

**enter**<sup>ie</sup>

# Modelo económico del software

Noviembre, 2007



# Índice

---

<b>1. Introducción</b>	<b>3</b>
<b>2. Mercado del software</b>	<b>7</b>
2.1. El mito del sector concentrado	7
2.2. La oferta de software	11
2.3. La demanda de software	14
2.4. El precio	19
<b>3. Efectos micro y macroeconómicos del software</b>	<b>25</b>
3.1. Software propietario comercial vs. OSS	25
3.2. Aproximación microeconómica	30
3.3. Enfoque macroeconómico	36
<b>4. El papel de las AAPP ante el software libre y el comercial</b>	<b>41</b>
4.1. ¿Está justificada la intervención pública? Los fallos de mercado	41
4.2. El análisis de costes y beneficios y el principio de la neutralidad tecnológica	46
4.3. Experiencias internacionales de promoción pública del software libre	48
4.4. Algunos ejemplos del posicionamiento de España	55
<b>Conclusiones</b>	<b>59</b>
<b>Referencias bibliográficas</b>	<b>63</b>



# 1.



## Introducción

---

### **Naturaleza del software. Software como bien económico**

**Desde un punto de vista económico, el software** es considerado a menudo como un tipo especial de bien basado en la información. Los bienes basados en la información son la materialización del conocimiento, y normalmente se les sitúa en un punto intermedio entre los privados puros y los públicos puros, ya que comparten características de ambos en distintos grados.

Como la información, el software es un bien inmaterial, lo que lo distingue de la mayor parte de los bienes producidos en la economía. Sin embargo, tanto el software como la información necesitan una infraestructura de soporte material para poder ser útiles. La información tiene valor como consecuencia de su capacidad de *informar*, pero precisa de un soporte material para su almacenamiento, transporte y acceso. De forma similar, el valor del software deriva de lo que puede *hacer*, pero necesita algún tipo de hardware para realizar sus funciones.

El software, por tanto, combina características que lo sitúan a medio camino entre los bienes materiales y los basados en la información. Al igual que la información, puede ser replicado con unos costes marginales prácticamente nulos, pero su alto grado de funcionalidad (valor derivado de lo que puede hacer) lo diferencia de la información como bien económico. Del mismo modo, los

**Tabla 1.1. Clasificación económica de los bienes de acuerdo con su naturaleza**

		Grado de rivalidad en el consumo	
		Bienes rivales	Bienes no rivales
Posibilidad de exclusión mediante precio	Bienes excluibles	<b>Galletas</b>	<b>Señal de TV por cable</b>
	Bienes parcialmente excluibles	-	<b>Software</b>
	Bienes no excluibles	<b>Peces en el mar</b>	<b>Teorema de Pitágoras</b>

Fuente: Romer (1993)

productos de software también disponen de algunas características propias de los bienes físicos.

Por estas razones, el software se configura como un tipo de bien peculiar. En la tabla 1.1 pueden verse ejemplos de los distintos tipos de bienes, de acuerdo con la clasificación económica tradicional y generalmente aceptada, en la que el software aparece como un bien para el que no existe rivalidad en el consumo (su empleo por parte de un usuario no impide que otro lo pueda emplear también) y que es parcialmente excluible a través de mecanismos de precios.

La primera cuestión que debe aclararse es si esta caracterización puede considerarse válida. El software tiene, sin duda, características propias de los bienes basados en la información, pero también de los bienes materiales. Fruto de esta dualidad, se producen combinaciones inusuales de propiedades de ambos tipos de bienes que confluyen en el software.

Una muestra de este hecho es que los productos de software, al igual que la información, pueden ser replicados con unos costes marginales próximos a cero.

Pero, al mismo tiempo, los productos de software y otras soluciones basadas en el software exhiben un grado de funcionalidad de la que carecen los bienes basados puramente en la información. Así, mientras la información es valorada por su capacidad para *informar* o influir, el software es valorado por lo que es capaz de hacer él mismo.

Esta última propiedad es la que ha llevado a algunos autores —Mäkelä, M. M. (2005); Messerschmitt, D. G. y C. Szyperski (2003)— a afirmar que el software se valora por su *comportamiento*. En este sentido, se parece a la maquinaria, arquetipo de los bienes físicos.

Sin embargo, el software puede distinguirse del resto de productos industriales en una serie de dimensiones, incluso en el caso de las industrias altamente tecnológicas.

En primer lugar, la inversión en I+D necesaria para generar el software se realiza fundamentalmente en la fase de desarrollo, mientras que los costes de producción son prácticamente cero.

Además, aunque los productos de software son *productos* en sentido estricto, tienen un

comportamiento similar al de los servicios: son inmateriales y requieren de un proveedor (humano o mecánico) para transmitir su contenido. Además, el software puede ser personalizado y adaptado a las necesidades de los distintos segmentos de mercado, áreas de negocio y clientes (Mäkelä, M. M., 2005).

Por otra parte, las externalidades de red desempeñan un papel fundamental en este sector; papel que ha incrementado su importancia con Internet y el canal electrónico de transmisión provisto por la Red.

El papel de las autoridades es especialmente relevante cuando se trata el tema de la información y las creaciones en el mercado. Desde hace tiempo quedó claro que los mercados de la información y las creaciones

e invenciones, en los cuales se incardina el software, no pueden funcionar del mismo modo que lo hacen los mercados de mercancías físicas.

La razón es que el conocimiento tiene la propiedad que en economía se denomina como no rivalidad; es decir, que el uso del conocimiento por parte de una persona no impide que otra lo pueda emplear también. Además, por su propia naturaleza, es difícil excluir del uso del conocimiento o las creaciones e invenciones a personas no autorizadas. Los economistas llaman bienes públicos a los que verifican estas características. Como consecuencia, en el mercado del software será necesaria la actuación de las autoridades en determinadas materias, como el respeto a los derechos de propiedad, cuestión que será tratada de forma específica en el capítulo 4 □



# 2.

## Mercado del software

---

### 2.1. El mito del sector concentrado

**Probablemente, más de uno dudaría de la salud mental de quien dijese que sectores como las zapaterías o las librerías están más concentrados que el software.** Pues bien, ésta parece ser la realidad en Estados Unidos, según su oficina estadística (Bureau of Economic Analysis, <http://bea.gov/>).

Y es que uno de los mitos más extendidos en el ámbito del sector del software es que se trata de un sector altamente concentrado, donde unas poquísimas empresas concentran gran parte del mercado, llegando a una situación de oligopolio de facto.

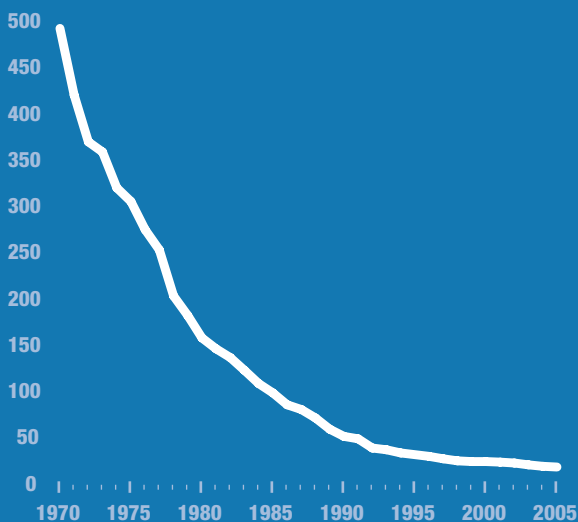
Este tipo de argumento es muy utilizado por determinados grupos de interés (léase, los defensores del software libre), que promueven el establecimiento de medidas especiales de protección de sus actividades, apoyándose en la supuesta falta de competencia en el sector. Sin embargo, los últimos datos de concentración en Estados Unidos parecen desmentir este argumento. La realidad indica que el sector del software comercial está relativamente poco concentrado, tanto dentro del gran sector de la información como en comparación con otros grandes sectores de la economía norteamericana.

## Producción y precios no indican alta concentración

La teoría económica predice que en un mercado con competencia limitada se va a restringir artificialmente la producción para poder mantener precios más elevados que los de competencia perfecta. De esta forma, los productores conseguirán obtener beneficios extraordinarios respecto a la situación de competencia.

En el mercado norteamericano, la tendencia de producción y precios parece haber sido muy distinta. La evolución del precio medio de los productos de software ha descendido drásticamente desde las décadas ya lejanas de los 70 y 80 (gráfico 2.1).

### 2.1. Evolución de los precios del software en EEUU (1985=100)

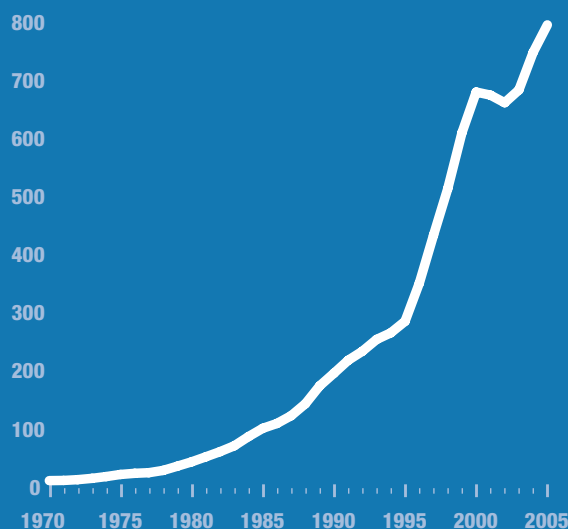


Fuente: Bureau of Economic Analysis (2006)

Incluso en los últimos diez años han seguido produciéndose notables abaratamientos de los productos de este mercado. Desde 1995, el software se ha abaratado un 40 por 100 de media en Estados Unidos. Y en los últimos cinco años, el descenso de precios ha sido del 23 por 100.

Esta reducción de precios ha venido acompañada de un incremento prácticamente ininterrumpido de la producción, con la única salvedad de la crisis de las TIC de primeros años de la década actual. En los últimos veinte años la producción se ha multiplicado por 8 (gráfico 2.2) y desde 1995 el crecimiento ha sido del 180 por 100.

### 2.2. Evolución del valor de la producción de software en EEUU (1985=100)



Fuente: Bureau of Economic Analysis (2006)

La evolución de los precios y la producción cuestiona, por tanto, el argumento de que el sector del software no se ha comportado de forma competitiva. A la luz de estos datos no es de extrañar que, cuando se comparan sus cifras con las de otros sectores, nos encontremos con que la concentración empresarial en el software, si se caracteriza por algo, es por su reducido nivel.

## La concentración real del sector

Una medida estándar para medir la concentración es el índice de Herfindahl-Hirschman (HHI), que ofrece una escala que oscila entre 0, para el caso de mínima concentración,

hasta 10.000 para el caso de monopolio. El sector norteamericano del software<sup>1</sup>, presentaba en el año 2000 un índice HHI de 244.

Para juzgar si el valor alcanzado es elevado o no respecto a otros sectores, basta mencionar que más de la mitad de los de la economía estadounidense presentó índices HHI superiores a los del software. Algunos de los capítulos más importantes en dicha economía, como el automóvil o los cereales, mostraron valores de concentración muy superiores a los del software: 2.506 y 2.446, respectivamente<sup>2</sup>.

## El software dentro del macrosector de información

La contabilidad nacional norteamericana, en su identificación de grandes sectores económicos, asigna el código NAICS 51 al de la información. Cuando se desciende hasta sectores al nivel de cuatro dígitos, nos

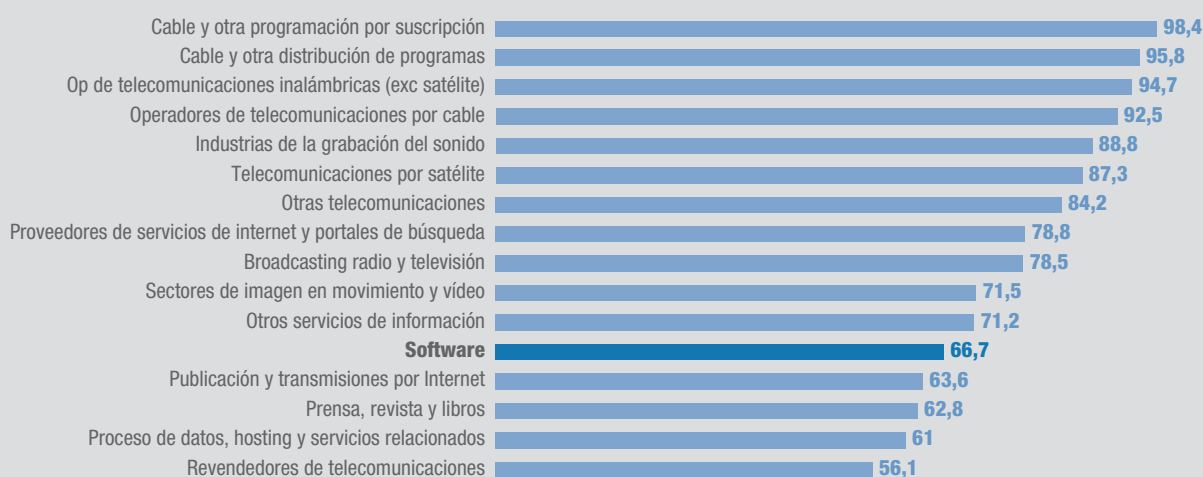
encontramos con 16 mercados, uno de los cuales es el de software (5112).

Dentro del macrosector de la información, el software ofrece unos niveles de concentración bastante reducidos, tal y como se observa en los gráficos 2.3 y 2.4. En primer lugar, cuando se consideran sus 50 mayores empresas (gráfico 2.3), se comprueba que la facturación conjunta asciende al 66,7 por 100 del total del sector.

Con esa cifra, el software se sitúa como el quinto sector menos concentrado dentro del macrosector de la información. En el extremo contrario se sitúan el cable y las telecomunicaciones, que resultan los más concentrados dentro de aquél.

Se puede argumentar que un análisis considerando las 50 mayores empresas puede ocultar la existencia de unas pocas que captan la mayor parte de la cuota de mercado. En otras palabras, y yendo al ejemplo

### 2.3. Sector de la Información en EEUU. Proporción de la facturación de cada subsector (NAICS 4 dígitos) generada por las 50 mayores empresas 2002, %



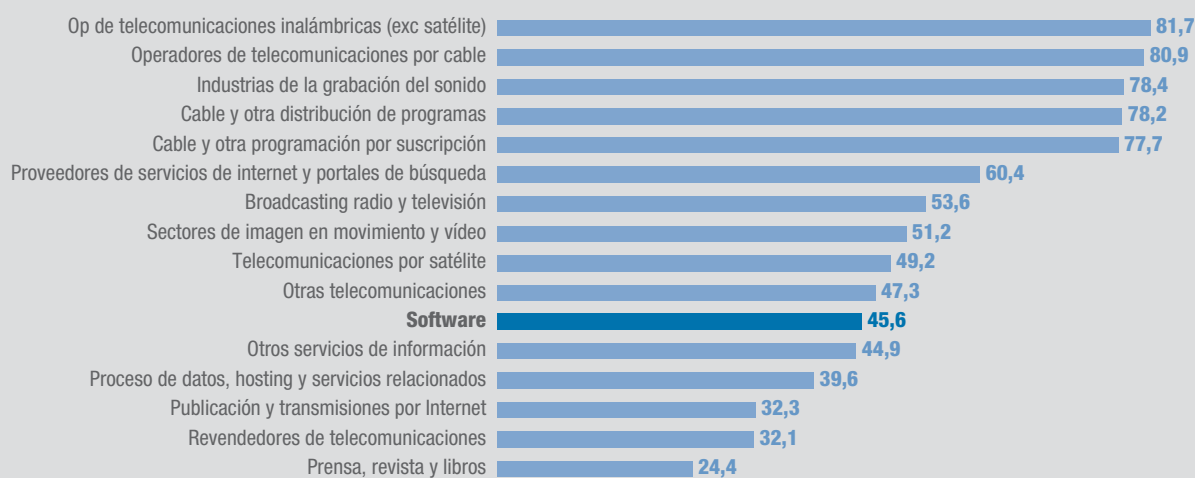
Fuente: Bureau of Economic Analysis (2006). US Economic Census 2002

<sup>1</sup> El sector del software en Estados Unidos (código 5112 de la North American Industry Classification System, NAICS) se define del siguiente modo: 'establishments in this industry carry out operations necessary for producing and distributing computer software, such as designing, providing documentation, assisting in installation, and providing support services to software purchasers. These establishments may design, develop, and publish, or publish only.'

<sup>2</sup> Padilla, A. J. et al. (2004).

## 2.4. Sector de la Información en EEUU. Proporción de la facturación de cada subsector (NAICS 4 dígitos) generada por las 8 mayores empresas

2002, %



Fuente: Bureau of Economic Analysis (2006). US Economic Census 2002

concreto, puede darse el caso de que media docena de empresas de software representen una proporción muy elevada del mercado y las otras 44 apenas añadan cuota a esa cifra total del 66,7 por 100.

Para solucionar dicho problema se ha construido el gráfico 2.4, en el que se plantea el mismo análisis, pero en vez de considerar la cuota de las mayores 50 empresas de cada mercado sólo se tiene en cuenta la de las ocho mayores.

El resultado es que esas ocho mayores empresas generan una facturación que representa el 45,6 por 100 del total. Sólo cinco sectores, dentro del macrosector de la información, están menos concentrados a este nivel.

La conclusión es que, dentro del ámbito de los sectores TIC, resulta al menos cuestionable señalar al sector del software comercial como uno de los más concentrados. La realidad apunta más bien hacia lo contrario. Y este hecho también se sostiene cuando se estudian las concentraciones en sectores muy alejados del mundo de la información o las telecomunicaciones.

## Comparación con otros sectores

El software, dentro del macrosector de la información, no se caracteriza por un elevado grado de concentración, tal y como se ha visto. A una conclusión similar se llega cuando se compara con otros sectores de la economía estadounidense, sobre muchos de los cuales no es habitual tener una imagen de sector concentrado.

La tabla 2.1 refleja algunos ejemplos al mismo nivel de desagregación contable (cuatro dígitos NAICS) que pueden compararse con el sector del software. Los resultados resultan sorprendentes a primera vista, puesto que las empresas dedicadas a la venta de zapatos, las tiendas de venta de electrodomésticos o las librerías, por citar sólo tres ejemplos, presentan niveles de concentración superiores a todos los niveles a los que se producen en el software.

Otros sectores, como el transporte aéreo o los vehículos a motor y componentes, también presentan niveles de concentración empresarial mucho más elevados que los del software, aunque esto podría resultar más previsible.

**Tabla 2.1. Concentración empresarial en sectores escogidos de la economía de EEUU (2002)**

Código NAICS	Sector	Proporción de mercado cubierta por las mayores:			
		4 empresas	8 empresas	20 empresas	50 empresas
5112	Software	39,5	45,6	56,3	66,7
4811	Transporte aéreo	32,6	48,7	67,6	85,5
4231	Vehículos a motor y componentes	53,6	67,4	76,1	81,8
4482	Zapaterías	39,9	52,4	68,1	76,3
4512	Librerías y tiendas de música	62,3	67,3	70,6	74,1
4461	Farmacias	52,8	61,0	65,6	68,4
4431	Almacenes electrónica-electrodomésticos	53,6	58,1	64,7	67,7
4451	Supermercados ( <i>Grocery Stores</i> )	31,0	43,4	54,5	65,3
4481	Almacenes de ropa	28,0	37,8	52,0	65,3

Fuente: enter a partir de Bureau of Economic Analysis (2006), US Economic Census 2002

El riesgo de crear una imagen de un sector que no corresponde con la realidad puede tener consecuencias perjudiciales. Un ejemplo reciente procede de la Unión Europea, donde las autoridades en materia de competencia, tanto estatales como comunitarias, han cargado en varios casos contra este sector de manera injustificada (Bote, V., 2006).

## 2.2. La oferta de software

### Elevados costes fijos, economías de escala e inversiones arriesgadas

El software es un bien basado en la información, cuya producción requiere elevados costes fijos en una fase muy temprana del proceso que, una vez concluida, permite replicar el producto de forma intensiva a un coste muy reducido. Se trata, además, de un tipo de bien para el que no existe rivalidad en el consumo; factor que, como veremos, está directamente relacionado con el riesgo de las inversiones para producirlo.

La producción de software es una actividad caracterizada por la existencia de *economías de escala*. La existencia de éstas es una peculiaridad propia de algunos mercados en los que los productores pueden incrementar la producción soportando incrementos de coste menos que proporcionales. La razón por la que sucede en el mercado del software deriva directamente de que para crear un producto es preciso incurrir en elevados costes en la fase de diseño (costes fijos elevados), a lo que se unen los muy reducidos ligados a la réplica y distribución del producto. Es decir, lo difícil y costoso es obtener la primera unidad del producto, ya que en adelante los costes marginales son prácticamente nulos.

Las economías de escala permiten a un proveedor con elevada cuota de mercado fijar precios más atractivos para los consumidores ya que, al tener un gran volumen de producción, los costes medios se han reducido sustancialmente. En otros términos, los elevados costes fijos se han diluido entre un gran número de consumidores. Al igual que con los efectos de red (véase sección 2.3), las economías de escala son un efecto que alimenta la dinámica de incrementos en las cuotas de mercado en el sector del software.

