estas propiedades en los diagramas; otras herramientas (las meramente gráficas según nuestra clasificación) permiten mostrar determinadas propiedades aunque sin definir las formalmente como tales en el modelo. Otra posible mejora sería dar mayor peso a unos criterios que a otros en la evaluación global, ya que no es tan importante, por ejemplo, la posibilidad de ocultar unas operaciones y mostrar otras (para reflejar

distintos grados de abstracción en la representación de la clase), como la posibilidad de mostrar que una operación es abstracta. En todo caso, es discutible si esta ponderación influiría en el resultado final, ya que las herramientas que son buenas en un aspecto tienden a serlo en los demás; por otra parte, habría que añadir estos factores de ponderación en las tablas de evaluación, con el consiguiente trabajo añadido y el peligro de introducir arbitrariedades.

La consideración de las reglas de coherencia entre diagramas tiene mucho interés, ya que conduce a la construcción de modelos que son globalmente correctos, no sólo a nivel de diagrama. Se trata de un aspecto que ya no es puramente notacional sino más bien semántico, y que tropieza con indudables dificultades, ya que en este aspecto el propio lenguaje UML contiene aún importantes lagunas e indefiniciones, por ejemplo en la relación entre operaciones, mensajes, eventos y acciones.

Conclusión

Lo cuantitativo no lo es todo en la tarea de seleccionar una herramienta CASE. No obstante, resulta de gran ayuda si se combina adecuadamente con apreciaciones cualitativas. Nuestra clasificación de herramientas en tres categorías (gráfica, sintáctica y semántica) es un primer paso en esta dirección.

La metodología que hemos desarrollado puede emplearse para seleccionar herramientas de acuerdo con las necesidades y expectativas del usuario. Más aún, esperamos que sirva para mejorar la calidad de las herramientas que ya existen en el mercado, especialmente de aquéllas que están más seriamente comprometidas con la fidelidad al estándar de UML.

Anexo I

Versiones de las herramientas evaluadas

Primera fase

- MagicDraw UML Standard Edition (No Magic)
- Rational Rose 2000 (Rational)
- Visual UML Standard Edition (Visual Object Modelers)

Segunda fase

- MagicDraw UML Standard Edition 5.5 (No Magic)
- Mega Suite 5.3 (Mega International)
- Object Domain R3 Evaluation Edition 3.0 (Object Domain Systems)
- Objecteering Personal Edition 5.2 (Softeam)
- Poseidon for UML Community Edition 1.3 (Gentleware)
- Rational Rose 2002 Enterprise Edition (Rational)
- Together Whiteboard (TogetherSoft)
- Visual UML Standard Edition 2.9 (Visual Object Modelers)

Tercera fase

- Argo UML 0.14 (Tigris)
- MagicDraw UML Standard Edition 7.1 (No Magic)
- Microsoft Visio 2003 (Microsoft)
- Object Domain R3 Evaluation Edition 3.0 (Object Domain Systems)
- Objecteering Personal Edition 5.2 (Softeam)
- Proxy Designer 1.0.1 (ProxySource)
- Rational Rose 2002 Enterprise Edition 5.20 (Rational)
- seCAKE 1.4.1 (dTinf)
- UML Modeller 1.2 (Umbrello)
- UML Diagrammer 4.16 (Pacestar)
- Visible Analyst 7.5 (Visible Systems)
- Visual Paradigm for UML 3.0 Community Edition (Visual Paradigm)
- Visual UML Standard Edition 3.2 (Visual Object Modelers)

Referencias

- [1] Paloma Bello Millán, Gaspar Blein Cuadrillero, Javier García Cuesta, Raquel Gómez Fuentes, Francisco León Nieto, Francisco Javier Martínez Sanz, Mónica Revenga Becedas, Roberto Rosingana Merlo, Victor Santos Morales, Ana Sanz Esteban, Francisco del Valle Hernández. *Evaluación de herramientas CASE para UML*. Informe Interno del Departamento de Informática, Universidad Carlos III de Madrid, 2004.
- [2] Gaspar Blein Cuadrillero, Jesús Sancho Resino. *Actualización de UML CASE Tool Review*. Informe Interno del Departamento de Informática, Universidad Carlos III de Madrid, 2002.
- [3] Tom Boive, Peter Ordén. *UML CASE Tool Review*. Informe Interno del Departamento de Informática, Universidad Carlos III de Madrid, 2001.
- [4] Grady Booch, James Rumbaugh, Ivar Jacobson. *The Unified Modeling Language User Guide*. Addison-Wesley, 1998.
- [5] Desarrollos para las Tecnologías de la Información (<http://www.dtinf.es>).
- [6] Bruce Eckel. UML Tool Survey. (<http://www.mindview.net/WebLog/log-0041>).
- [7] Objects by Design Inc. (<http://www.objectsbydesign.com/>).
- [8] Object Management Group. *Unified Modeling Language Specification*, Version 1.5, March 2003 (Version 1.3, June 1999, Version 1.4, September 2001).
- [9] Object Management Group. *Unified Modeling Language Superstructure Specification*, August 2003, ptc/03-08-02.
- [10] James Rumbaugh, Ivar Jacobson, Grady Booch. *The Unified Modeling Language Reference Manual*. Addison-Wesley, 1998.