La consecuencia inmediata de la existencia de economías de escala en el mercado del

software es que éste no puede caracterizarse en ningún caso por un comportamiento de pura competencia, ya que en dicha situación el precio iguala el coste marginal (que es prácticamente nulo) y un productor no podría recuperar nunca los costes fijos iniciales, con lo que estaría condenado a sufrir pérdidas continuas, por lo que le resultaría óptimo no producir y abandonar el mercado.

### Economía de alcance y grandes empresas

Una estrategia que acometen las empresas en sectores como el software, con costes fijos elevados y marginales reducidos, es la producción de varios productos interrelacionados, que comparten los costes iniciales. El fenómeno conocido como *economías de alcance* hace referencia a este hecho: se trata del caso en el que una empresa que produce dos productos alcanza, debido a las ganancias de eficiencia, mayores niveles de producción que dos empresas distintas que estuviesen cada una especializada en la fabricación de un único producto.

Las economías de alcance están presentes en numerosos sectores. Es habitual que las empresas automovilísticas produzcan vehículos de distinta naturaleza: no sólo automóviles de distinta gama, sino también furgonetas y camiones. De esta forma, los elevados costes fijos que entraña poner en marcha una actividad productiva de esta naturaleza quedan repartidos entre los distintos productos, lo que supone un ahorro de coste, al aprovechar en unos productos los esfuerzos realizados en materia de investigación en otros.

Otro ejemplo clásico es el sector del transporte de mercancías por carretera, para el que se demostró que una empresa tenía que ser suficientemente grande para poder combinar cargas en puntos intermedios del recorrido, llevar sus camiones proporcionalmente más llenos que los de una empresa pequeña, reducir así los costes e incrementar la rentabilidad. En definitiva, una empresa grande podía competir en mejores condiciones en dicho mercado. (Véase J. S. Wang y A. F. Friedlaender, 1985, "Truck Technology and Efficient Market Structure", *Review of Economics and Statistics*, 67, pp. 250-258).

Por otra parte, en el sector del software también se verifica la existencia de economías de alcance, que actúan conjuntamente con las economías de escala y favorecen la formación de empresas grandes, que ofrecen varios productos distintos, pero relacionados, y de esta forma pueden competir de manera más eficiente.

Por tanto, la estrategia que eligen los productores está basada en la diferenciación de sus productos y aplicaciones con respecto a los de sus competidores. De esta forma, se evitan los desastrosos resultados de la competencia perfecta, ya que ésta requiere la homogeneidad de los productos presentes en el mercado.

Los costes de desarrollo de software son elevados y, además, *hundidos*. Con este término se hace referencia al coste en que se incurre con independencia del éxito o fracaso del proyecto productivo, lo cual es especialmente relevante en los casos en que un software concreto se diseña para un fin específico.

Los ingresos derivados de un producto de software pueden ser muy elevados si el producto es un éxito y el mercado lo acepta y adopta, o prácticamente nulos si el software no es del gusto de los usuarios por cualquier razón (no responde a sus necesidades y demandas, porque es de difícil utilización o poco *amigable*, etc.) y lo rechazan. En cualquiera de los dos casos extremos, los costes incurridos para la creación son prácticamente idénticos, por lo que las grandes inversiones en materia de software se realizan sin una idea clara de su éxito futuro. Puede calificarse, por tanto, como una actividad con grandes niveles de riesgo.

Para mitigar el riesgo, las empresas utilizan estrategias de diversificación, invirtiendo en múltiples productos con ciclos de vida solapados, lo que permite que los *ganadores* compensen las pérdidas de los que no tienen éxito. Además, una vez que un producto tiene éxito, los ingresos y costes derivados de las actualizaciones son mucho más predecibles, lo que contribuye a reducir el riesgo conjunto de la actividad.

De nuevo encontramos argumentos que explican el por qué de la existencia de em-

presas grandes en el sector del software, ya que son las que de manera más fácil pueden adoptar la estrategia descrita.

Otra estrategia para mitigar el riesgo, muy frecuente en desarrolladores de software libre, está basada en la imitación de productos comerciales de éxito. De hecho, es tan frecuente que cuestiona uno de los argumentos más utilizados para defender un apoyo de las autoridades al software libre: que este tipo de actividad se caracteriza por unas dosis elevadas de innovación. La realidad apunta más bien hacia lo contrario.

## El reto de la interoperabilidad

Uno de los grandes retos que el sector del software tiene que afrontar, desde el punto de vista de la oferta, es la interoperabilidad, referida a la habilidad de dos o más redes, sistemas, dispositivos, aplicaciones o com-

ponentes para intercambiar información y utilizar los datos intercambiados.

Debido a su naturaleza, gran parte del análisis que rodea la interoperabilidad es necesariamente técnico. Sin embargo, el logro de la interoperabilidad deberá medirse en última instancia por la experiencia del usuario. Lo relevante para el productor es que los productos y servicios implantados proporcionen el grado de interoperabilidad que demanda el usuario.

La interoperabilidad se manifiesta en la satisfacción del usuario: fundamentalmente, una mayor disponibilidad de información y servicios, así como la ausencia de frustración y dificultades en los intentos de llevar a cabo la operación que desea. Una solución interoperable satisface la expectativa de que funcionará sin la intervención del usuario a la hora de facilitar el intercambio de información entre diferentes platafor-

### Microsoft y la interoperabilidad

Es común considerar, en el discurso desinformado sobre interoperabilidad, que una de las empresas que más se opone a ella es Microsoft. En realidad, esta empresa está desarrollando importantes iniciativas facilitadoras.

Así, ha impulsado el *Interoperability Executive Customer Council*, un foro en el que importantes CIO's de entidades de todo el mundo, tanto públicas como privadas, trabajan conjuntamente con Microsoft para transmitir necesidades y requerimientos en materia de interoperabilidad. Se trata de un ejemplo más de la relevancia que para el futuro del negocio del software tiene prestar atención a las demandas de los usuarios; factor que los directivos de esta empresa de software parecen haber asimilado.

En paralelo a esta iniciativa con los clientes, Microsoft también ha sido fundadora de la *Interoperability Vendor Alliance*, cuyo objetivo es trabajar en materia de interoperabilidad con otros proveedores de software, como Novell y JBoss.

En noviembre de 2006, Microsoft y Novell llegaron a un acuerdo de colaboración para producir, comercializar y dar soporte a un conjunto de nuevas soluciones que permitan a los productos de ambas empresas funcionar mejor conjuntamente. El acuerdo se adoptó hasta —al menos— el año 2012. Con este nuevo modelo, los clientes de ambas empresas podrán disfrutar de mayores niveles de interoperabilidad entre Windows y Linux. Las dos manifestaron también que los acuerdos alcanzados en materia de negocio y patentes permitirán ofrecer el mayor nivel de interoperabilidad posible, con la seguridad de que ambas empresas respaldan las soluciones ofrecidas a los clientes.

Otra actuación de Microsoft favorecedora de la interoperabilidad ha respondido al deseo manifestado por sus clientes, especialmente los gubernamentales, de que se adopte un formato XML abierto para los productos de Office. El nuevo formato XML abierto reporta ventajas respecto a la situación previa en materia de robustez, eficiencia, seguridad y, por supuesto, interoperabilidad, ya que todo el proceso de intercambio de datos entre aplicaciones de Microsoft Office y los sistemas corporativos empresariales se simplifica enormemente. Al no necesitar acceso a aplicaciones Office, las soluciones pueden alterar la información contenida dentro de un documento de Office, o incluso crearlo desde cero utilizando las tecnologías y herramientas estándar para manipular XML.

mas, redes de computación, aplicaciones, dispositivos y otras entidades de sistemas. La interoperabilidad se logra si dos (o más) redes, sistemas, dispositivos o componentes pueden intercambiar información y utilizarla sin que pierda ninguna de sus capacidades básicas.

Las variaciones en la experiencia del usuario pueden surgir del servicio o dispositivo elegido en particular. Para evaluar la interoperabilidad, es importante ver la capacidad del usuario para recibir, acceder, almacenar, modificar, ampliar, mostrar y transmitir información utilizando múltiples dispositivos y servicios. Es importante decir que la interoperabilidad no es sólo una construcción teórica: el objetivo es que el usuario final vea cumplidos sus criterios de exigencia relacionados con la calidad del rendimiento.

La información es el asunto objeto de intercambio en la definición de interoperabilidad, y ser de cualquier clase, siempre que pueda transferirse electrónicamente en forma digital sobre redes de comunicaciones: voz, imágenes, documentos, contenidos de entretenimiento, comunicaciones tipo radiodifusión, credenciales de seguridad, *cookies*, formularios e interfaces de usuario, código de software ejecutable, etc.

La necesidad de interoperabilidad aumenta a medida que la capacidad de servicio de los pequeños dispositivos y la disponibilidad de los sistemas distribuidos están superando las restricciones tradicionalmente aceptadas. Esta mayor expectativa, parcialmente generada por las brillantes proyecciones del sector tecnológico, hace que los usuarios se encuentren receptivos, especialmente cuando comprueban que la falta de interoperabilidad acarrea aspectos negativos. Por lo tanto, lograr la interoperabilidad, allí donde es necesaria, ha adquirido mayor urgencia.

## 2.3. La demanda de software

Tradicionalmente, al abordar el estudio del software desde una perspectiva económica, la atención ha tendido a centrarse en el lado de la oferta: problemas de economías de escala, elevados costes fijos, riesgo de las inversiones, etc. Sin embargo, el estudio de la demanda en este mercado es de gran importancia, ya que provee de información clave sobre el valor que los consumidores asignan a los productos de software –una cuestión sobre la que los efectos de red son fundamentales– y, además, es de gran utilidad para entender la forma de establecer los precios.

Dos cuestiones importantes desde el punto de vista de la demanda de software son, en primer lugar, la importancia de los efectos de red en la valoración que los usuarios de software realizan de estos productos y, en segundo, los *costes de ajuste* que experimenta un usuario cuando se plantea un cambio importante en los productos de software que emplea.

### Tamaño de mercado y efectos de red

En general, la teoría económica define una externalidad como una situación en la que las acciones de un agente afectan a otros (positiva o negativamente), sin que se produzca una compensación monetaria para la parte perjudicada. Existen multitud de ejemplos de externalidades, en múltiples contextos.

Uno de los casos más citados de externalidades positivas es la educación y formación, ya que no sólo beneficia a la persona que la recibe (habitualmente la que paga

por ella), sino que su entorno personal y profesional también disfruta de sus efectos positivos. Por el contrario, y como ejemplo habitual de externalidad negativa, nos encontramos con la contaminación medioambiental, ya que el perjuicio de la actividad contaminante de una actividad industrial, por ejemplo, no sólo lo padece la empresa que genera la contaminación, sino todo su entorno.

### a. Ausencia de red:

Los elementos son independientes



En el caso de ausencia de efectos de red, no existe ninguna interconexión entre usuarios de un determinado producto, por lo que todos los consumidores le asignarían un valor de mercado que derivaría exclusivamente de las características intrínsecas del mismo, con independencia de si el producto es utilizado por un gran número de consumidores o, por el contrario, su empleo se reduce a círculos muy particulares y poco numerosos.

Esta situación, en la que los diferentes usuarios son independientes entre sí, es cada vez menos frecuente en el mercado de productos de software. En su ámbito concreto, cada vez con mayor frecuencia el valor de un

determinado producto depende en muchas ocasiones no sólo de las cualidades propias del mismo, sino que resulta un elemento decisivo el número de usuarios que lo utiliza o recurre a otro compatible. Cuanto mayor es el número de usuarios, es decir cuanto mayor es el tamaño del mercado para ese producto concreto, mayor es el valor real y percibido del software. Este efecto es conocido como efecto de red o externalidad de red.

En principio, el valor de un producto de software derivará de su utilidad intrínseca; es decir, qué hace, cómo de bien lo hace y qué efecto positivo genera su uso. Pero, además, existe un claro efecto de red: el tamaño del parque de usuarios de un software determinado incrementa las oportunidades de uno concreto de utilizarlo, lo que aumenta su propio bienestar.

### b. Efecto directo de red:

Elementos directamente relacionados



El caso descrito se conoce como *efecto directo de red*: el valor que un individuo asigna al producto depende directamente del tamaño de la red, esto es, del número de usuarios que, junto con él, emplean dicho producto. Los ejemplos de efectos directos de red son muy frecuentes en la vida moderna: por

ejemplo, los que derivan del uso de la telefonía, ya que cuanto mayor sea el número de personas con teléfono, mayor es la utilidad de poseer uno; de forma similar, existen los mismos efectos en la utilización de un sistema de correo electrónico o de uno de mensajería instantánea.

### c. Efecto indirecto de red:

Dependencia mutua de un elemento complementario



Frente a los efectos directos de red, también existen los denominados *efectos indirectos de red*, en los que el valor adicional del producto deriva de la dependencia colectiva de algún tipo de bien complementario, como la información disponible o la existencia de aplicaciones complementarias al producto. En este caso, el consumidor no se beneficia directamente de un mayor tamaño de la red (no se trata de que pueda interactuar con mayor número de personas), sino de que la mayor dimensión de la red propicia mayores inversiones en productos complementarios, y esta mayor inversión genera valor para el usuario de forma indirecta.

Un ejemplo de estos efectos indirectos de red está en Internet ya que, cuanto mayor sea el número de usuarios, mayor es la cantidad de contenido que los proveedores es-

tarán dispuestos a proporcionar y, en última instancia, mayor será el valor que obtenga cada usuario concreto.

En ocasiones, se producen combinaciones de ambos tipos de efectos, directo e indirecto. Piénsese, por ejemplo, en el caso de un producto de software concreto, como un procesador de textos. El producto tiene un valor intrínseco para un usuario aislado, ya que su utilización le proporciona una utilidad muy clara: crear, modificar, almacenar y conservar escritos. Al mismo tiempo, surgen efectos directos de red: cuanto mayor sea el número de usuarios del mismo procesador de textos, o de otros que sean compatibles, cada uno de ellos verá incrementada la utilidad del procesador ya que podrá intercambiar sus documentos con el resto, lo que le crea oportunidades de incrementar el valor derivado del producto. Existen, además, efectos indirectos de red: al crecer el número de usuarios, el lado de la oferta reacciona y surge una serie de productos adicionales, como pueden ser manuales orientados a usuarios que permitan aprender a utilizar el procesador de manera más eficaz, o nuevas aplicaciones compatibles que incrementen sus funcionalidades.

Al igual que para un procesador de textos, los efectos directo e indirecto pueden aplicarse a gran variedad de productos de software, como los sistemas operativos. En la actualidad, los usuarios de Linux están aprovechando estos efectos de red, ya que cuanto mayor es el número de usuarios, mayor es el número de proveedores de aplicaciones que desarrollarán versiones compatibles con este sistema operativo, lo que alimentará, a su vez, la llegada de nuevos usuarios. Pero este fenómeno no es propio, ni mucho menos, del software libre. Los usuarios de Windows también experimentaron dicho efecto en el momento en que el sistema no estaba tan extendido como en la actualidad.

## Características especiales de los mercados con economías de red

Los mercados en los que existen economías de red presentan características peculiares (Economides, 2006), reflejadas en la tabla 2.2.

**Tabla 2.2. Características de los mercados con economías de red**

1	El proveedor puede poner un precio en cualquiera de los extremos de la red o en ambos
2	Existencia de externalidades asociadas a los nuevos entrantes en la red
3	La senda de expansión del mercado es más rápida que en ausencia de red
4	La competencia perfecta es una estructura de mercado ineficiente
5	Aparición de guerras de estándares
6	Son mercados con grandes desigualdades en términos de cuotas de mercado y de beneficios
7	Una estructura monopolística puede maximizar el beneficio social
8	Elevados beneficios de una empresa no implican actuaciones anticompetitivas
9	Libertad de entrada en el mercado no implica competencia perfecta en el mismo
10	La intervención de las autoridades antimonopolio puede ser inútil
11	Se compite por el mercado, en vez de en el mercado
12	La dependencia del pasado adquiere una importancia esencial

Fuente: Economides, N. (2006)

- En primer lugar el proveedor tiene la capacidad de poner un precio en cualquiera o ambos extremos de la red. Un operador de telecomunicaciones podría fijar un precio por las llamadas que realiza un cliente, por las que recibe o por ambas. En el mercado de software, los proveedores también eligen el extremo de la red que desean cobrar. Un ejemplo claro es Adobe, que distribuye el programa *Acrobat Reader* de forma gratuita, mientras vende *Acrobat Distiller*, el programa que permite la creación de archivos que pueden ser leídos con el primero.
- La segunda característica de estos mercados radica en que con frecuencia el nuevo cliente que entra a formar parte de la red no es *recompensado* por el beneficio que produce en los demás como consecuencia de su entrada. Es el fenómeno típico conocido como externalidad, abordado en la sección 4.1. Un mecanismo que las empresas tienen para compensar este efecto es la discriminación de precios, tratado en la sección 2.4: el proveedor de software podría ofrecer condiciones favorables a usuarios importantes para maximizar la contribución que realizan en el mercado a través del efecto de red.
- En tercer lugar, los mercados con efectos de red suelen caracterizarse por una senda de penetración (expansión de la red) mucho más rápida que en sectores en los que no existen estos efectos.
- En presencia de externalidades de red, la competencia perfecta es una estructura de mercado ineficiente. La teoría económica sostiene que bajo competencia perfecta el tamaño de la red que se alcanza es inferior al socialmente óptimo, de lo que derivan, al menos, dos resultados importantes: una estructura de mercado distinta de la de competencia perfecta puede ser socialmente eficiente, y, por otra parte, en determinados casos podría justificarse una subsidiación pública en los primeros momentos, para potenciar el efecto de red, como sucedió con el nacimiento de Internet.
- La quinta característica es la aparición, en ocasiones, de *guerras de estándares* en los mercados con economías de red. La compatibilidad es la característica que permite hacer realidad la complementariedad entre productos. Algunos productos, por sus características intrínsecas, son directamente combinables. Pero, en otros muchos casos, la complementariedad real sólo puede lograrse a través de la aceptación de unos estándares técnicos específicos. Los creadores de estos productos tienen en su mano hacerlos parcial o totalmente incompatibles con el resto, mediante la creación de diseños propietarios o por el rechazo directo de la interconexión. De hecho, no siempre responde al interés de una empresa que sus productos sean plenamente compatibles con los de sus competidores, y el grado de compatibilidad constituirá una decisión estratégica de primer orden. Economides (2006) demuestra que cuanto más intensa sea la externalidad de red mayor es el incentivo de una empresa para ser incompatible con respecto a los productos sustitutivos.

## Características especiales de los mercados con economías de red (continuación)

- En sexto lugar, los mercados con fuertes efectos de red, en los que las empresas eligen sus propios estándares son mercados con grandes desigualdades en términos de cuotas de mercado y beneficios, ya que una empresa con una elevada cuota de mercado puede ofrecer mayor variedad de productos complementarios y, por tanto, su producto es más valorado por los consumidores, lo que repercute en mayores ventas. Un ejemplo lo tenemos en el mercado de sistemas operativos para PC's.
- Un resultado llamativo, también presentado por Economides (2006), es que en este tipo de mercados una estructura monopolística puede maximizar el beneficio social, debido a que cuando los efectos de red son muy fuertes una cuota de mercado muy elevada para una plataforma concreta implica importantes beneficios de red asociados a dicha plataforma, de los que se benefician tanto consumidores como productores. De hecho, una situación en la que un monopolio se escinde en dos empresas con sistemas incompatibles reduciría, en vez de incrementar, el beneficio social, ya que las externalidades de red serían mucho menos importantes. Esto se debe a que una estandarización *de facto* es valiosa, aunque la lleve a cabo un monopolista.
- Otra característica importante de las economías de red es que, al ser la desigualdad algo natural en estos mercados, no debe presumirse la existencia de acciones anticompetitivas al observar cuotas desiguales de mercado o elevados beneficios de una empresa. Son, como se ha visto, consecuencia del éxito social de una red creada en torno a un producto concreto.
- En noveno lugar, la existencia de efectos de red en un mercado implica que la libertad de entrada en el mismo no tiene por qué implicar una estructura de competencia perfecta. La eliminación de las barreras de entrada no suele afectar significativamente a la estructura de mercado, pese a que se estimula la competencia.
- Como consecuencia, la intervención de las autoridades antimonopolio puede ser inútil. Puesto que la estructura natural de un mercado con fuertes efectos de red es aquella en la que existen grandes diferencias en cuotas de mercado, la intervención podría llegar a ser contraproducente.
- Esta estructura no implica que no exista competencia en los mercados con fuertes efectos de red. Lo que sucede es que la competencia es de naturaleza distinta: se compite *por el mercado*, en vez de *en el mercado*. En un contexto en el que la empresa que crea el producto o la plataforma predominante obtiene gran parte del beneficio suele llevar a una competencia intensa para convertirse en la empresa dominante. Este es el tipo de estrategias que se han observado en torno a empresas como eBay, Amazon, Yahoo o Google.
- Por último, la presencia de efectos de red concede una importancia especial a la *dependencia del pasado*, en el sentido de que un sistema o red depende crucialmente de las decisiones pasadas de consumidores y productores. Por ejemplo, el precio al que en un momento del tiempo se podía vender un reproductor de VHS dependía del número de reproductores vendidos con anterioridad. La existencia de una base de consumidores ya consolidada favorece al incumbente en estos mercados, frente al que los competidores han de ofrecer productos más atractivos o a mejor precio para compensar dicha dependencia.

## Lock-in

Los consumidores de productos de software experimentan con frecuencia elevados *costes de ajuste (switching costs)* cuando reemplazan un producto por otro alternativo, lo que suele constituir un impedimento para los proveedores que compiten para atraer nuevos clientes.

Una aplicación completa de software puede estar compuesta y depender simultáneamente

de un número determinado de elementos complementarios, incluyendo diferentes componentes, equipo de hardware y software. Existen, además, inversiones complementarias menos tangibles, como el entrenamiento y la formación de los trabajadores necesarios para el uso de la aplicación.

En caso de que el consumidor se plantease cambiar de proveedor de aplicaciones tendría que hacer frente a los costes aso-

ciados al cambio, que pueden extenderse al hardware, si necesita otro de distintas características, la nueva infraestructura de software, la formación del personal y la sustitución de herramientas de gestión o administrativas.

Pero los costes al adquirir un nuevo producto de software o cambiar el que se utilizaba, son de dos tipos:

- Por una parte, los directos asociados con el producto (equipo, licencias de software, entrenamiento, etc.), que son sobre los que el nuevo proveedor puede influir con distintas estrategias de precios.
- Además, existen otros adicionales, que se denominan *costes de ajuste*, sobre los que el nuevo proveedor no puede influir y que, hasta cierto punto, podrían ser manipulados por un proveedor existente. Sería el caso, por ejemplo, de un proveedor que incrementase los *costes de ajuste* al no facilitar apoyo para la portabilidad de sus aplicaciones de software, de forma que creara una barrera que dificulta la transición a otra plataforma

alternativa. En un caso extremo, un cliente podría quedar *atrapado*, lo que se identifica con el término *lock in*. Por otra parte, el contrapunto para el proveedor es que el valor de su plataforma, al establecer esas barreras, sería menor para un nuevo cliente que se plantee adoptarla.

Un proveedor alternativo puede enfrentarse al *lock in* subvencionando los costes de ajuste que soporta el cliente o, alternativamente, haciendo que su producto sea compatible con el software del proveedor original, ya sea de forma unidireccional o bidireccional. En ambos casos, los costes del nuevo proveedor se verían incrementados.

## 2.4. El precio

El precio es un elemento fundamental en la relación entre clientes y proveedores de cualquier bien o servicio, por lo que la teoría económica ha dedicado una gran atención a los mecanismos de determinación de precios de los distintos tipos de bienes

**Tabla 2.3. Ejemplos de fijación de precios para aplicaciones de software**

Tipo	Descripción
Licencia individual	Se trata de un precio fijo por el cual un individuo adquiere el derecho a instalar y utilizar una aplicación. Es un procedimiento típico en aplicaciones orientada a particulares (no empresas).
Site license	A cambio de un precio fijo, se obtiene el derecho a la instalación y uso de la aplicación sin límites dentro de una organización. El precio está basado en el tamaño de la organización y surge de un proceso de negociación.
Seat or host license	A cambio de un precio fijo se obtiene el derecho a la instalación y uso ilimitado de la aplicación en un ordenador específico, sin importar quién es el usuario del mismo. Suele ser posible negociar descuentos en función del número de <i>seats</i> .
Floating license	A cambio de un precio fijo se obtiene el derecho a que un determinado número máximo de usuarios puedan utilizar una aplicación simultáneamente. Normalmente se permite la instalación de la aplicación en un gran número de ordenadores y a través de una licencia de servidor se monitoriza y limita el número simultáneo de usuarios.
Suscripción	Similar a las tres opciones anteriores, excepto en que se realizan pagos periódicos y fijos tan pronto como la aplicación se empieza a utilizar y se incluyen las actualizaciones de la aplicación.
Licencia basada en uso o transacción	Similar a la anterior, salvo en el hecho de que los pagos están basados en mediciones del uso de la aplicación (número de horas o de transacciones realizadas, por ejemplo).

Fuente: Messerschmitt, D. G. y C. Szyperki (2003)

y estructuras de mercado. En el caso del software, la práctica habitual es la venta de una licencia de uso.

La existencia de importantes economías de escala en la producción y oferta de software —analizada en la sección 2.2— convierte la fijación de precios en una cuestión de gran interés para los proveedores, ya que los costes unitarios, un elemento habitualmente utilizado como referencia para fijar los precios, no ofrecen una información muy útil, al descender muy rápidamente con el tamaño de mercado. Ésta y otras consideraciones son las que han provocado el desarrollo de una amplia variedad de estrategias de fijación de precios en este mercado.

## **Distribución de los ingresos a lo largo del tiempo**

La creación de software no implica únicamente el desarrollo inicial del mismo, sino también el mantenimiento (reparación de errores o defectos), las actualizaciones (que proporcionan capacidades progresivamente mayores o afrontan nuevas necesidades) y el apoyo técnico al cliente. Cuando se interpreta la venta de software como la venta de un servicio deben contemplarse en el precio este tipo de costes operativos.

En ese sentido, una cuestión fundamental es cómo repartir a lo largo del ciclo de vida del software el pago del cliente, de modo que se recuperen los costes y se obtenga un beneficio. Es decir, ¿es preferible hacer soportar al cliente una gran parte del coste del software en un momento inicial o, por el contrario, ofrecer el producto inicial a bajo precio e ir recuperando los costes en momentos posteriores de su ciclo de vida?

Cada estrategia tiene ventajas e inconvenientes. Una repercusión temprana al cliente de los costes del software tiene la ven-

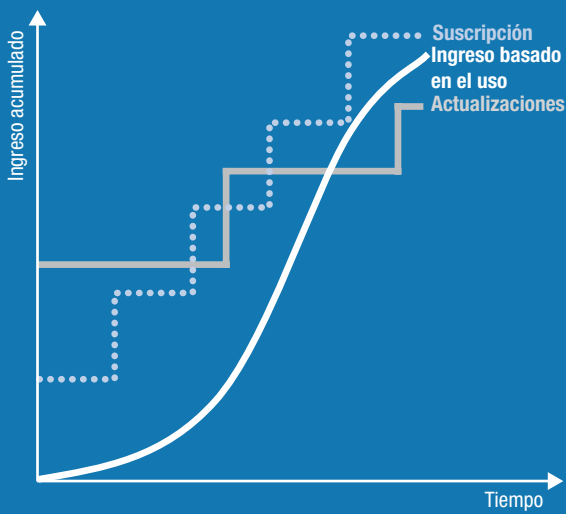
taja de eliminar incertidumbres en el proveedor, ya que le permite recuperar más rápidamente la inversión realizada. Pero esta estrategia tiene el riesgo potencial de hacer más lenta la extensión del producto, dificultando el crecimiento de la base de clientes y el aprovechamiento de las economías de escala. Por otra parte, la ventaja de un aplazamiento en la repercusión a los clientes del coste del software es hacerlo más atractivo, al resultar menos costosa su adquisición inicial.

La estrategia de cobro basada en actualizaciones, por otra parte, plantea el problema potencial de hasta cuándo actualizar un producto y cuándo cambiar a uno nuevo. En ocasiones, un productor puede ver duplicado su esfuerzo de actualización cuando existen varias versiones de un producto similar (por ejemplo, Windows 95 y Windows NT antes de la introducción de Windows XP).

Cada producto de software, en función de sus características intrínsecas y del cliente objetivo al que va dirigido, exigirá una estrategia distinta. Cada una de las opciones analizadas en la tabla 2.3 se adapta mejor a una u otra estrategia extrema.

El gráfico 2.5 refleja el proceso de acumulación de ingresos a lo largo del tiempo, de acuerdo con tres estrategias alternativas de fijación de precios. Lo habitual para los cuatro primeros tipos de licencias contemplados en la tabla 2.3 es cobrar a los clientes por las actualizaciones de los programas. Las nuevas versiones de los programas ofrecen características, calidad y rendimientos mejorados y suelen venderse a precios menores que los que se fijan a los que compran el producto por primera vez. De esta forma, se está reconociendo la utilidad que tiene la base de anteriores versiones, así como que la nueva versión del programa puede ofrecer un valor incremental menor que el software inicial.

## 2.5. Distribución temporal de los ingresos según opciones alternativas de fijación de precios



Fuente: Messerschmitt, D. G. y C. Szyperki (2003)

En la estrategia de suscripción, el cliente realiza pagos regulares, que van incrementando los ingresos del proveedor de una forma más equilibrada en el tiempo. Para algunos clientes, este tipo de estrategia puede suponer una ventaja, ya que les permite conocer con anticipación los costes, con lo que su presupuesto es más fácil. La estrategia de suscripción es especialmente adecuada cuando además de proporcionar el software se incluye la prestación de servicios operativos.

La opción basada en el uso puede resultar atractiva para los clientes que consideren que el valor del software está directamente relacionado con la intensidad con que se utiliza, así como para los proveedores que confían en que su producto va a tener un alto grado de uso. Una desventaja para el proveedor radica en que los ingresos se obtienen de forma más lenta en el tiempo y, en un caso extremo, pueden no llegar a materializarse.

## Estrategias de discriminación de precios

Un tema importante en materia de precios es si se puede tratar de modo diferente (precios distintos) a distintos tipos de clientes. Se considera que no existe discriminación de precios cuando los repercutidos a los clientes están basados exclusivamente en los costes marginales del proveedor. Es decir, que la diferencia entre precio y coste marginal es la misma para cualquier cliente. Esta diferencia, fija para todos los clientes, es con la que el proveedor espera recuperar los elevados costes fijos de su inversión.

Desde un punto de vista teórico, en mercados muy competitivos puede forzarse una estructura de precios no discriminatoria y

**Tabla 2.4. Estrategias de discriminación de precios**

Tipo	Descripción
De primer grado	Consiste en cobrar a cada cliente su precio de reserva (el precio máximo que está dispuesto a pagar por el producto). El problema fundamental de esta estrategia es informacional: la enorme cantidad de información que se necesita de los clientes.
De segundo grado	Práctica basada en el cobro de precios unitarios distintos por cantidades diferentes de un mismo bien o servicio. Es el caso de los descuentos efectuados por comprar grandes cantidades de un producto.
De tercer grado	Dividir a los consumidores en varios grupos con demandas diferenciadas y cobrarles un precio diferente a cada uno de ellos.
Versionado	Crear un conjunto de variantes del producto que ofrece diferentes niveles de prestaciones y dejar que los consumidores determinen voluntariamente su valoración del producto a través de su elección.
Intertemporal	Separar a los consumidores en grupos con diferentes demandas y cobrarles distintos precios en cada momento.
Según intensidad de uso	Estrategia basada en el cobro de precios más elevados durante los períodos punta, en los cuales la limitación de capacidad hace que los costes marginales sean más elevados.

Fuente: enter

unos márgenes sobre el coste marginal muy reducidos. Este sistema, sin embargo, dificultaría la recuperación de los costes fijos, especialmente los asociados a la creación del producto y sus actualizaciones.

Existen, en el caso del software, costes sustanciales que no quedan reflejados en una estructura de precios no discriminatoria y deben, por tanto, incorporarse al margen comprendido entre precio y coste marginal. Para lograrlo, la estrategia necesaria es la minimización de la competencia directa mediante la diferenciación de los productos y la explotación de mecanismos de mercado como los efectos de red.

Pero existe otra alternativa para recuperar los elevados costes fijos derivados de la creación y mantenimiento de los productos de software: la discriminación de precios, que es un sistema de fijación en el que los precios reflejan la disposición al pago de cada grupo de consumidores, en vez de ser una cuantía fija por encima de los costes. Es decir, para un mismo producto se fijarán distintos precios en función de cuál sea la predisposición al pago por parte de un consumidor.

Existen distintas formas teóricas de discriminar precios (tabla 2.4). Pueden abarcar, desde una estrategia de precio totalmente individualizada, en la que cada consumidor paga en función del valor que le proporciona el software (discriminación de primer grado), o una segmentación en grupos de clientes homogéneos (por ejemplo, particulares, pequeñas empresas, grandes empresas, fundaciones, etc.), de los que se supone que extraen un valor similar al resto de componentes del mismo grupo (discriminación de tercer grado), hasta un sistema de versionado, donde cada cliente elige la versión de un producto (con más o menos aplicaciones y utilidades) más atractiva para sus intereses y a su disposición a pagar.

Este último tipo de estrategia, dada la voluntariedad de la elección, es muy popular en numerosos sectores (vehículos, etc.) En el caso del software, en ocasiones una variante básica del producto se ofrece a un precio reducido, o incluso gratuito, aunque quizás con una duración determinada, mientras que las variantes más sofisticadas se venden a precios superiores.

El sistema de versionado es particularmente apropiado en el mercado del software, debido a que el coste marginal del desarrollo de versiones no es muy elevado: es cuestión de desarrollar el producto de mayor valor y, a partir del mismo, inhabilitar determinadas prestaciones para tener, así, el producto básico. Todo esto, desde luego, debe ser previsto en una fase temprana del diseño del producto. Desde el punto de vista del diseño, la *modularidad* es de gran utilidad, puesto que garantiza que se puedan añadir o quitar módulos al producto de software y, de esta forma, se dispone de una herramienta sencilla para generar distintas versiones. Como consecuencia, también habrá de definirse en una fase temprana del proceso de diseño la estrategia de precios que se va a aplicar.

## **Empaquetamiento y venta conjunta**

El empaquetamiento y venta conjunta de productos es otra estrategia de precios complementaria de la discriminación. En el caso del software, las posibilidades son múltiples: desde el empaquetamiento de productos dentro de una *suite* ofimática (procesador de textos, hoja de cálculo, etc.), hasta el empaquetamiento con versiones futuras, soporte al cliente, etc., pasando por la opción muy extendida de integración de hardware y sistema operativo al adquirir un ordenador.

La ventaja del empaquetamiento es que permite hacer frente a la dispersión de la de-

## Empaquetamiento y aumento de los ingresos

Gracias al empaquetamiento de productos es posible obtener un mayor volumen de ingresos que el que se obtendría vendiéndolos por separado en un contexto de demandas dispersas. Un ejemplo:

Un productor de dos productos (A y B) de software dispone de un mercado de 100 clientes potenciales. La mitad de ellos está dispuesta a pagar un precio máximo de 50 euros por A y 150 euros por B. La otra mitad pagaría como máximo 100 euros por A y 50 por B.

Si los productos A y B se vendiesen por separado, los precios que hacen máximo el ingreso del productor son: 50 euros para A (obtiene un ingreso de 5.000 euros, porque los 100 clientes lo comprarán a dicho precio) y 150 euros para B (el ingreso será de 7.500 euros, ya que sólo la primera mitad de los individuos pagará por ella). El ingreso total será de 12.500 euros.

Frente a este planteamiento, si el productor decide vender conjuntamente los productos A y B a un único precio, la primera mitad de clientes potenciales estaría dispuesta a pagar como máximo 200 euros por el paquete (A + B) y la otra mitad sólo 150 euros. En este caso, si el proveedor fija un precio de 150 euros por el paquete obtendrá unos ingresos de 15.000 euros, superiores a los que obtenía con la venta separada de los productos.

El secreto de la estrategia radica en que, al empaquetar los productos, se consigue reducir la dispersión en las predisposiciones al pago de los clientes: en el caso del paquete, el rango de predisposición es 150-200 euros, mientras que en productos individuales la mayor diferencia se producía en el producto B (50-150).

### La estrategia de empaquetamiento maximiza el ingreso

	Producto A	Producto B	Producto A+B
50 clientes tipo Z	50	150	200
50 clientes tipo Y	100	50	150
Precio óptimo	50	150	150
Máximo ingreso	5.000	7.500	15.000

manda de los clientes, que siempre supone un reto para el proveedor.

La estrategia de empaquetamiento no implica un elevado coste para los proveedores

de software, por lo que constituye una herramienta de fijación de precios siempre interesante para obtener el máximo volumen de ingresos y muy utilizada en determinadas aplicaciones (por ejemplo, las de ofimática)



# 3.

## Efectos micro y macroeconómicos del software

---

### 3.1. Software propietario comercial vs. OSS

**El análisis de los elementos del mercado de software** realizado en el capítulo anterior ofrece base suficiente para determinar las diferencias existentes entre el software propietario, de clara vocación comercial, y el software libre, que en muchos casos no tiene una orientación comercial clara, ya que detrás del mismo muchas veces subyacen intereses propios de los desarrolladores (ganancia de prestigio dentro de la comunidad de los ingenieros de software, etc.) en mayor medida que intereses de naturaleza mercantil.

El objetivo del presente apartado es sistematizar y acotar las diferencias entre ambas modalidades de software, para lo que resulta útil reflexionar sobre los aspectos contemplados en la tabla 3.1, donde se comparan las posiciones del software propietario y del software libre en varias dimensiones relevantes.

En primer lugar, los creadores de software propietario hace un uso casi exclusivo del mercado como herramienta de coordinación entre oferta y demanda de sus productos de software, mientras que una tendencia frecuente entre los creadores de software libre es excluir al mercado como mecanismo de asignación y provisión de

**Tabla 3.1. Comparación software comercial: software propietario vs. libre**

	Software propietario	OSS
<b>Coordinación oferta-demanda</b>	El mercado como herramienta de coordinación	Coordinación autónoma entre los ingenieros de software
<b>Orientación en el desarrollo</b>	Orientación hacia las necesidades del cliente y usuario	Orientación hacia los intereses del desarrollador
<b>Incentivos a la innovación</b>	Consecución de beneficios económicos para el innovador. Propiedad de las innovaciones	Intereses personales que no coinciden necesariamente con los de los usuarios
<b>Compatibilidad-interoperabilidad</b>	Fuertes incentivos a la compatibilidad	Riesgos de ramificaciones
<b>Corrección de fallos</b>	Fácil instalación, altos niveles de compatibilidad en diferentes entornos de hardware, pero procesos de corrección más dilatados en el tiempo	Rápida disponibilidad, no siempre es sencilla la compatibilidad ni la instalación
<b>Adaptación a las necesidades del cliente (<i>customizability</i>)</b>	Adaptación dentro de las posibilidades definidas por el marco	Grandes posibilidades de adaptación, pero para los usuarios experimentados
<b>Señalización</b>	Depende de la política de publicaciones del productor del software	Fuertes efectos de señalización, por la visualización de las contribuciones individuales

Fuente: *enter*, a partir de Koothis, S., M. Langenturth y N. Kalwey (2003)

información, por lo que la coordinación es mucho más difícil, ya que se realiza de forma autónoma entre los propios ingenieros.

Otro elemento distintivo es la orientación seguida en el desarrollo del software. En otras palabras, cuáles son los intereses que subyacen y prevalecen. En el caso del software propietario, dada su vocación decididamente comercial, se persigue ante todo satisfacer las necesidades de clientes y usuarios. Por el contrario, en el caso del software libre (especialmente aquél cuyo fin último no es la comercialización), los intereses personales del desarrollador son prioritarios, aunque vayan en detrimento de características como la facilidad de uso, fundamentales para el usuario final.

El motor de la innovación es en ambos casos muy distinto. Mientras en el software propietario es la obtención de un producto de éxito en el mercado con el que obtener un beneficio económico para el innovador, gracias a que puede apropiarse del mismo por la existencia de derechos de propiedad, en el caso del software libre pesan los intereses personales del desarrollador (reputa-

ción, etc.) y no tanto el logro de una rentabilidad económica asociada al producto.

Un aspecto en que vuelven a surgir diferencias entre software propietario y libre es la corrección de fallos en los programas. De acuerdo con los defensores del software libre, los procesos de corrección de fallos son más rápidos en este caso puesto que, al disponer de acceso al código fuente, los programadores pueden realizar correcciones directamente. Éstas, en el caso del software propietario, son más lentas puesto que, tras la detección de los fallos y su comunicación al propietario, éste debe acometer las modificaciones y difundirlas en el mercado. Sin embargo, las garantías de compatibilidad del software corregido en diferentes entornos de hardware son mayores en el caso del software propietario, puesto que el proceso está centralizado y existe riesgo comercial en caso de proveer una solución insatisfactoria.

Otro campo de diferenciación entre software propietario y libre es la posibilidad de adaptación a los requerimientos particulares de un cliente. Mientras con el software

propietario es más difícil, con el libre, dadas las posibilidades de modificación del código fuente, las posibilidades están abiertas. Pero a nadie se le escapa que la proporción de usuarios de software capaces de modificar los programas para adaptarlos a sus necesidades personales, y que por tanto pueden beneficiarse de esta flexibilidad del software libre, es marginal.

### Software libre, más basado en la imitación que en la innovación

El software libre aparecido hasta la fecha es, en términos generales, poco innovador. Salvo algunas excepciones (la familia de sistemas operativos BSD, la aplicación Sendmail, el lenguaje Perl o el sistema X Windows), la actividad fundamental de los creadores de software libre ha sido desarrollar programas que se limitan a replicar las funciones de otros —software propietario— que ya estaban en el mercado.

Algunos autores, como Lawrence Lessig, no son de la misma opinión, a pesar de la evidencia. Así, Lessig señala que el software libre es el *alma de Internet*, lo que no corresponde con la realidad, puesto que si bien es cierto que la construcción de Internet ha estado basada en protocolos libres, éstos son algo muy distinto al software libre: los protocolos libres pueden implementarse tanto en software libre como propietario. Además, no hay que olvidar que, en su origen, Internet no tenía fines comerciales (sólo desde 1991 se alteró esta situación), por lo que no tenía sentido que las empresas que trabajaron en su nacimiento desarrollasen aplicaciones de software comercial, puesto que no hubieran podido beneficiarse de ellas <sup>(a)</sup>.

La conclusión que se extrae al estudiar la realidad del mercado es que la gran mayoría del software comercial es en la actualidad *propietario*, que tiene su origen en ideas innovadoras, desarrolladas por sus creadores con el fin de obtener una rentabilidad económica de las mismas. Este proceso de innovación no se ha detenido y, como consecuencia, las principales aplicaciones de software propietario y comercial (procesadores de texto, hojas de cálculo, bases de datos, videojuegos, etc.) que se utilizan hoy día difieren enormemente de las empleadas años atrás.

La innovación ha sido, por tanto, una clave esencial del software comercial durante las últimas décadas. Frente a esto, la mayor parte del software libre se ha centrado en generar versiones gratuitas de productos comerciales que ya existían en el mercado.

<sup>(a)</sup> Padilla, et al. (2004)

Por último, un factor de diferenciación es la señalización de los programadores dentro de la comunidad de creadores de software, en función de sus contribuciones personales, que es superior en el ámbito del software libre. De hecho, en dicho caso, muchas veces es la única motivación para el desarrollo de software, ya que no existen incentivos económicos.

### Fortalezas del software propietario

Las principales fortalezas del software propietario aparecen reflejadas en la tabla 3.2. Una primera es de naturaleza estructural. La protección que los creadores de software propietario buscan para sus invenciones garantiza que, a través de la defensa de los derechos de propiedad, se creen fuertes incentivos a la investigación y la innovación. Por el contrario, en el modelo de software libre, todos los resultados están abiertos para su utilización por otras personas.

Por tanto, el modelo económico que subyace en el software propietario permite la apropiación de beneficios por parte del inversor, a través de la explotación comercial de su invención. Este requisito es imprescindible para garantizar que acometerá proyectos de inversión. La protección de los derechos de propiedad en el caso del software ya fue discutida en ENTER (2006), por lo que basta recordar que la posibilidad de no ofrecer ningún tipo de protección convierte en insostenibles, desde el punto de vista económico, las iniciativas de software no comercial.

Una segunda ventaja del software propietario se basa en que la guía de su diseño es la satisfacción de las necesidades de los clientes-usuarios; un elemento que en muchos desarrollos de software libre queda relegado por detrás de otras consideraciones, puesto que, en este caso, el objetivo último es el prestigio

**Tabla 3.2. Fortalezas del software propietario**

<b>Fuente de innovación</b>	Debido a la garantía de los derechos de propiedad, fuertes incentivos a la investigación. En el modelo OSS, por el contrario, los desarrollos están a disposición de todas las partes.
<b>Orientación al usuario o cliente</b>	La orientación al usuario es un requisito necesario para el éxito comercial. Por contraste, la orientación del OSS es hacia el desarrollador, lo que hace desaparecer mecanismos de mercado.
<b>Productos sencillos para el usuario</b>	Es una vía para lograr la mayor difusión del software comercial, al ampliar la base de clientes. Esta característica no es común en el OSS.
<b>Altos niveles de estandarización, compatibilidad y actualizaciones</b>	Fuertes intereses de los proveedores de software comercial para que sus productos sean altamente estandarizados. La compatibilidad se produce en diferentes entornos de hardware. Garantía de disponibilidad de drivers y fácil instalación, así como de actualizaciones regulares del software.
<b>Ausencia de fragmentaciones</b>	Se previenen las fragmentaciones y las versiones incompatibles al tener el proveedor del software comercial todos los derechos sobre el mismo.
<b>Amplia variedad en el software de aplicaciones</b>	Existe una amplísima gama de aplicaciones disponibles para plataformas propietarias, en las que está garantizado el buen funcionamiento de las aplicaciones en diferentes configuraciones de hardware.

Fuente: enter, a partir de Koothis, S., M. Langenturth y N. Kalwey (2003)

dentro de la comunidad de programadores y no la obtención de un beneficio comercial. Mientras que los mecanismos de mercado garantizan que el desarrollo del software propietario tiene en cuenta las necesidades y gustos de los usuarios finales, y que sus productos satisfacen las necesidades de los consumidores en términos de facilidad de uso y reducidos costes de cambio y aprendizaje, la ausencia de este mecanismo de control en el caso del software no comercial aleja el interés de los programadores del de los usuarios no especializados.

Ligada con esta cuestión se presenta la tercera ventaja: la prioridad asignada al diseño de productos de un sencillo manejo por parte del usuario-cliente final. Además de procurar la sencillez intrínseca del producto, éste suele acompañarse de documentación relacionada con el empleo del software. Estos factores resultan necesarios para facilitar la mayor difusión posible en el mercado.

La cuarta fortaleza que se detecta en el software propietario es el notable grado de estandarización de sus productos, lo que facilita la compatibilidad en diferentes entornos de hardware y garantiza la disponibilidad de *drivers* y actualizaciones regulares. En línea

con esta fortaleza, está una quinta: la ausencia de fragmentaciones, ya que el elevado nivel de estandarización va unido a que el proveedor del software es el propietario de los derechos sobre el mismo. En el entorno del software libre, en el que las mejoras, versiones y actualizaciones de programas se realizan de forma descentralizada, se corre el riesgo de que se generen versiones de programas no compatibles entre sí.

Por último, otra fortaleza detectada dentro del campo del software propietario es la enorme gama de aplicaciones disponibles en el mercado que son compatibles con plataformas propietarias. Estar sometido a las fuerzas del mercado, que castigan un mal producto expulsándolo del circuito comercial, tiene como consecuencia la garantía de que dichas aplicaciones funcionarán correctamente en diferentes configuraciones de hardware.

## Desventajas del software propietario

Pese a las indudables ventajas que presenta el software propietario sobre el libre, es posible señalar algunas desventajas, puestas de manifiesto por los defensores del software libre.

En primer lugar, la imposibilidad de que un usuario que detecte un error de programación pueda solucionarlo por sí mismo. Lo más que podrá hacer es informar del problema al fabricante y esperar que éste le facilite una solución. Frente a esta realidad, cabe el matiz de que el vendedor de software propietario tiene fuertes incentivos para solucionar el problema rápidamente y hacer extensiva la solución a todos los usuarios de la forma más sencilla posible. De no ser así, su reputación comercial se verá seriamente dañada.

Otra crítica al software propietario, procedente de la comunidad de ingenieros, es que sus productos son revisados por un número menor de individuos que el software libre. A pesar de ello, las revisiones a las que se someten los productos de software propietario son en general más ordenadas y estructuradas, ya que en el caso del software libre el estudio se realiza de forma descoordinada, puesto que cada programador presta atención a la parte que más le interesa.

Finalmente, se ha señalado como desventaja la imposibilidad de que un usuario con conocimientos suficientes pueda modificar el código del programa para adaptarlo a sus requerimientos, lo que sí puede hacer en el caso del software libre.

## Fortalezas del software libre

Los productos de software libre tienen, a su vez, determinadas ventajas sobre el prope-

tario comercial. La primera viene marcada por la posibilidad que tienen los usuarios avanzados de adaptar el software a sus necesidades, gracias a la posibilidad de acceder al código fuente. Debe mencionarse, no obstante, que sólo una fracción muy pequeña de usuarios de software en el mundo pueden beneficiarse de esta ventaja, ya que se requiere un elevado conocimiento de programación informática para poder lograr este objetivo con garantías.

Otra de las fortalezas del software libre frecuentemente señalada es la rápida expansión del conocimiento que subyace como consecuencia de la ausencia de derechos de propiedad y la ya mencionada posibilidad de acceder al código fuente.

Es innegable que disponer de acceso al código fuente permite una más rápida difusión del conocimiento dentro de la comunidad de especialistas. Sin embargo, también se ha señalado que, precisamente por la ausencia de derechos de propiedad, el software libre desemboca en desincentivos a la producción del conocimiento que se materializa en el software. En otras palabras, se reduce la producción de software porque el autor no puede apropiarse de los beneficios de su creación.

Otro de los puntos fuertes del software libre se basa en que permite más fácilmente la señalización pública de los desarrolladores de este tipo de software dentro de la comunidad de ingenieros informáticos. Es decir,

**Tabla 3.3. Fortalezas del OSS**

<b>Adaptación a las necesidades de los usuarios avanzados</b>	La adaptación es ilimitada en la medida en que el código fuente es abierto. Sin embargo, sólo se benefician los usuarios con el conocimiento suficiente para realizar modificaciones.
<b>Rápida expansión del conocimiento</b>	La ausencia de derechos de propiedad sobre el OSS y el acceso al código fuente permite que el conocimiento pueda expandirse más fácilmente dentro de la comunidad de ingenieros de software.
<b>Reputación</b>	El OSS permite más fácilmente la señalización pública de los desarrolladores.

Fuente: enter, a partir de Koeths, S., M. Langenfurth y N. Kalwey (2003)

el mecanismo reputacional en dicha profesión es más visible.

Finalmente, los defensores del software libre también señalan como una de sus ventajas la mayor protección que, según ellos, consiguen los usuarios. En su opinión, poder acceder al código fuente hace más difícil la introducción de código cuya finalidad sea *espíar* al usuario. Sin embargo, no está claro si esta ventaja realmente existe en la práctica o se trata de una posibilidad teórica puesto que, a falta de casos concretos de espionaje en el ámbito del software propietario, no tiene sentido prejuzgar que las empresas productoras vayan a delinquir.

### Desventajas del software libre

La primera desventaja, ya aludida, del software libre es el riesgo de fragmentación al que se enfrenta; es decir, la aparición de versiones de un mismo producto incompatibles entre sí como consecuencia de la multiplicidad de *aportaciones*. El resultado es que se pueden producir casos de una aplicación diseñada para funcionar en una versión de Linux que no pueda utilizarse en otra versión del mismo sistema operativo. Para paliar este problema se ha puesto en marcha la iniciativa *Linux Standard Base*.

La segunda desventaja deriva de las dificultades que tienen los programadores para obtener una compensación económica directa por el tiempo y esfuerzo dedicados a la creación, dado que el código fuente está disponible para cualquier persona interesada. La consecuencia lógica es la reducción en el esfuerzo dedicado a esta actividad por parte de sus creadores, lo que terminará limitando la innovación que se produzca en este terreno.

Ligado con lo anterior, muchas empresas tampoco disponen de incentivos sólidos para invertir en el desarrollo de proyectos

de software libre no comercial, dadas las dificultades para lograr una rentabilidad suficiente. Por el contrario, las empresas dedicadas a la creación de software comercial sí tienen estos incentivos, puesto que su preocupación por satisfacer las necesidades de los usuarios crea un potencial para la obtención de rentabilidad.

## 3.2. Aproximación microeconómica

*El coste de los servicios de software en una estrategia empresarial a largo plazo. Evidencia internacional*

Un error muy extendido entre la opinión pública es considerar que el software libre resulta, a priori, una opción menos costosa que el software propietario.

El error se produce por la realización de una comparación parcial, que sólo incorpora la información relativa al coste de las licencias, pero deja de lado el resto de elementos de coste vinculados a la adopción y utilización de software libre. Precisamente, estos elementos del coste pueden ser de tal magnitud que los de la licencia sólo sean una proporción muy limitada del coste total, lo que implica que prestarles atención preferente puede conducir a conclusiones equivocadas.

De hecho, cuando una organización se plantea la posible migración de un sistema de software a otro (por ejemplo, desde entorno Windows a Linux) debería considerar un escenario de costes a medio o largo plazo, lo que introduce en la función relevante dimensiones que van mucho más allá del coste de la licencia.

Así, debe tenerse en cuenta, por ejemplo, la eventual necesidad de modificar el hardware, los costes de formación del personal especializado (personal de TI, encargado de la puesta en marcha y el mantenimiento de los equipos en la organización) y de los trabajadores de la organización, el coste de las aplicaciones de software compatibles con el nuevo sistema operativo, etc.

La valoración económica que aborda el estudio de los costes de las alternativas con esta visión amplia se denomina *Total Cost of Ownership* (TCO). El horizonte temporal considerado ha de ser suficientemente largo (normalmente en torno a cinco años), de forma que puedan incorporarse todos los costes relevantes. Un plazo de tiempo dilatado es, además, un horizonte razonable para la adopción de decisiones en un contexto empresarial.

Numerosos estudios han señalado el error de ofrecer una respuesta cortoplacista basada en la mera consideración del coste de las licencias y realizado comparaciones en

materia de costes de adopción entre opciones de software libre y propietario.

Uno de los estudios más conocidos es el realizado por IDC en 2002, cuyo objeto era comparar, a lo largo de un período de cinco años, los TCO's de sistemas respectivamente equipados con Windows 2000 Server y Linux Server. Para ello, se obtuvieron datos de 104 empresas norteamericanas, seleccionadas aleatoriamente de una lista provista por *Network World*, una publicación de IDG.

El estudio realizó estimaciones desagregadas por las categorías de coste contempladas en la tabla 3.4. Se aprecia una interpretación amplia y comprensiva del concepto de coste, ya que los costes propios de las licencias de software son sólo un apunte dentro de una amplia especificación de posibles costes soportados por una organización empresarial en el terreno de los servidores.

Un primer resultado del estudio era el reducido peso de los costes de adquisición del software (licencias) dentro del conjunto total

**Tabla 3.4. Estudio IDC (2002). Windows 2000 frente a Linux.**

Categorías de coste contempladas

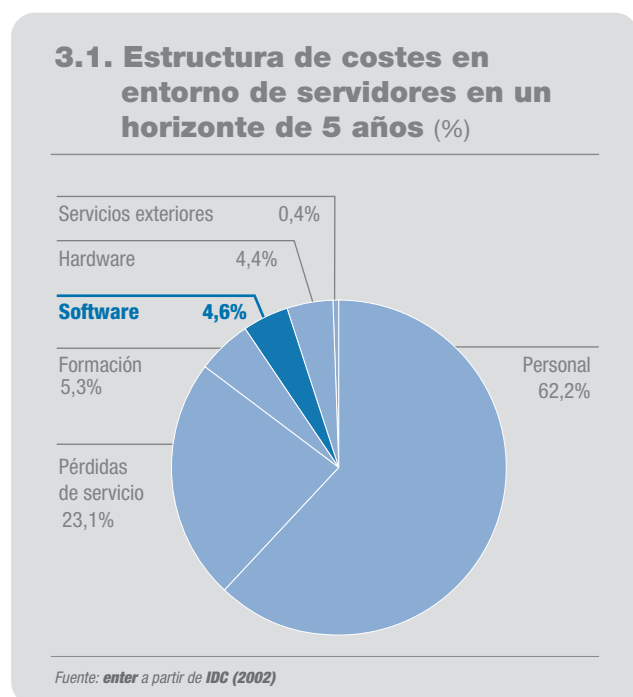
<b>Hardware</b>	Adquisición de nuevo hardware, instalación (puesta en marcha y actualizaciones) y mantenimiento.
<b>Software</b>	Distintas categorías: <ul style="list-style-type: none"> <li>• Sistema operativo. Adquisición, instalación (costes iniciales más los derivados de las actualizaciones), entrenamiento específico del personal de TI y mantenimiento.</li> <li>• Aplicaciones adecuadas al nuevo sistema operativo.</li> </ul>
<b>Personal</b>	El coste derivado del personal de TI necesario para la puesta en marcha y mantenimiento de los equipos puede verse dramáticamente afectado en función del tipo de sistema operativo (y software adicional) utilizado en los servidores.
<b>Pérdidas de servicio (menor fiabilidad)</b>	La implantación de un nuevo sistema operativo en los servidores incrementa inicialmente el riesgo de aparición de fallos que derivan en pérdidas inesperadas de servicio, lo que genera costes adicionales en términos de: <ul style="list-style-type: none"> <li>• Tiempo de trabajo del personal de TI necesario para reiniciar, ajustar o reconfigurar un servidor para un funcionamiento sin fallos con el nuevo sistema operativo. Depende de la frecuencia de los fallos, el tiempo necesario en repararlos y el salario medio del personal de TI.</li> <li>• El coste para la organización, en términos de pérdidas de productividad de los usuarios finales. A su vez, es función de la frecuencia y duración de los fallos, el número de empleados afectados y el salario medio.</li> </ul>
<b>Formación</b>	Los cambios introducidos en el software requieren destinar recursos para formar a los técnicos de TI en el nuevo sistema operativo. Serán proporcionales al número de empleados de TI y la magnitud del cambio.
<b>Servicios externalizados</b>	El volumen de servicios de TI externalizados depende del tipo de software que se establece como sistema operativo.

Fuente: enter, a partir de IDC (2002)

de costes contemplado en el período de cinco años, tal y como se puede apreciar en el gráfico 3.1, en el que se recoge la estructura media de costes para todos los escenarios contemplados, tanto para el caso de Windows 2000 Server como para Linux Server.

El coste de las licencias de software representaba sólo un 4,6 por 100 del TCO en los cinco años contemplados como horizonte relevante de una organización. Tampoco el coste asociado a la necesidad de nuevo hardware era elevado en el cómputo total, ya que sólo suponía un 4,4 por 100.

Por el contrario, los costes asociados al personal eran, con mucha diferencia, los más elevados: más del 62 por 100. El soporte técnico para la puesta en marcha y mantenimiento de los sistemas informáticos de una organización, elemento muy intensivo en mano de obra, se convertía en la partida fundamental del coste a medio plazo. Por tanto, el peso de este componente bajo sistemas operativos alternativos, y no el coste de las licencias, va a resultar el factor desequilibrante fundamental a la hora de elegir estrategias alternativas.



El estudio de IDC (2002), además de descomponer el TCO en las seis categorías contempladas en la tabla 3.4, comparó el coste derivado de la utilización de Windows 2000 Server y Linux Server en cinco escenarios distintos:

- **Servidores dedicados a servicios de red**
- **Servidores dedicados a servicios de archivos**
- **Servidores dedicados a servicios de impresión**
- **Servidores dedicados a servicios de seguridad**
- **Servidores dedicados a servicios de Internet**

La estructura de costes de cada una de estas alternativas aparece reflejada en la tabla 3.5. En tres de los cinco escenarios, el coste de las licencias de software era sustancialmente más elevado en el caso de la opción representada por el software propietario (Windows 2000 Server). En promedio, el coste del software era un 82,7 por 100 más elevado que en caso de Linux Server.

No puede olvidarse que el coste de las licencias de software es una parte muy pequeña del TCO, ya que supone solamente un 4,6 por 100 del total. Que las licencias y el software sean más caros en el caso del software propietario no es factor relevante. De hecho, la opción de software propietario genera, según el estudio, un ahorro del 30,6 por 100 en la partida de personal, la más importante del coste. Este ahorro, para situarlo en el contexto adecuado, es en media 8,7 veces superior al que el software libre producía en la categoría de licencias.

En los costes relacionados con pérdidas de servicio, la opción Linux Server mejoró los resultados de Windows 2000 Server en tres de los cinco tipos de servicio. El ahorro medio logrado por Linux fue del 21,3 por 100 con respecto a la opción de Microsoft. Pero este ahorro era cinco veces inferior al que la opción de software propietario había logrado en la categoría de personal.

**Tabla 3.5. Estudio IDC (2002). Windows 2000 frente a Linux.**

Comparación del TCO para Windows 2000 Server y Linux Server por tipo de servicio y categoría de coste (\$)

	Red		Archivos		Impresión		Seguridad		Internet	
	MS	Lx	MS	Lx	MS	Lx	MS	Lx	MS	Lx
Hardware	1.211	1.004	5.703	3.139	1.173	2.172	1.653	2.041	7.087	3.006
Software	211	940	3.988	1.009	1.665	340	5.829	6.609	7.107	1.390
Personal	8.392	8.201	54.030	81.204	40.247	59.080	50.609	71.056	15.102	23.015
Pérdidas servicio	1.412	1.494	30.133	20.788	38.857	39.746	10.335	4.385	1.646	1.541
Formación	534	677	5.191	7.670	4.787	5.282	2.000	6.445	1.304	1.584
Serv. exteriores	26	946	3	570	121	369	49	440	59	64
<b>TOTAL</b>	<b>11.787</b>	<b>13.263</b>	<b>99.048</b>	<b>114.381</b>	<b>86.849</b>	<b>106.989</b>	<b>70.495</b>	<b>90.975</b>	<b>32.305</b>	<b>30.600</b>
Diferencia MS-Lx	-12,50%		-15,50%		-23,20%		-29,10%		5,30%	

Fuente: enter, a partir de IDC (2002)

Por el contrario, cuando se hace referencia a la partida de formación, la opción representada por Windows 2000 Server volvió a mostrarse claramente superior a Linux Server, ya que supuso menores costes en todos los renglones, con un ahorro promedio del 36,2 por 100.

Cuando el análisis se realiza en términos de los distintos tipos de servicio, la opción representada por el software propietario de Microsoft se demostró claramente superior a Linux Server en cuatro de los cinco escenarios planteados, los de red, archivos, impresión y seguridad, con ahorros del 12,5, 15,5, 23,2 y 29,1 por 100 respectivamente. Sólo en la categoría de servidores dedicados a servicios de Internet el sistema operativo Linux Server supuso un TCO inferior a Windows 2000 Server, con un ahorro relativamente modesto, del 5,3 por 100.

La alternativa representada por el software propietario resultó ser, con carácter general, más económica que la de software libre. Un resultado que cuestiona la sabiduría popular, según la cual la gran ventaja del software libre es constituir una opción más barata.

Un estudio de estas características en el ámbito de los ordenadores personales ofre-

cería, con total seguridad, resultados mucho más desfavorables para el software libre, pues en el entorno de los PC se pondrían de manifiesto con mayor intensidad la vocación comercial y la preocupación por satisfacer al cliente por parte de las alternativas de software propietario comercial.

Así, Smith, D. M et al. (2003) señalan que *'la migración a Linux en el entorno de los ordenadores personales sólo resulta razonable en un número muy limitado de situaciones. La migración a Linux sólo debe plantearse en aquellos casos en que los ordenadores personales trabajen con un número reducido de aplicaciones de función fija o de bajo nivel, como introducción de datos o centro de llamadas. En estos casos, el coste de migración podría ser lo suficientemente bajo como para justificar el cambio a Linux.'*

Un estudio que analizó el TCO en el segmento de PC fue realizado por Unilog para la ciudad de Munich, que en noviembre de 2001 se planteó la actualización de los 14.000 PC de sus funcionarios municipales. El sistema operativo utilizado hasta entonces era Windows NT 4.0 y el paquete de ofimática empleado, Microsoft Office 97/2000.

Las alternativas planteadas para la migración fueron:

- **MS XP + MS Office XP**
- **MS XP + OpenOffice**
- **GNU/Linux + OpenOffice**
- **GNU/Linux + OpenOffice + PC Emulation**
- **GNU/Linux + OpenOffice + Terminal Server**

Los costes de migración de cada una de las cinco alternativas se reflejan en la tabla 3.6. El período considerado para la estimación de los TCO fue de cuatro años, suficientemente extenso para que se tuviesen en cuenta todos los componentes de coste. La opción que resultaba más económica era la que suponía una migración hacia el sistema operativo Windows XP y el paquete Office XP.

El caso de Munich, donde finalmente se optó por migrar hacia alternativas de software libre, se estudia en el apartado de experiencias internacionales del capítulo siguiente.

**Tabla 3.6. Estudio de Unilog para la ciudad de Munich.**

Comparación de costes de migración, millones \$

Alternativas	Coste
MS XP + MS Office XP	34,18
MS XP + Open Office	39,75
GNU/Linux + Open Office	45,77
GNU/Linux + Open Office + PC emulation*	35,94
GNU/Linux + Open Office + Terminal Server	50,00

\* (WINE/WMWare)

Fuente: enter, a partir de Unilog (2003)

## La fiabilidad del software, un aspecto clave en las empresas

La fiabilidad de un producto de software es una característica que citan a menudo los profesionales de las TI como una necesidad clave para cualquier tecnología y para cualquier empresa, porque el suministro de unos servicios fiables es una expectativa no negociable que exigen de los departamentos de TI las personas encargadas de la toma de decisiones empresariales y los usuarios finales.

El problema de la falta de fiabilidad proviene de dos escenarios, que a menudo se combinan y pueden llegar a afectar negativamente a la productividad empresarial y a la capacidad de las empresas para competir y crecer:

- Las soluciones empresariales no se encuentran disponibles cuando se necesitan —esto significa no sólo cumplir con las expectativas diarias y constantes de las TI (piense en el correo electrónico cada mañana), sino que también incluye el suministro sistemático y previsible de nuevas capacidades para ajustarse a las necesidades cambiantes de la empresa (piense en una compañía que incorpore personalización a su sitio web de comercio electrónico para seguir el ritmo de los competidores); y
- La capacidad de las TI para satisfacer las necesidades empresariales sólo mediante una combinación de “acciones heroicas” para superar comportamientos imprevisibles o fallos en las tecnologías, lo que puede generar unos días de trabajo muy intensos o soluciones a corto plazo para satisfacer las necesidades pero que, al mismo tiempo, ponen en riesgo la capacidad de las TI para ofrecer unas soluciones fiables y a largo plazo dentro del presupuesto.

Thompson (2005) desarrolló un estudio orientado a comparar la fiabilidad de dos plataformas: una basada en Windows Server System de Microsoft y otra en SuSE Linux Enterprise Server de Novell— bajo unos requisitos empresariales en evolución a lo largo de un período largo de tiempo. El experimento comparaba Windows 2000 Server con SuSE Linux Enterprise Server 8, simulando el período de un año desde el 1 de julio de 2004 hasta el 30 de junio de 2005. Durante dicho período, se simuló la evolución de una compañía de comercio electrónico con requisitos empresariales cambiantes, manteniendo al mismo tiempo la seguridad mediante la aplicación de parches. A finales del período, ambos sistemas pasaron a la versión más reciente de sus respectivos sistemas operativos, Windows Server 2003 y SuSE Linux Enterprise Server 9.

## La fiabilidad del software, un aspecto clave en las empresas (continuación)

Los parches de seguridad se aplicaron en intervalos de 1 mes, mientras que los nuevos requisitos empresariales aparecieron en periodos de tres meses. El experimento lo realizaron tres administradores expertos de Windows y tres administradores expertos en SuSE Linux <sup>(a)</sup>. La muestra, aunque era demasiado pequeña para ofrecer comparaciones estadísticas concluyentes, arroja algo de luz sobre algunas diferencias clave del modelo entre las plataformas.

Los resultados de este estudio inicial mostraron algunos patrones interesantes. Uno de los beneficios más repetidos de Linux era su gran modularidad y la granularidad del control que los administradores tienen sobre el sistema. El experimento mostró que dicha flexibilidad también conducía a una cierta ambigüedad para los administradores en lo que se refiere a los caminos que debían seguir para resolver algún conflicto.

En lo que respecta a Linux, cada administrador utilizó caminos diferentes para resolver conflictos de dependencia que surgieron cuando se instalaron nuevos componentes. El resultado fue unas soluciones cada vez más complejas y heterogéneas en el tiempo.

Por el contrario, Microsoft ha seguido una filosofía que denomina *innovación integrada*, donde gran parte de las principales funciones del sistema se incorporan al mismo sistema operativo. Durante el experimento, todos los administradores de Windows siguieron una ruta bastante homogénea para instalar parches y para aplicar actualizaciones de componentes según los requisitos de cada momento.

Los resultados mostraron la contribución de las diferentes tecnologías a la fiabilidad en el mundo real:

- **Respecto a la capacidad de ofrecer soluciones de forma previsible y sistemática**, dos de los tres administradores de Linux no fueron capaces de satisfacer todas las necesidades empresariales dentro del marco temporal del estudio. Por el contrario, los tres administradores de Windows cumplieron con todos los requisitos empresariales. La mayor complejidad fueron las dependencias de los componentes -a las que se debía realizar un seguimiento- fue un factor contribuyente clave para explicar el fracaso en el cumplimiento de los requisitos empresariales para la solución de Linux.
- **En lo referido a la capacidad de ofrecer un servicio fiable de forma eficiente**, utilizando para ello tiempos “de parada de servicio” (con un tiempo máximo permitido de 10 horas por tarea) para los administradores de Linux que no fueran capaces de cumplir con un requisito particular, como media los tres administradores de Linux fueron un 70 por 100 más lentos que sus homólogos de Windows a la hora de cumplir con los objetivos empresariales. Esto se vio impulsado en parte por la mayor cantidad de fallos del sistema que sufrieron los administradores de Linux (14 comparados con los 0 fallos de los administradores de Windows) y por la mayor cantidad de parches que necesitaron aplicar en los sistemas de Linux (187 en total, comparados con los 39 para Windows).
- **Respecto a la capacidad de mantener el control sobre la complejidad del entorno y puede ‘mantenerlo sencillo’**, el único administrador de Linux que tuvo éxito a la hora de cumplir con todos los requisitos, instaló los componentes y las versiones de los componentes que no se encontraban soportadas directamente por el fabricante (y que en algunos casos habían sido compiladas de manera personalizada) y que puso efectivamente su *sistema* en una configuración no soportada. Mientras que la configuración cumplió con los requisitos en funcionalidad, el administrador se quedó “solo” para resolver fallos potenciales del sistema en el futuro. También tiene una carga administrativa mayor para las TI, dado que cualquier parche futuro para los componentes no soportados tendría que ser recogido de fuentes alternativas y, en algunos casos, editado a nivel de código fuente y precompilado. En lo que se refiere a Windows, el sistema se mantuvo mediante los componentes ofrecidos por Microsoft o por una empresa suministradora de componentes, y todas las configuraciones se encontraban dentro de los límites del soporte.

Los resultados del experimento, que reafirman aspectos parciales del estudio de IDC (2002), pusieron de manifiesto que la fiabilidad en el ámbito empresarial de los productos de software libre resultó francamente deficiente en comparación con los de software propietario, un resultado que contradice una de las reivindicaciones más comunes de los defensores de la primera de las modalidades de software.

(a) Cada administrador de SuSE Linux contaba con, al menos, 5 años de experiencia administrando Linux en un entorno empresarial, 2 años administrando distribuciones SuSE Linux y 1 año administrando SuSE Linux Enterprise Server 8. Los administradores de Windows contaban todos al menos con 5 años de experiencia administrando servidores Windows en un entorno empresarial y 2 años administrando Windows Server 2003. Padilla, et al. (2004)

### 3.3. Enfoque macroeconómico

#### *Fomento del empleo y la innovación derivado de la industria del software comercial*

El papel decisivo de las nuevas tecnologías de la información y la comunicación (TIC) en el crecimiento del PIB y de la productividad de las economías modernas es un fenómeno ampliamente estudiado. Numerosos trabajos, como los de Jorgenson (2001, 2005), Van Ark e Inklaar (2005), Van Reeden y Sadun (2005) y ENTER (2006) han abordado esta cuestión desde distintas perspectivas.

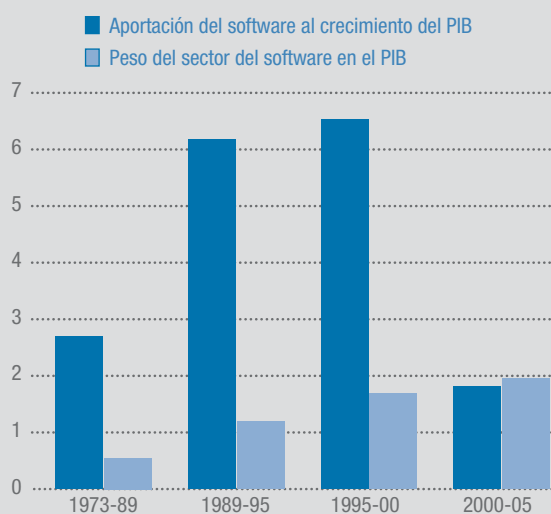
El sector del software comercial, dada su condición de integrante del conjunto de sectores TIC, ha desempeñado en las últimas décadas un importante papel como motor de las economías. El análisis del impacto macroeconómico del software comercial suele chocar con problemas metodológicos de compleja solución<sup>3</sup>, por lo que el estudio de algunas cuestiones del presente apartado sólo se realiza para la economía de los Estados Unidos, aunque no cabe esperar un comportamiento sustancialmente distinto en las economías europeas más avanzadas.

#### Efecto en el crecimiento del PIB

Un primer aspecto a destacar es el gran impacto del sector del software comercial en el crecimiento de la economía de los Estados Unidos, sobre todo cuando se tiene en cuenta su discreto tamaño dentro del total del PIB.

<sup>3</sup> Los problemas suelen estar relacionados con la disposición de series temporales de precios de los productos de software que permitan realizar comparaciones homogéneas. En este sentido, tienen plena aplicación las valoraciones utilizando índices de precios hedónicos. Mientras que para algunos países, como EEUU, se han construido largas series temporales de precios para el sector del software, en otros países no están disponibles.

#### 3.2. Sector del software comercial en EEUU. Peso en el PIB y aportación al crecimiento (% del total)



Fuente: *enter* a partir de Jorgenson (2005)

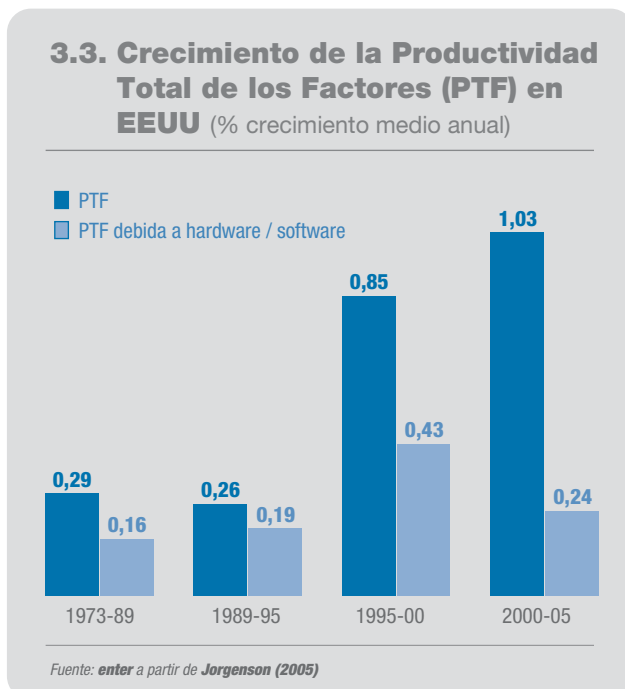
En el gráfico 3.2 se aprecia cómo el peso del sector del software comercial en la economía estadounidense ha venido creciendo de forma sistemática desde hace más de 30 años. En 1973 apenas suponía un 0,3 por 100 del total del PIB y hubo que esperar hasta 1990 para que, por primera vez, superase el 1 por 100. En los últimos 15 años, el peso prácticamente se ha duplicado y en 2005 representó un 1,82 por 100. Debe señalarse, no obstante, que esta cifra es algo inferior a las cotas alcanzadas en el bienio 2000-2001, cuando superó ligeramente el 2 por 100 del PIB.

Pese a su pequeño tamaño en términos de PIB, el sector del software comercial en los EE UU ha tenido un gran impacto en el crecimiento de la economía, tal como se aprecia en el gráfico 3.2. A modo de ejemplo, basta señalar que en la década de los 90 algo más del 6,3 por 100 del avance del PIB estadounidense se explicó, de forma exclusiva, por el impulso del software comercial.

## Impacto en la productividad

El dinamismo del sector de software comercial no se materializa solamente en un fuerte impulso al crecimiento económico. También ha tenido, en el pasado y conserva hoy día, un notable impacto en el aumento de la productividad de las economías desarrolladas.

El caso concreto de la economía norteamericana aparece reflejado en el gráfico 3.3, en el que se representa el crecimiento de la productividad total de los factores experimentada desde 1973, así como la parte achacable exclusivamente a la utilización de nuevos hardware y software en los procesos productivos. La consideración conjunta de hardware y software se realiza por las obvias complementariedades entre ambos elementos (un producto de software genera crecimientos de productividad si se emplea junto con el hardware adecuado, y lo mismo sucede con un nuevo producto de hardware).



En el gráfico 3.3 se aprecia claramente que el crecimiento de la productividad de la economía norteamericana en las últimas décadas es difícil de entender si no se considera

el papel protagonista que han tenido tanto el hardware como el software, ya que conjuntamente explican más del 52 por 100 de dicho crecimiento desde 1973. En el último período analizado se registra una caída en la contribución, pero no cabe esperar que este cambio suponga una alteración de la tendencia anterior, ya que no se puede olvidar que en dicho período queda encuadrada una crisis importante del sector TIC, que condiciona el resultado.

## Impulso del empleo. El caso de Windows Vista

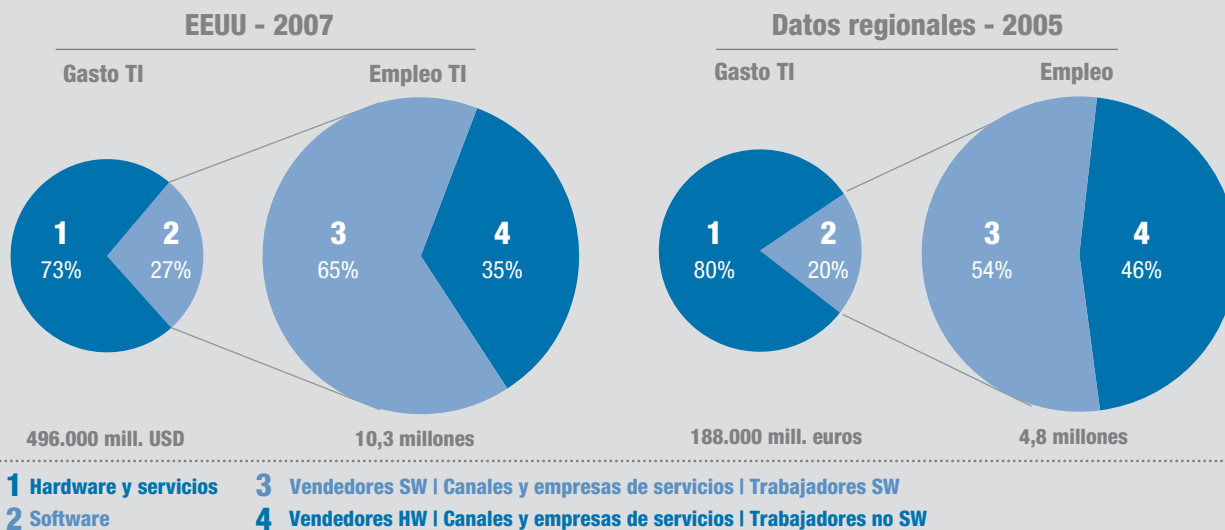
Otro aspecto que resalta la importancia del software comercial en una perspectiva macroeconómica es su contribución, en términos de empleo, al sector de las tecnologías de la información (TI).

En concreto, varios estudios han puesto de manifiesto que, dentro del sector de las TI, la relevancia del software comercial es mayor en términos de empleo que en términos de facturación. En el gráfico 3.4 se puede apreciar este fenómeno en los Estados Unidos: mientras que las ventas de software comercial suponen el 27 por 100 de la facturación del sector de TI, los empleos relacionados directamente representan el 65 por 100 del total del empleo de TI.

En Europa sucede algo similar. El gráfico 3.5 realiza la misma comparación para seis países europeos (Francia, Alemania, Reino Unido, España, Polonia y Dinamarca) y las conclusiones son parecidas: mientras las ventas de software representan el 20 por 100 de la facturación del sector de TI, los empleos dependientes son el 54 por 100 de la ocupación TI.

El peso específico del empleo en el sector del software dentro del sector de TI es una de las razones por las que cualquier inno-

### 3.4. / 3.5. Importancia del software dentro del sector de TI en el mundo



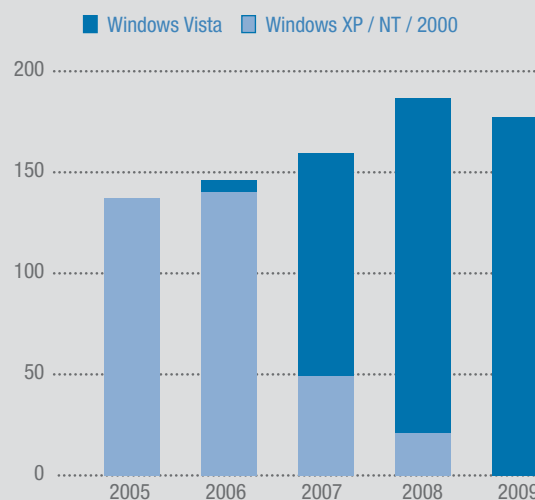
Fuente: Gantz, J. F., A. Gillen y M. Warmerdam (2006a,b)

vacación o nuevo producto de software que triunfe en el mercado tiene unos efectos macroeconómicos potenciales que van mucho más allá de los ya analizados.

Una forma de resaltar la relevancia que la introducción de nuevos productos de software comercial tiene desde una perspectiva macroeconómica es analizar un ejemplo concreto, como puede ser el reciente lanzamiento de Windows Vista por Microsoft. Varios estudios (Gantz, J. F., A. Gillen y M. Warmerdam, 2006) han prestado atención a este lanzamiento y tratado de medir sus consecuencias en términos de crecimiento del empleo y la facturación en el sector TI.

El lanzamiento de Windows Vista, el nuevo sistema operativo de Microsoft que sucede a Windows XP, se produjo a finales de 2006 para unos clientes empresariales seleccionados, aunque su difusión masiva se está produciendo en 2007. Las previsiones apuntan a que este año más del 68 por 100 de los nuevos clientes de sistemas operativos de Microsoft en el mundo utilizará Windows Vista y que, en 2009, la utilización por los nuevos clientes del sistema Windows será del 100 por 100.

### 3.6. Nuevos clientes de sistemas operativos de Microsoft en el mundo (millones de unidades vendidas)



Fuente: Gantz, J. F., A. Gillen y M. Warmerdam (2006a,b)

Los importantes efectos macroeconómicos derivados de la introducción de Windows Vista se deben, en primer lugar, al volumen de personas que lo adoptará en los próximos años. Las previsiones realizadas evalúan el número de nuevos clientes de Windows Vista, entre 2007 y 2009, en más de 430 millones en el mundo. No es

de extrañar que se prevea un gran impacto macroeconómico, dada la magnitud del cambio que supone.

Los estudios realizados destacan el efecto dinamizador que el lanzamiento de Windows Vista va a tener en el empleo de TI en el mundo. La tabla 3.6 refleja las estimaciones cuantitativas del empleo de TI que en 2007 va a estar relacionado directamente con el nuevo sistema operativo.

Cuando se tienen en cuenta los empleos directamente dedicados a la creación, venta, distribución, mantenimiento o prestación de servicios relacionados con sistemas operativos de Microsoft, se está hablando de aproximadamente el 20 por 100 del empleo TI, tanto en los Estados Unidos como en los principales países europeos o, lo que es lo mismo, que la actividad profesional de unos 2,8 millones de trabajadores de TI en sus respectivas áreas (hardware, software, servicios, etc.), va a estar vinculada en 2007 al nuevo sistema operativo.

**Tabla 3.6. Impacto de Windows Vista en el empleo de TI en 2007**

Pais	Total Empleo TI (miles)	Empleo TI relacionado con Vista (miles)*	Windows Vista (%)
EEUU	10.300	1.800	17,5
Alemania	1.607	320	19,9
R. Unido	1.406	282	20,1
Francia	1.250	249	19,9
España	513	115	22,4
Polonia	275	79	28,7
Dinamarca	151	27	17,9

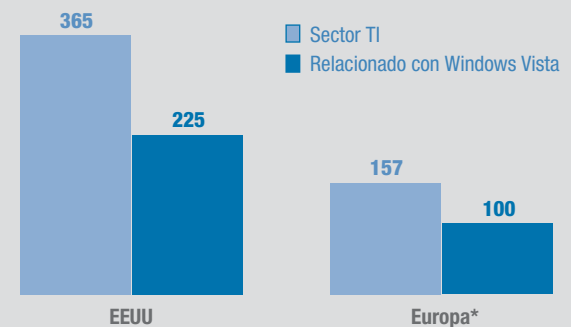
\* Empleos en empresas que crean, venden, distribuyen o prestan servicios de hardware o software relacionados con sistemas operativos de Microsoft, y empleados de TI en organizaciones que desarrollan aplicaciones, servicios o mantenimiento de dichos hardware o software.

Fuente: enter, a partir de Gantz, J. F., A. Gillen y M. Warmerdam (2006a,b)

Las cifras expuestas en la tabla 3.6 no hacen referencia al empleo creado gracias a Windows Vista, sino a los trabajadores cuya actividad va a estar directamente relacionada con dicho sistema operativo. La

medición del impacto cuantitativo *diferencial* generado por la aparición del nuevo sistema operativo, esto es, el impacto macroeconómico en términos de empleo, es la cuestión clave a analizar. Aparece reflejado en el gráfico 3.7.

**3.7. Ganancias de empleo de TI, total y debido a Windows Vista, en EEUU y Europa (2007, miles de empleos)**



\* Europa6: Alemania, Reino Unido, Francia, España, Polonia y Dinamarca

Fuente: Gantz, J. F., A. Gillen y M. Warmerdam (2006a,b)

Las previsiones de nueva ocupación en el sector TI en 2007 son en torno a 365.000 y 157.000 empleos Estados Unidos y los seis países europeos indicados, respectivamente. De estos nuevos empleos, de acuerdo con las estimaciones de Gantz, J. F., A. Gillen y M. Warmerdam (2006a,b), algo más del 62 por 100 corresponde exclusivamente al efecto generado por el lanzamiento de Windows Vista.

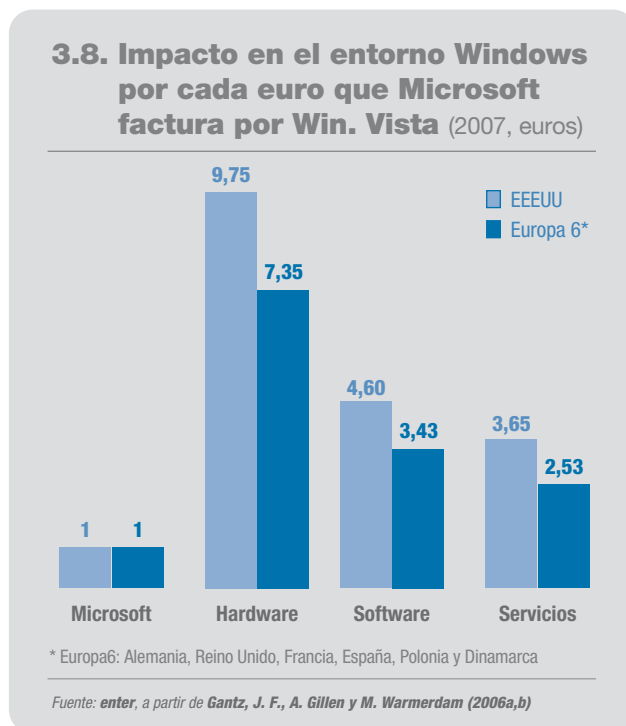
La evaluación de la distribución espacial de estos nuevos empleos sitúa 225.000 de los mismos en los Estados Unidos y unos 100.000 en los seis países europeos que se contemplaron en el estudio.

Otro efecto macroeconómico derivado de la comercialización de Windows Vista que ha sido estimado en los estudios realizados es el impacto que, en términos de facturación, tendrá en las empresas TI que generan productos relacionados con el *entorno Windows* (producen o comercializan hardware

que incorpora este software, diseñan o comercializan aplicaciones compatibles con Windows, prestan servicios relacionados con este sistema, etc.)

Lo que se ha estimado es, en esencia, el *efecto multiplicador* de los ingresos de Windows Vista; es decir, cuánto crece la facturación de dichas empresas por cada euro que ingresará Microsoft por la venta del nuevo sistema. Si el grueso de los ingresos fuese para Microsoft, el resto de empresas no experimentaría cambios significativos en su facturación y podría concluirse que el impacto macroeconómico del nuevo sistema operativo, en términos de facturación, es limitado. Pero el gráfico 3.8 supone un argumento a favor de la idea contraria.

De acuerdo con las estimaciones realizadas por Gantz, J. F., A. Gillen y M. Warmerdam (2006a,b), por cada euro que Microsoft facture por la comercialización de Windows Vista, las empresas del *entorno Windows* ingresarán 18 euros adicionales en Estados Unidos y 13,31 euros en los seis países europeos estudiados (gráfico 3.8) Más



de la mitad de esos ingresos adicionales irá al segmento del hardware sobre el que discurre Windows Vista y el resto se repartirá entre los productores y comercializadores de otros paquetes de software que utilicen dicho sistema operativo y los proveedores de servicios

# 4.

## El papel de las administraciones públicas ante el software libre y comercial

### 3.1. ¿Está justificada la inversión pública? Los fallos de mercado

**Un fallo de mercado se produce cuando el mercado, por sí solo, no es capaz de asignar correctamente los recursos.** La existencia de fallos de mercado ha sido muy analizada en economía y se han determinado, en principio, cuatro razones por las que pueden producirse (tabla 4.1).

El primer origen de los fallos de mercado, la existencia de información incompleta, no parece ser el caso del software. El mercado de software es suficientemente

**Tabla 4.1. Fallos de mercado**

<b>Información incompleta</b>	Cuando los agentes participantes en el mercado disponen de información incompleta, sus decisiones serán ineficientes con carácter general.
<b>Monopolio natural</b>	Responde a la situación en que sólo existe mercado para una única empresa, por lo que, de forma natural, se genera un monopolio para abastecer de un determinado producto.
<b>Bienes públicos</b>	Bienes que por su naturaleza (falta de rivalidad en su consumo e imposibilidad de exclusión mediante precio) no pueden ser ofrecidos por el mercado.
<b>Externalidades</b>	Situaciones en las que las actuaciones de individuos o empresas afectan, positiva o negativamente, a otros, sin que dicho efecto quede recogido en los precios del mercado.

transparente, ya que tanto los productores como los consumidores disponen de la información relevante: los primeros, en todo lo relacionado con sus proveedores y los mercados a los que dirigirse, y los consumidores, a su vez, conocen las alternativas y los precios de los productos que pueden cubrir sus distintas necesidades.

La existencia de monopolios naturales en el ámbito del software comercial tampoco puede sustentarse sobre los datos de evolución de precios y producción de este mercado. En la sección 2.1 se ha podido comprobar que el mercado del software comercial ha registrado importantes reducciones medias de los precios en las últimas décadas y notables incrementos en la producción, comportamientos ambos que no corresponden a una situación de monopolio natural.

La naturaleza peculiar de los productos de software comercial, fácilmente replicables a un coste marginal próximo a cero, induce la necesidad de intervención de las autoridades públicas en el ámbito de la protección de los derechos de propiedad, cuestión abordada en profundidad en un estudio anterior (ENTER, 2006). Es una cuestión que no implica que el software comercial sea inferior en ninguna medida al software libre no comercial, sino que el funcionamiento de las economías de mercado requiere la preservación de los derechos de los creadores para que se mantenga un proceso de innovación.

La cuarta fuente de fallos de mercado, las externalidades, tiene una manifestación en el ámbito del software, aunque no es propia sólo de este mercado. Es conocido que la inversión en I+D, sobre todo en investigación básica, es una actividad con grandes externalidades, puesto que la sociedad obtiene beneficios derivados de dicha inversión, muy superiores en conjunto a lo que el investigador llega a apropiarse. Este hecho tiene una consecuencia inmediata: en ausencia de un

mecanismo que corrija las externalidades, el nivel de esfuerzo e inversión en I+D resulta muy inferior al socialmente deseable.

En la medida en que la innovación en materia de software responde a esta caracterización es normal que se promuevan iniciativas públicas, como el apoyo financiero del gobierno a los esfuerzos en I+D o el refuerzo de los mecanismos de protección de las innovaciones, para corregir los fallos de mercado.

Sin entrar a fondo en la discusión sobre si el apoyo público a la I+D en el sector del software es o no acertado desde un punto de vista económico, sí que resulta inadecuada, por sus consecuencias negativas, una actuación discriminatoria basada en ofrecer apoyo público a la I+D exclusivamente para software libre, un fenómeno que se ha producido recientemente en algunos países.

En apartados anteriores se han estudiado las características diferenciales entre software libre y propietario (sección 3.1), así como la evolución reciente del mercado. Del análisis realizado se desprende que no parecen existir en este sector fallos de mercado de tal magnitud que justifiquen la intervención de los poderes públicos a través de una regulación orientada a reducir los precios o incrementar la competencia o, más aún, a reordenar el mercado, favoreciendo algunas opciones frente a otras: software libre frente a propietario.

La evidencia, puesta de manifiesto en la sección 2.1, es más bien la contraria, ya que la experiencia histórica apunta a que el sector del software ha funcionado de manera bastante eficiente sin necesidad de intervenciones públicas, al menos desde la perspectiva del consumidor, que ha podido beneficiarse de notables reducciones de precios, aumento en la diversidad y gama de productos ofrecidos y una gran innovación.

De forma más concreta, desde determinados grupos (normalmente parte interesada) se ha argumentado que la posición de algunas empresas de este mercado es muy fuerte frente a las demás. Es decir, se alerta de la posible existencia de actuaciones en régimen de monopolio de facto o de oligopolio.

Pero los estudios realizados (véase Padilla, A. J. et al., 2004) apuntan a que dicho predominio suele caracterizarse por ser de naturaleza temporal y limitado a productos muy específicos. La consecuencia lógica ya se ha presentado en los gráficos 2.3 y 2.4: el sector del software está bastante fragmentado en su conjunto y, en muchos casos, se caracteriza por altas dosis de competencia, con constantes entradas y salidas de empresas en el mercado y alteraciones en las posiciones de liderazgo.

## Razones esgrimidas para la intervención

A pesar de la evidencia económica al respecto, el sector del software ha sido uno de los focos donde más se ha concentrado el debate en torno a la intervención de las autoridades para alterar la libre actuación de las empresas privadas.

La intervención de las autoridades puede estar económicamente justificada en los casos que se han referido previamente, cuando se corregían fallos de mercado causados por externalidades o por dificultades de los autores para apropiarse de sus creaciones.

Sin embargo, el debate sobre la intervención del sector público se centra más en otra cuestión. Las administraciones públicas son importantes demandantes de software, por lo que han de tomar, como cualquier otro usuario (individual o empresarial), decisiones sobre la elección de los productos de software que utilizarán. Dichas de-

**Tabla 4.2. Razones esgrimidas por las autoridades públicas para proporcionar apoyo político al software libre**

<b>Seguridad, estabilidad, privacidad</b>	El software libre es más seguro y estable que el propietario, al tiempo que garantiza mejor la seguridad del usuario.
<b>Ahorro de costes</b>	La utilización de software libre permite ahorro de costes en la administración pública.
<b>Independencia</b>	Al emplear software libre se evita la dependencia de grandes empresas multinacionales.
<b>innovación</b>	El software libre es más innovador.
<b>Estímulo de la competencia</b>	El apoyo al software libre permite estimular la competencia en el mercado.
<b>Ayudas a la industria nacional</b>	El interés nacional puede justificar el apoyo a la industria nacional de software libre, para permitir su desarrollo, imposible en un entorno no protegido.
<b>Motivos ideológicos</b>	Optar por el software libre incrementa la libertad de los ciudadanos y crea un nuevo escenario para la acción democrática.

Fuente: enter, a partir de Padilla et al. (2004)

cisiones, que en el ámbito empresarial se adoptan casi sin excepción con un criterio de comparación de costes y beneficios, no siempre se ajustan a estos criterios en el ámbito público.

En este sentido, como señalan Padilla, A. J. et al. (2004), en los últimos tiempos se han producido casos en los que desde ciertas instancias políticas se ha querido intervenir para favorecer el software libre. Estas iniciativas van normalmente mucho más allá de terrenos puramente técnicos y económicos y se adoptan dentro del ámbito político, en el que las cuestiones referentes al análisis de costes y beneficios son obviadas en muchos casos.

Las razones esgrimidas para apoyar estas iniciativas políticas a favor del software libre son de naturaleza variada (tabla 4.2) y, en muchos casos, sencillamente falsas.

Una primera razón, frecuentemente utilizada, es que el software libre es más seguro o

más estable que el propietario. Así lo afirmaba, en 2003, el gobierno alemán, por citar un ejemplo. Y así lo ha señalado el Congreso de los Diputados español en una Proposición no de Ley que se debatió en diciembre de 2006 (sección 4.3)

La realidad en torno a la mayor seguridad del software libre es bastante distinta a lo que se afirma. Por una parte, hay expertos que señalan que los numerosos ataques que ha sufrido el sistema operativo Windows no se deben a que Linux (por poner un ejemplo) sea más seguro, sino a que es el sistema operativo más utilizado y, por tanto, el más atractivo para los *hackers*. Según este argumento, si el mercado de sistemas operativos estuviese dominado por Linux, los ataques estarían mayoritariamente orientados hacia él.

La experiencia en sistemas operativos de servidores, donde la presencia de Linux es mucho más intensa, corrobora esta sospecha, ya que se ha mostrado vulnerable a ataques como el caso del virus *Slapper*. Los estudios realizados al respecto no ofrecen ningún resultado concluyente que apoye la tesis de que el software comercial sea menos seguro que el software libre.

Por otra parte, los argumentos que justifican que el software libre permite una mejor solución de los errores de programación presentan debilidades, ya que no hay evidencia de que exista un mayor número de individuos interesados por descubrir errores en el software libre que en otro tipo de software. Además, las revisiones del software comercial son más ordenadas y estructuradas que las del software libre, tal como se comentó en la sección 3.1.

En el ámbito de la privacidad, no están claros los incentivos que tendrían los productores de software comercial para enmascarar en sus programas procedimientos para obtener

de forma ilícita información de los usuarios, debido al coste reputacional y económico que soportarían cuando estos hechos fuesen conocidos. Por tanto, el peso de los argumentos que defienden la superioridad del software libre en materia de privacidad puede estar basado, no tanto en lo que dicen los hechos, sino en especulaciones sobre lo que teóricamente podría producirse.

La segunda gran línea argumental que las autoridades públicas han utilizado para defender una apuesta unilateral por el software libre es que permite un gran ahorro de costes en relación con la utilización de software propietario. La evidencia aportada en la sección 3.2 parece desmentir de forma tajante dicha presunción.

Un tercer argumento empleado es que el uso de software libre evita una excesiva dependencia de un proveedor que, en muchos casos, es una empresa multinacional.

No se expresa de forma clara cuál es el problema derivado de que el proveedor sea una multinacional: las empresas de estas características demuestran, precisamente, que sus productos son preferidos por los consumidores en todo el mundo. Además, la pérdida de reputación que pueden experimentar si ofrecen un trato deficiente a una administración pública sería muy elevada, ya que este tipo de contratos son muy *visibles*: constituyen una buena publicidad para la empresa que los suscribe y, por otra parte, la calidad del servicio es monitorizada por muchos agentes además de por cliente y proveedor.

Precisamente, la dependencia de un proveedor es una cuestión que se ha discutido en la sección 2.3, en relación con el problema del *lock-in*. La empresa multinacional tiene mucho más que perder que un pequeño proveedor no empresarial como consecuencia de un servicio insatisfactorio, por lo que su grado de compromiso con el cliente

suele ser mucho mayor. Además, la adaptación del producto a las necesidades de la administración pública lo convierte en un producto no del todo estándar, esté basado en software propietario o libre, por lo que el descuelgue del proveedor perjudica a la administración en ambos casos.

Otra cuestión que se suele argumentar con frecuencia para apoyar el software libre es atribuirle la condición de más innovador que el propietario. Este argumento ha sido cuestionado con anterioridad (sección 2.2). La experiencia pone de manifiesto que el software libre, salvo contadas excepciones, no ha sido más innovador que el propietario. Pero, además, el grado de innovación de un tipo de software no debiera tener gran relevancia como factor decisorio cuando una administración pública se plantea qué tipo de software adopta, porque el tipo de tareas que se van a desarrollar son rutinarias y relativamente tasadas.

El papel que la administración, en su papel de cliente y demandante en el mercado del software, puede adoptar para estimular la competencia es otro argumento empleado con frecuencia para justificar el apoyo al software libre. Se arguye que si la administración apoya a este tipo de productos consigue incrementar la competencia en este mercado, al brindar a nuevos actores la posibilidad de que desarrollen sus productos.

La justificación de la intervención, vía el estímulo de la competencia, deja a un lado el hecho de que en el contexto del mercado global de productos de software la dimensión como cliente de una administración es muy reducida, por lo que sus decisiones difícilmente van a modificar la situación competitiva del mercado.

Por otra parte, no debe obviarse que la competencia también se estimula de forma eficiente por el simple hecho de que la admi-

nistración elija, de acuerdo a unos criterios objetivos basados en el coste y en sus necesidades, el mejor producto de software a su disposición. Y, de hecho, desde la perspectiva de la legislación europea de competencia, resultaría chocante que una administración pudiese apoyar de forma unilateral a una parte de los competidores del mercado de software, cuando no puede hacerlo en otros sectores.

En cuanto a la competencia, finalmente, la evidencia empírica reflejada en la sección 2.1 pone en duda la utilización de este principio como justificación de la intervención pública. En el mercado del software ya existe competencia y en aquellos ámbitos en los que el software libre se ha mostrado más competitivo (caso de los servidores) ya goza de una cuota de mercado muy elevada, sin necesidad de intervenciones de la administración.

El apoyo político al software libre también se ha justificado aludiendo a la necesidad de apoyar y estimular la industria nacional. Este principio resulta difícil de compatibilizar con el de defensa de la competencia, ya que ésta puede, perfectamente, proceder del exterior.

Además, la defensa de una industria nacional naciente ha sido un error repetido durante siglos por las políticas proteccionistas de innumerables países, en prácticamente todos los sectores económicos. Al proteger *temporalmente* una determinada actividad, como podría ser el desarrollo de software libre, se eliminan los incentivos para que ésta tenga que competir, por lo que al final la protección tiene que mantenerse de forma permanente.

La utilización de principios ideológicos para justificar la defensa del software libre por parte de la administración se basa en atribuir a este tipo de productos de software unos valores relacionados con la libertad y

la democracia superiores a los productos sujetos a las leyes del mercado.

La adopción de esta idea resulta cuanto menos paradójica, ya que el mecanismo de mercado es el que ha facilitado y permitido ejercer la defensa de las libertades a todos los agentes. Son innumerables las voces reputadas que han defendido este planteamiento, desde Milton Friedman, para el que la libertad de mercado y el mecanismo de los precios han sido las soluciones a los más graves problemas económicos de las sociedades occidentales, hasta el filósofo español Gustavo Bueno, para quien *'si no existe el mercado, no existe la democracia'*.

## 4.2. El análisis de costes y beneficios (*value for money*) y el principio de la neutralidad tecnológica

Del análisis anterior se deduce que no existen fallos de mercado en el terreno del software que justifiquen una intervención en defensa, protección y promoción del software libre por parte de las administraciones públicas. En los casos en los que se ha actuado de esa forma se han empleado argumentos que pueden ser y de hecho han sido suficientemente desmontados.

El paso siguiente es determinar cuáles deben ser los criterios que guíen las intervenciones de las autoridades en el terreno del software. Existe un amplio consenso en torno a la bondad de la adopción de dos principios.

En primer lugar, el análisis de costes y beneficios. Los recursos utilizados por las ad-

ministraciones públicas proceden de los impuestos de los ciudadanos y sus rectores deben asumir la responsabilidad de emplearlos de forma eficiente. Por tanto, en tanto que las administraciones son usuarias de software, deben tomar sus decisiones de elección de productos de acuerdo con un principio de racionalidad económica, en la que un análisis riguroso de los costes y beneficios de cada opción sea el criterio fundamental de decisión.

Este principio ha sido adoptado como fundamental por gobiernos de países tan avanzados como Alemania, Estados Unidos, Reino Unido o Dinamarca, entre otros.

A la hora de valorar los costes de las distintas alternativas de software es necesario plantear un escenario de medio y largo plazo (sección 3.2), puesto que las estructuras de coste de las alternativas son muy variadas. En última instancia se persigue determinar cuál es la opción que mejor se ajusta a las necesidades con un menor coste. Se trata, por tanto, de hacer el mejor uso posible de los recursos destinados a cubrir las necesidades de software. Este principio también es denominado *value for money*.

Relacionado con ese principio está el de la neutralidad tecnológica. En un análisis riguroso de costes y beneficios no cabe ningún planteamiento, de naturaleza ideológica o de otro tipo, que deje al margen a ninguna opción de software derivada de su naturaleza (libre o propietario). Todas las opciones han de ser tenidas en cuenta y valoradas de acuerdo con un mismo rasero. Debe adoptarse, por tanto, una actitud pragmática ante el software, lo que implica una posición de *neutralidad* respecto a la tecnología estudiada.

Pero el principio de neutralidad tecnológica no sólo debe ser tenido en cuenta por la

administración pública en su condición de usuario de productos de software. También, y de manera más importante, debe pesar en su papel como generadora de legislación, ya que en varios países existen presiones para que las autoridades aprueben medidas de protección y promoción de un determinado tipo de software, lo que supone una violación del principio de neutralidad tecnológica. En este sentido puede mencionarse la Proposición no de Ley en el Congreso de los Diputados español, relativa a la promoción del software libre, recientemente aprobada (diciembre de 2006).

Es destacable el hecho de que hayan surgido iniciativas internacionales, tanto públicas como privadas, para defender la adopción estos principios fundamentales en el ámbito del software. Una de estas ellas es la denominada *Iniciativa para la Elección de Software –Initiative for Software Choice–* ([www.softwarechoice.org](http://www.softwarechoice.org)).

Dicha iniciativa se define a sí misma como *‘una coalición internacional de pequeñas y grandes empresas destinada a impulsar la competencia dentro del mercado del software, permitiendo que éste se desarrolle y crezca a salvo de obstáculos e impedimentos creados por preferencias o imposiciones de los gobiernos.’* El objetivo es proporcionar, de manera activa, información sobre los beneficios de este método y su valor para innovaciones futuras a los legisladores y reguladores de todo el mundo.

Esta asociación ha establecido su propio catálogo de principios que deben guiar la elección de productos de software, plenamente compatibles con los de análisis de costes y neutralidad tecnológica. Dichos principios son:

- ***Elección del software por sus méritos y no por preferencias categóricas.*** Todos los productos de software ofrecen

beneficios y costes variables. Las entidades públicas deberían disponer del software que les proporcione una mejor satisfacción de sus necesidades, evitando preferencias hacia el software de código fuente abierto, el software propietario, el híbrido u otros modelos de desarrollo de software. Los gobiernos reciben un mejor servicio cuando pueden seleccionar y elegir el software entre una amplia gama de productos, basándose en consideraciones como el valor, el coste total de la propiedad del software, el conjunto de características, el rendimiento o la seguridad. Los gobiernos deberán permitir que el mercado siga fomentando la innovación en el desarrollo de software y evitar intervenir por medio de preferencias o requisitos de contratación pública que discriminen a un modelo de software en favor de otro.

- ***Promover una amplia disponibilidad de investigaciones financiadas por los gobiernos.*** Durante muchos años, los gobiernos han proporcionado contribuciones importantes a la tecnología, financiando las investigaciones básicas de software. Cuando se utilizan fondos públicos para respaldar las investigaciones y el desarrollo de software, las innovaciones resultantes de esas obras se deberían conceder bajo licencia, de modo que tomen en consideración tanto la conveniencia de compartir ampliamente esos avances como el deseo de aplicarlos a productos comerciales. La amplia difusión de los resultados contribuye a mantener un ciclo sostenible de innovación, donde la financiación por parte del gobierno de las investigaciones básicas hace crecer el conjunto de conocimientos disponibles para el público, además de contribuir a que también se produzcan avances en los productos comerciales. A su vez, éstos crean los empleos, beneficios e ingresos fiscales

necesarios para financiar futuras investigaciones públicas.

- **Fomentar la interfuncionalidad del software por medio de normas de plataforma neutrales.** El establecimiento de estándares voluntarios, impulsados por la industria, es el modo más eficaz de desarrollar normas neutrales con respecto a plataformas basadas en el mercado. Cuando esas normas estén abiertas y disponibles para todos, por medio de concesiones de licencias razonables y no discriminatorias, ayudarán a los desarrolladores a crear productos que puedan ser interfuncionales entre ellos. Es importante que las políticas de los gobiernos reconozcan que las normas abiertas -a disposición de todos los desarrolladores de software- no sean sinónimo de fuente abierta ni la requieran, sea para su adopción o utilidad. Los desarrolladores de software comercial que no hacen público su código fuente proporcionan contribuciones tecnológicas y de propiedad intelectual que se necesitan para desarrollar nuevas normas. Las políticas del gobierno sobre normas de software no deberían efectuar discriminaciones a favor ni en contra de ningún modelo de desarrollo en particular.
- **Mantener una protección firme de la propiedad intelectual.** Los legisladores no deberían hacer que las opciones rígidas de concesión de licencias de propiedad intelectual sean una condición previa de aceptabilidad para la contratación pública, ni ejercer discriminación entre los desarrolladores que decidan conceder licencias de su propiedad intelectual en términos comerciales y los que decidan no cobrar honorarios por ello. Los desarrolladores de software comercial y los basados en la comunidad dependen de los derechos de pro-

piedad intelectual, aun cuando algunos de ellos tratan de obtener compensación por el ejercicio de sus derechos y otros renuncien a hacerlo. Al permitir a los poseedores de esos derechos ofrecer una gama de licencias de propiedad intelectual, se fomentan las opciones y se impulsa la innovación.

### 4.3. Experiencias internacionales de promoción pública del software libre

*'A product replaces another one if it offers customers an incentive for a change which outweighs the change costs or which overcomes the resistance to change. A replacement product offers customers a high value than the product that was previously used.'*

Michael Porter

En los últimos años, los promotores y defensores del software libre han conseguido llamar la atención de gobernantes y responsables de numerosos países. Como resultado, las administraciones públicas de algunos de ellos han empezado a actuar en una doble vertiente: por una parte, adquiriendo software libre para utilización propia y, por otra, desarrollando iniciativas legislativas para promoverlo.

Resulta cuanto menos llamativo que la mayor parte de dichas iniciativas no haya sido respaldada por estudios económicos que hayan dejado constancia de la superioridad, técnica y económica, del software libre frente a las alternativas ofrecidas por el mercado. En general, el motor de las iniciativas no ha sido alimentado con rigor económico, sino con motivaciones ideológicas.

Un repaso por la experiencia internacional en materia de iniciativas públicas de promoción del software libre, sirve para destacar algunos casos notables, tanto de iniciativas bien fundamentadas y sustentadas con estudios solventes como de otras promovidas pese a la ausencia de argumentos sólidos de tipo económico.

## Alemania

El KBSt (*Koordinierungs und Beratungsstelle der Bundesregierung für Informationstechnik in der Bundesverwaltung*), organismo encargado de la coordinación de la política relacionada con las TI en la administración federal alemana, publicó en julio de 2003 un informe en el que marcaba unas líneas maestras para juzgar una posible migración hacia sistemas informáticos basados en el software libre en las instituciones públicas.

El informe se guía por el principio inspirador de que la directriz principal a la hora de juzgar cualquier tipo de migración debe ser el análisis de costes y beneficios (*value for money*). A partir de dicho principio, el KBSt desarrolla una metodología que permite evaluar la conveniencia de migrar hacia sistemas basados en software libre, que se resume en los siguientes puntos:

- El enfoque adecuado para evaluar la conveniencia de utilizar un determinado producto de software es el análisis coste-beneficio.
- Cuando este enfoque resulta demasiado complicado, puede optarse por un planteamiento más pragmático, basado en dos elementos:
  - Un análisis de tipo monetario, en el que se calcula el coste del producto teniendo en cuenta los costes directos

e indirectos en un contexto temporal de cinco años. Del mismo modo se calculan los posibles ahorros de coste derivados de su implantación. Se trata de un enfoque virtualmente idéntico al analizado en la sección 3.2.

- Un análisis cualitativo de los beneficios derivados de la implantación del producto. Precisamente en este bloque entran en juego todos los elementos que los defensores del software libre consideran esenciales para decantar la balanza en su favor.
- De acuerdo con el KBSt, la decisión final debe atribuir prioridad al criterio monetarista y sólo debe recurrir al análisis cualitativo cuando los resultados no sean concluyentes, dado el campo para la arbitrariedad que subyace en este segundo análisis.
- El KBSt asume la reducida dimensión de la administración, en el sentido de que una decisión de migración de un sistema a otro no alterará las condiciones generales del mercado de software, ya que el cambio no es de un volumen tal como para alterar la competencia.

En cuanto a las experiencias concretas relacionadas con la introducción de software libre en la administración pública alemana, son numerosas y han sido muy estudiadas. Dos ejemplos concretos, el Parlamento alemán y la ciudad de Munich, resultan singulares por la adopción de decisiones en un sentido contrario al indicado, por el coste relativo de las opciones planteadas.

La iniciativa denominada Bundestux realizó una labor de lobby para conseguir la implantación de Linux en las infraestructuras de TI del Parlamento alemán. La iniciativa defendía que la *'introducción de un sistema operativo libre en el Parlamento alemán'*

constituía una *‘señal necesaria para Alemania, tanto por razones de índole reguladora como de competencia.’*

Se encargó un estudio de viabilidad a la empresa de consultoría Infora, quien recomendó el empleo de Linux en el Parlamento sólo en el caso de los servidores de e-mail. De acuerdo con el informe, en el resto de servicios, así como en las aplicaciones de escritorio, era mejor seguir utilizando productos del entorno Windows, debido a que los productos de software libre todavía ofrecían

funcionalidades muy insuficientes desde el punto de vista de los usuarios finales.

El Parlamento alemán desoyó los consejos del informe que había encargado y decidió utilizar Linux en los servidores de archivo e impresión, así como en el servicio de directorio, pese a que estas opciones costaban unos 80.000 euros más que las alternativas propuestas por la consultora. Para justificar esta decisión, los responsables apuntaron a las ventajas estratégicas que podría suponer en el futuro, pese a su difícil y etérea cuantificación.

### El insólito caso de la ciudad de Munich

En noviembre de 2001, las autoridades de la ciudad alemana de Munich decidieron evaluar el coste que tendría la migración del software utilizado hasta ese momento por sus 14.000 funcionarios. Las alternativas planteadas incluían tanto opciones de software propietario como de software libre.

En abril de 2002 se encargó un estudio sobre esta cuestión a la consultora Unilog, que se llevó a cabo entre agosto y diciembre de ese año, con un añadido en julio de 2003. El objetivo del estudio era analizar el estado de las infraestructuras de TI del ayuntamiento y servicios anejos de la ciudad, mediante un cuestionario y, en segundo lugar, evaluar las posibles configuraciones alternativas desde una múltiple perspectiva que tuviese en cuenta la viabilidad técnica, la comparación entre costes y beneficios, así como otros criterios de tipo estratégico.

El punto de partida, determinado por el software utilizado en el momento de acometer el estudio, consistía en la utilización de Windows NT 4.0 como sistema operativo y Microsoft Office 97/2000 como paquete de ofimática. Las alternativas estudiadas fueron cinco:

- **MS XP + MS Office XP**
- **MS XP + OpenOffice**
- **GNU/Linux + OpenOffice**
- **GNU/Linux + OpenOffice + PC Emulation (WINE / VMWare)**
- **GNU/Linux + OpenOffice + Terminal Server**

Aunque el uso extendido de aplicaciones especializadas presentaba un reto potencial para la migración, ya estaba prevista una sustitución de dichas aplicaciones para el año 2007, con independencia de la alternativa que se terminase eligiendo para la migración de sistemas operativos y paquetes de ofimática. Por tanto, Unilog consideró que tanto las aplicaciones más estandarizadas como las especializadas se reemplazarían sin un esfuerzo adicional, puesto que ya estaba programado un cambio de generación de las mismas. Eso sí, se estimó que mantenerse en el entorno Windows permitiría seguir usando parte de dicho software, aunque incluso en el caso de la actualización desde Office97 hasta OfficeXP requeriría la adaptación o reescritura de un 20 por 100 de macros, formularios y otras herramientas existentes.

Por otra parte, Unilog estimó que tanto la utilización de Windows XP como de Linux requeriría PCs con una velocidad mínima de procesamiento de 500 MHz y una memoria de 256 MB, lo que suponía la necesidad de renovar aproximadamente la mitad de los ordenadores utilizados hasta ese momento. En cuanto al resto de hardware (impresoras, escáner, etc.) serían necesarios nuevos *drivers*, que era previsible estuviesen disponibles de forma más sencilla para el entorno XP que para Linux.

De acuerdo con toda esta información, Unilog calculó el coste total de la migración en un horizonte de cuatro años para cada una de las cinco alternativas (tabla 4.3)

En todas las opciones contempladas, más de la mitad del coste total estaba ligado a la formación en el uso de los nuevos sistemas y aplicaciones, aunque en las que incluían productos de Microsoft dicho coste era algo menor.

### El insólito caso de la ciudad de Munich (continuación)

La conclusión inicial de Unilog fue que la solución XP/XP ofrecía una ligera ventaja en términos de coste frente a las alternativas.

Pero las autoridades municipales solicitaron a Unilog que considerase criterios estratégicos y cualitativos en su recomendación, difícilmente expresables en términos monetarios.

Estos criterios, relacionados con aspectos como la legislación en materia de privacidad, el impacto de la seguridad de las tecnologías de la información, el atractivo de las condiciones de trabajo para el personal, la complejidad en el mantenimiento de los sistemas elegidos, o valores como la utilización de estándares abiertos, el carácter independiente del proveedor del software, la estabilidad y continuidad en la relación comercial futura o la protección de las inversiones, planteaban un reto para una medición rigurosa, puesto que introducen notables componentes de arbitrariedad, al tiempo que desdeñan el criterio de Total Cost of Ownership (TCO). La empresa evaluadora asignó a estos conceptos una puntuación, que podía oscilar entre 0 y un máximo de 10.000 puntos.

Las autoridades municipales consideraban que la dependencia de Microsoft era aceptable siempre que se mantuviese una independencia en áreas cruciales, como la posibilidad de desarrollar servicios de *backoffice* sin emplear productos de Microsoft. Unilog estimó que la nueva generación de productos de esa empresa dificultaba esta tarea, por la mayor integración de sus productos, lo que le restó puntos a las opciones que los incluían.

De acuerdo con estos nuevos criterios cualitativos, las alternativas de software libre ganaron posiciones, al puntuar mejor en aspectos como privacidad, seguridad o flexibilidad, sobre los que, como se ha señalado en secciones previas, las ventajas son espurias. Por otra parte, se valoraron positivamente aspectos un tanto inauditos, como el carácter no comercial de las opciones de software libre utilizadas, lo que potencialmente puede crear problemas futuros, al no existir mecanismos de mercado que garanticen la continuidad en el tiempo de los productos de software utilizados.

La inclusión de estos criterios cualitativos dio un vuelco a la clasificación a favor de las opciones de software libre, pero tras la publicación de la clasificación, tanto Microsoft como los proveedores del software libre (SuSE Linux AG y IBM Deutschland GmbH) realizaron nuevas ofertas que fueron consideradas en el estudio y las opciones fueron reevaluadas de acuerdo con las nuevas condiciones comerciales. Los resultados finales del estudio fueron publicados en julio de 2003.

La nueva propuesta de Microsoft incluía un compromiso para que la fijación de precios y condiciones contractuales durante los seis años del acuerdo no se viese influenciada por su posición de dominio en determinados segmentos del mercado del software. La cooperación se extendía, además, con un compromiso de Microsoft para desarrollar estándares para la administración pública de la ciudad en los productos de la compañía, y respuestas concretas a los requerimientos municipales en materia de seguridad. Aunque la propuesta no incluía descuentos directos, suponía ahorros de varios millones de euros. Estas propuestas elevaron notablemente las puntuaciones de Microsoft, tanto en los aspectos cuantitativos como cualitativos.

IBM y SuSE también modificaron sus propuestas iniciales, ofreciendo asesoramiento gratuito durante el proceso de migración. SuSE ofreció un emulador de PC alternativo a un precio menor. También se ofrecieron mejoras en OpenOffice, lo que permitió ganar algunos puntos adicionales a las opciones de software libre.

Una vez evaluadas las nuevas propuestas, la opción XP/XP volvió a liderar la clasificación conjunta (criterios de coste y cualitativos) de las opciones de migración. Sin embargo, las autoridades municipales optaron por la migración a un sistema con Linux y OpenOffice. Cabe cuestionarse, en este caso, cuál fue la utilidad del estudio realizado si al final las decisiones se adoptaron de forma arbitraria. Además del coste adicional de la opción elegida habría que añadir el propio coste del estudio en el *debe* de la decisión de las autoridades munitenses, ya que se adoptó dejando al margen el hecho incuestionable de que los habitantes de dicha ciudad iban a pagar varios millones de euros más por una combinación de productos de software que, incluso aceptando la cuestionable batería de indicadores cualitativos, quedó peor puntuada que la opción de software propietario.

**Tabla 4.3. Estudio de Unilog para la ciudad de Munich.**

Compración de costes de migración, millones \$

Alternativas	Coste
MS XP + MS Office XP	34,18
MS XP + Open Office	39,75
GNU/Linux + Open Office	45,77
GNU/Linux + Open Office + PC emulation*	35,94
GNU/Linux + Open Office + Terminal Server	50,00

\* (WINE/WMWare)

Fuente: enter, a partir de Unilog (2003)

## Reino Unido

El Reino Unido dio a conocer su política sobre el uso de software libre en la administración pública en 2002. La posición británica es eminentemente pragmática y su política desde esa fecha se articula en términos de una serie de principios generales:

- En los procesos de adquisición de productos de TI, se tendrán en cuenta tanto las alternativas de software libre como las de software comercial. La asignación de contratos se producirá de acuerdo con un análisis de los costes y beneficios de cada alternativa propuesta (*value for money*).
- En los futuros desarrollos y aplicaciones, se emplearán exclusivamente productos basados en estándares abiertos que permitan la interoperabilidad<sup>4</sup>. Del mismo modo, se considerará la posibilidad de obtener plenos derechos sobre el código fuente de los desarrollos realizados a medida y de las personalizaciones de paquetes comerciales contratados por la administración.

Alguno de los principios, sin embargo, no acaban de encajar bien en la tendencia de guiar las decisiones respecto al software de acuerdo con la neutralidad tecnológica y el análisis de costes y beneficios. Así, se introducen factores cualitativos difíciles de medir y potencialmente generadores de arbitrariedad, cuando se señala que se intentará evitar el *exceso de dependencia* con respecto a productos y servicios comerciales.

Pero este riesgo queda amortiguado cuando se tienen en cuenta las justificaciones esgrimidas por las autoridades británicas para

sustentar esta estrategia general en relación con el software.

Estas justificaciones se agrupan en cuatro argumentos:

- En primer lugar, establecer la eficiencia como principio rector de las compras de software en la administración, lo que supone elegir la mejor opción para cada situación, con independencia de si se trata de software libre, comercial o de una combinación híbrida de ambos.
- En segundo lugar, el logro de la interoperabilidad es esencial para asegurar la prestación de servicios públicos.
- También se considera deseable adoptar medidas que impliquen una reducción de costes y riesgos en relación con los sistemas informáticos de la administración. Este argumento da sustento al principio *value for money*.
- La última justificación hace referencia a la seguridad de los sistemas informáticos de la administración, cuestión que se considera vital y sobre la que juzga que las posiciones del software libre y el comercial están a la par. Sobre esta cuestión cabe destacar una conclusión a la que se llegó en un estudio técnico encargado por el gobierno británico a la consultora QinetiQ (véase Peeling, N. y J. Satchell, 2001), en el que se señala que las diferencias entre el software libre y el comercial no constituyen factores relevantes en la mejora o deterioro de la seguridad de las infraestructuras nacionales de tecnologías de la información.

<sup>4</sup> El empleo de estándares abiertos no implica necesariamente la utilización de software libre. El caso británico es un exponente de este hecho, ya que incluso en este contexto se recurre al software comercial. Este hecho pone de manifiesto, tal y como señalan Padilla, J. et al. (2004), que *'el software comercial es perfectamente compatible con la adopción por parte de la Administración Pública de estándares abiertos para el intercambio y almacenamiento de información.'*

## Dinamarca

El caso de Dinamarca es buen ejemplo de un país que ha dejado de lado posiciones *ideologizadas* respecto a la elección de productos de software en el seno de la administración y ha optado por una aproximación pragmática, basada fundamentalmente en los criterios de *Total Cost of Ownership* y de neutralidad tecnológica. Esta posición encierra una gran coherencia y un empleo responsable de los recursos de los contribuyentes.

En junio de 2003, Dinamarca puso en marcha una estrategia nacional en materia de software, dirigida por el Ministerio de Ciencia, Tecnología e Innovación. El objetivo principal de la citada estrategia se basaba en el estímulo de la competencia, la calidad de los servicios y la coherencia en las soluciones de software adoptadas en el sector público. Para el logro de los objetivos, la autoridad definió cuatro principios fundamentales que debían tenerse presentes a la hora de diseñar la estrategia:

- En primer lugar, lograr el máximo valor a cambio del dinero gastado (*maximum value for money*), con independencia del tipo de solución de software adoptada. De forma más concreta, se indicaba que cada institución individual debía adoptar la solución de software que proporcionase mayor valor en términos de mérito y de las necesidades concretas de las actividades locales, con independencia de si estas soluciones implicaban el uso de software libre o propietario.
- El segundo principio era el de competencia, independencia y libertad de elección. La competencia se entendía como un prerrequisito para el funcionamiento eficiente de un mercado de software, en el que todos los participantes en el sector debían ser capaces de ofrecer sus

productos a la administración en igualdad de condiciones. En este sentido, simplemente se abogaba por la eliminación de barreras a la libre competencia.

- El tercer principio era el de la flexibilidad y la interoperabilidad, en el sentido de que debía primarse aquel tipo de software que facilitase al máximo la interconexión con otros productos y tipos de software, así como entre instituciones, como una vía para mejorar la flexibilidad.
- Por último, las autoridades danesas abogaban por el fomento de la innovación en el ámbito del software, por lo que se consideraba que se debía ser receptivo en lo que respecta a nuevos modelos de adquisición y desarrollo de software.

A raíz de esta estrategia, Dinamarca ha adoptado una aproximación pragmática a la cuestión del papel que debe desempeñar el software libre en el seno de la administración. En este sentido, se ha determinado que ni el software libre ni el propietario deben ser opciones preferentes, sino que la elección dependerá de cuál de ellos proporcione el mayor valor para la organización en cada situación concreta.

Asimismo, el gobierno danés anunció un conjunto de iniciativas destinado a poner en marcha su política en materia de uso y adquisición de software, entre las que destacan:

- Desarrollo de un modelo de *Total Cost of Ownership* (TCO) que cubra todos los costes de establecimiento, soporte y gestión relacionados con las migraciones hacia nuevas soluciones de software, que debe ser utilizado en todas las administraciones públicas. Este modelo constituye un elemento fundamental para poder tomar decisiones justificadas desde un punto de vista económico

sobre el tipo de software que debe emplearse en la administración, como ya se ha visto con anterioridad.

- Desarrollo de programas piloto que permitan evaluar las ventajas e inconvenientes de las distintas opciones disponibles, tanto en el terreno del software comercial como del software libre.
- Vigilancia de la utilización de estándares abiertos, incluyendo XML, W3C y aquellos que facilitan la accesibilidad de los impedidos.
- Seguimiento del desarrollo de las soluciones de firma electrónica basadas en software libre.
- Estímulo a la difusión de información a través de una nueva web que incremente el conocimiento y la transparencia del mercado de software, a través de la centralización de todos los resultados y lecciones aprendidas de las distintas iniciativas, así como publicación de in-

formación sobre proyectos nacionales e internacionales en la materia.

## Estados Unidos

En Estados Unidos, el debate sobre la forma en que debían evaluarse las estrategias alternativas de elección de software en el seno de las administraciones públicas resultó clarificado a partir de la publicación en 2001 del Informe Mitre<sup>5</sup>. De acuerdo con éste, la decisión entre software libre o comercial debe estar fundamentada en torno a tres pilares:

- Los costes directos, entre los que se entienden incluidos todos aquellos ligados al ciclo de vida del software: licencias y actualizaciones, necesidades de hardware, costes de instalación, puesta en marcha y mantenimiento, gestión y desmantelamiento.
- Los costes indirectos, entre los que se incluyen el tiempo destinado al sopor-

### La postura de la Comisión Europea

La posición de la Comisión Europea (CE) en relación con las directrices relacionadas con las migraciones hacia sistemas informáticos basados en el software libre quedó reflejada en un informe de octubre de 2003, publicado por IDA, organismo dependiente encargado de coordinar el desarrollo de sistemas de comunicación entre las administraciones públicas.

En dicho documento se considera que el criterio de *value for money* debe ser fundamental en cualquier decisión relacionada con la adquisición de productos de software por parte de la administración. De acuerdo con IDA, las decisiones que se adopten deben venir precedidas de un estudio que tenga en cuenta:

- El coste del sistema informático cuya adopción se analiza a lo largo de un período de tiempo suficientemente largo, al menos cinco años.
- El coste de los sistemas alternativos en el mismo período de tiempo, teniendo en cuenta de forma específica los costes de migración.
- Un análisis de las fortalezas y debilidades de cada una de las alternativas planteadas.

Fuente: Padilla, J. et al. (2004)

<sup>5</sup> Mitre Corp. es una entidad financiada por el Departamento de Defensa cuya labor es la de prestar servicios de ingeniería de sistemas, I+D y soporte en tecnologías de la información al gobierno de Estados Unidos.

te técnico, la posible falta de disponibilidad del sistema, etc. Estos costes pueden ser especialmente significativos en el caso del software libre y afectan directamente al grado de productividad del usuario final.

- Los beneficios y riesgos de cada opción estudiada, entre los que se incluyen aspectos cualitativos como fiabilidad, interoperabilidad, escalabilidad, seguridad, facilidad de manejo, disponibilidad de aplicaciones compatibles, la posibilidad de personalización, etc.

Finalmente, aunque el Informe Mitre admitía la posibilidad de que el software libre desempeñe un papel importante como alternativa al propietario en el seno de las administraciones públicas, terminaba afirmando que era erróneo determinar a priori y con carácter general una superioridad de dicha opción.

## Nueva Zelanda

Nueva Zelanda también ha clarificado en los últimos años su posición respecto a las adquisiciones de software en el seno de la administración pública. En marzo de 2003 el ministerio de Servicios Estatales, a través de su unidad de E-government, realizó una serie de recomendaciones respecto al uso del software libre:

- El software de código abierto es con frecuencia una posible alternativa viable al software comercial, cada vez más empleada en los ámbitos público y privado.
- Los principios que deben guiar cualquier decisión sobre el software que se utilice en la administración pública deben ser *value for money* y adecuación para un propósito (*fitness for purpose*).

## 4.4. Algunos ejemplos del posicionamiento de España

España es uno de los países donde más ha calado la euforia por el software libre y donde de forma menos objetiva se han tenido en cuenta los criterios de comparación entre aquél y las alternativas de software comercial. Sin una justificación muy clara, en numerosas ocasiones se ha extendido la idea de que el software libre era superior al propietario y era necesario promocionarlo desde las administraciones públicas. Las iniciativas en este sentido se han desarrollado tanto en el ámbito regional, donde los gobiernos de algunas comunidades autónomas han promovido medidas encaminadas a fomentar el uso del software libre, como en el ámbito nacional, donde se han presentado y debatido iniciativas parlamentarias con el mismo fin.

En la escala regional se ha producido una polarización de las posiciones respecto al software libre. Algunas comunidades, como Asturias, Castilla y León, Madrid o Navarra han rechazado propuestas favorables a la adopción de software libre por la administración, que se han debatido en sus respectivos parlamentos. Otras, sin embargo, lideradas sobre todo por Extremadura, se han

## CENATIC: Dinero público para la promoción interesada del software libre

El 1 de julio de 2005 el Gobierno encargó al Ministerio de Industria, Turismo y Comercio la creación del Centro Nacional de Referencia de Aplicación de las Tecnologías de Información y Comunicación (CENATIC). La sede del centro se situó en Almendralejo (Badajoz).

El objetivo de CENATIC es la promoción del *'conocimiento del software basado en fuentes abiertas en la Administración y en los diferentes sectores de actividad y generar un polo de atracción de iniciativas empresariales que acabe consolidando en Extremadura un cluster de empresas del sector de las NTIC.'*

La puesta en marcha del centro se realizará mediante una Fundación promovida por el propio ministerio que pueda impulsar sus objetivos y que integrará la participación de los diversos estamentos de la administración pública, la empresa privada y la universidad.

Este centro tiene previsto realizar inversiones por un total de 2,29 millones de euros en el periodo 2007-2009 para impulsar el desarrollo de software libre en España. Adicionalmente, en el periodo 2008-2009, tiene previsto crear un centro de referencia de formación y certificación de recursos humanos especializados, actuaciones de consultoría técnica y legal, asesoramiento técnico y legal a administraciones y empresas y un laboratorio de certificación de productos, certificación de la compatibilidad de productos y tecnologías con sistemas basados en fuentes abiertas.

Resulta imposible encontrar en la web del ministerio ni en la del nuevo centro una argumentación sobre la bondad relativa del software libre en relación con el propietario que justifique la financiación pública, utilizando impuestos de los ciudadanos en una iniciativa de estas características.

mostrado bastante favorables a la adopción de software libre en los organismos públicos. Entre ellas, Andalucía y Comunidad Valenciana.

En Extremadura se ha desarrollado un sistema operativo propio basado en software libre, el Linex, que se distribuye de forma gratuita y se ha adoptado de forma generalizada en el sistema educativo y los centros públicos de enseñanza en dicha comunidad. Además, en noviembre de 2006 se creó un centro de desarrollo de software libre financiado con dinero público, el Centro Nacional de Aplicación de las Tecnologías de la Información y Comunicación (CENATIC), radicado en esa comunidad autónoma.

En el ámbito nacional se han presentado varias propuestas en los últimos años para impulsar el software libre dentro de la administración. En diciembre de 2004, Esquerra Republicana de Cataluña (ERC) presentó en el Congreso de los Diputados una Proposición de Ley de Medidas para la Implantación del Software Libre en la Administración del Estado, mientras que en febrero de 2005 Izquier-

da Unida registraba, a su vez, otra de índole similar. En ambos casos, el Pleno del Congreso rechazó su toma en consideración.

Posteriormente, en julio de 2006, el Grupo Parlamentario Socialista en el Congreso presentó una Proposición no de Ley relativa a la promoción del software libre. Esta iniciativa, que fue debatida y aprobada por el Pleno del Congreso en diciembre de ese año, defiende la adopción y la promoción del software libre en y desde la administración a partir de una serie de argumentos desarrollados en su exposición de motivos que, o bien son directamente falsos, o carecen de evidencia empírica que los sustente. Así, se considera que la utilización de software libre genera ventajas para la sociedad, en comparación con el software propietario debido a:

- El software libre es más barato: ya se ha demostrado en la sección tercera del presente documento que no es necesariamente así y que los estudios realizados que han tenido en cuenta un horizonte de medio plazo han arrojado evidencia más bien en el sentido con-

trario. Se ignora además que la empresa vendedora de software propietario también forma parte de la sociedad, por lo que los recursos que se dirigen hacia ella siguen manteniéndose dentro de la misma (la empresa de software comercial también genera puestos de trabajo, paga salarios, etc.).

- El uso de software libre fomenta la innovación tecnológica, lo que choca con la

evidencia de que gran parte de los desarrollos de software libre son réplicas de productos comerciales, con funcionalidades similares a las del software propietario, en el mejor de los casos.

- Al emplear software libre se gana independencia de un proveedor comercial. Esto es cierto, pero no se tiene en cuenta el mayor riesgo de que desaparezca la empresa que desarrolló el software libre

### **Texto de la Proposición no de Ley aprobada por el Congreso de los Diputados, aprobada el martes 12 de diciembre de 2006 (\*)**

«El Congreso de los Diputados insta al Gobierno a profundizar, incidir y poner en marcha las siguientes políticas:

1. Actuar desde el principio general de generar libertad de opción y elección entre la ciudadanía.
2. El Gobierno aplicará los criterios de idoneidad, seguridad e interoperabilidad tecnológica en el momento de adquirir software, valorando la oferta global, según lo previsto en los marcos jurídicos relativos a la tramitación electrónica del procedimiento administrativo y a la contratación pública y, en cualquier caso, de racionalidad técnica y económica, con el nivel de madurez y coherencia adecuado al previsible impacto y con el respeto al marco comunitario establecido para la contratación.
3. Fomentar y garantizar el impulso y adopción de estándares abiertos desde la Administración del Estado.
4. Profundizar en las políticas de I+D+i que favorezcan la industria y el desarrollo de software libre y de código abierto en España, que fomenten la adopción y la creación de estándares abiertos de software, y que promuevan la generación de soluciones que sólo existen bajo la forma de software propietario en el caso de aplicaciones destinadas al uso de la ciudadanía.
5. Promover la ejecución en toda la Administración General del Estado de lo previsto en los “criterios de seguridad, normalización y conservación de las aplicaciones utilizadas para el ejercicio de potestades” y en la “propuesta de recomendaciones a la Administración General del Estado sobre software libre y de fuentes abiertas” de mayo de 2005 y, en concreto, promover el impulso de lo previsto en la mencionada propuesta, para mejorar la racionalidad técnica y económica en las compras públicas.
6. El Gobierno tendrá muy en cuenta los criterios de la Comunicación de la Comisión Europea al Consejo y al Parlamento Europeo, sobre “Interoperabilidad para unos servicios de Administración electrónica paneuropeos”, de 13 de febrero de 2006.
7. Introducir el criterio de que en todo concurso público o compra de aplicaciones o desarrollos a medida por parte de la Administración Pública se deben evaluar tanto las soluciones de software libre como las de software propietario, en caso de existir, bajo los criterios de coste, funcionalidad, seguridad e interoperabilidad, evitando recomendaciones o preferencias sólo en función del tipo de licencia de la solución.
8. Los programas y aplicaciones de la Administración General del Estado o compartidas con las Comunidades Autónomas destinadas a la ciudadanía se distribuirán en las diferentes lenguas del Estado. Igualmente, se promoverá la existencia de aplicaciones en las diversas lenguas del Estado.
9. Basar los planes de formación de toda la Administración del Estado, tanto los dirigidos a su personal como a la ciudadanía, en fomentar una formación orientada a la función, más que a la creación de meros usuarios de productos concretos.»

(\*) Esta Proposición no de Ley fue aprobada con el voto favorable de 299 diputados, una abstención y un voto en contra.

específico, dada la solidez de las grandes empresas de software comercial.

- El software libre es más seguro y garantiza mejor la privacidad de los datos. Pensar que por tener acceso al código fuente es más fácil revisar y detectar la introducción de ‘código malicioso o espía’ ignora el hecho de que se pone a disposición de cualquier usuario la posibilidad de alterarlo, por lo que no es nada claro que el nivel de seguridad vaya a ser mayor.
- El empleo de software libre facilita el fomento de las lenguas propias. Sin duda esto garantiza un incremento del coste; lo que queda por demostrar es si genera beneficio social.

En cuanto a las recomendaciones realizadas por el Congreso al Gobierno tratan, por una parte, de definir unas líneas generales de una estrategia de adopción de software por parte del Ejecutivo y, por otra, fijar una estrategia de promoción del software libre.

En el primero de los objetivos parece chocante y puede entrar en conflicto que se re-

comiende que toda elección de software se supedite a un análisis basado en ‘criterios de coste, funcionalidad, seguridad e interoperabilidad’ con el hecho de que se solicite ‘garantizar el impulso y adopción de estándares abiertos desde la Administración del Estado’.

En el segundo aspecto (punto 4 de la Proposición no de Ley) se genera una contradicción evidente. Resulta cuestionable que se recomiende la utilización de dinero público para financiar proyectos de software libre cuya finalidad es generar aplicaciones y productos sustitutivos de otros ya existentes, desarrollados por empresas privadas (software comercial). Es decir, se está proponiendo financiar desde el Gobierno una actividad paralela que compita con la que desarrolla una empresa privada en régimen de competencia. En cualquier otro sector, este hecho sería inmediatamente denunciado a las autoridades de la competencia, tanto nacionales como europeas.

Por otra parte, ¿cómo es posible que se defienda el software libre aludiendo a la innovación y luego se financie con dinero público la réplica de productos comerciales?

# Conclusiones

---

**El software es un tipo de bien peculiar.** Tiene, sin duda, características propias de los bienes basados en la información, pero también de los bienes materiales. Fruto de esta dualidad, se producen combinaciones inusuales de propiedades de ambos tipos de bienes que confluyen en el software.

Una muestra de este hecho es que los productos de software, al igual que la información, pueden ser replicados con unos costes marginales próximos a cero. Pero, al mismo tiempo, los productos de software y otras soluciones basadas en el software exhiben un grado de funcionalidad de la que carecen los bienes basados puramente en la información. Así, mientras la información es valorada por su capacidad para *informar* o influir, el software es valorado por lo que es capaz de hacer él mismo. Por esta razón se ha llegado a afirmar que el software se valora por su *comportamiento*. En este sentido, se parecería a la maquinaria, arquetipo de los bienes físicos.

Sin embargo, el software puede distinguirse del resto de productos industriales en una serie de dimensiones, incluso en el caso de las industrias altamente tecnológicas. En primer lugar,

la inversión en I+D necesaria para generar el software se realiza fundamentalmente en la fase de desarrollo, mientras que los costes de producción son prácticamente nulos, por lo que la oferta de productos de software está sujeta a importantes economías de escala.

Por otra parte, la demanda de estos productos se configura en un contexto en el que las externalidades de red desempeñan un papel fundamental, que se ha visto reforzado con Internet y el canal electrónico de transmisión provisto por la Red.

Como consecuencia de todas estas peculiaridades, tanto del producto en sí como de su demanda y su oferta, ha surgido un mercado de software también peculiar, en el que conviven dos *modelos de negocio* claramente diferenciados, el modelo del software propietario comercial y el modelo de software libre.

Ambos modelos presentan ventajas específicas al tiempo que inconvenientes, pero en general se ha constatado que existe una serie de mitos en torno al software libre, como que se trata de una opción más barata, más segura y fiable que la representada por el software propietario, no se sustentan cuando se enfrentan a la realidad. Numerosos estudios han puesto de manifiesto que tanto a nivel micro como macroeconómico el software propietario ofrece resultados generalmente superiores a los del software libre, sobre todo porque el primero alinea los intereses de productores y consumidores.

La reacción de los defensores del software libre ha sido, cuanto menos, curiosa. La forma elegida para alcanzar mayor implantación ha sido la de denunciar la existencia de fallos de mercado que provocan una gran concentración en este sector (un argumento que los datos no apoyan) en vez de centrarse en atender las necesidades y deseos de los usuarios finales, los clientes. Como consecuencia, se ha actuado a modo de lobby buscando una protección pública bajo la cual puedan desarrollar su expansión. Sin embargo, no existen fallos de mercado en el sector del software que justifiquen una intervención en defensa, protección y promoción del software libre por parte de las administraciones públicas. En los casos en los que se ha actuado de esa forma se han empleado argumentos que, o bien son dudosos o bien directamente falsos.

Numerosos países y administraciones han percibido este hecho, por lo que han adoptado unos criterios objetivos a la hora de adoptar y legislar sobre los productos de software. En este sentido, se ha desarrollado un amplio consenso en torno a la bondad de la adopción de dos principios.

En primer lugar, el análisis de costes y beneficios. Los recursos utilizados por las administraciones públicas proceden de los impuestos de los ciudadanos y sus rectores deben asumir la responsabilidad de emplearlos de forma eficiente. Por tanto, en tanto que las administraciones son usuarias de software, deben tomar sus decisiones de elección de productos de acuerdo con un principio

---

de racionalidad económica, en la que un análisis riguroso de los costes y beneficios de cada opción sea el criterio fundamental de decisión. Este principio ha sido adoptado como fundamental por gobiernos de países como Alemania, Estados Unidos, Reino Unido o Dinamarca, entre otros.

A la hora de valorar los costes de las distintas alternativas de software es necesario plantear un escenario de medio y largo plazo, puesto que las estructuras de coste de las alternativas son muy variadas. En última instancia se persigue determinar cuál es la opción que mejor se ajusta a las necesidades con un menor coste. Se trata, por tanto, de hacer el mejor uso posible de los recursos destinados a cubrir las necesidades de software. Este principio también es denominado *value for money*.

Relacionado con ese principio está el de la neutralidad tecnológica. En un análisis riguroso de costes y beneficios no cabe ningún planteamiento, de naturaleza ideológica o de otro tipo, que deje al margen a ninguna opción de software derivada de su naturaleza (libre o propietario). Todas las opciones han de ser tenidas en cuenta y valoradas de acuerdo con un mismo rasero. Debe adoptarse, por tanto, una actitud pragmática ante el software, lo que implica una posición de *neutralidad* respecto a la tecnología estudiada.

Pero el principio de neutralidad tecnológica no sólo debe ser tenido en cuenta por la administración pública en su condición de usuario de productos de software. También, y de manera más importante, debe pesar en su papel como generadora de legislación, ya que en varios países existen presiones para que las autoridades aprueben medidas de protección y promoción de un determinado tipo de software, lo que supone una violación del principio de neutralidad tecnológica.

El consenso internacional, por tanto, ha definido *de facto* como buenas prácticas en relación con la adopción de software los principios de *value for money* y de neutralidad tecnológica. De acuerdo con esto, las administraciones públicas deben evitar dejarse llevar por el entusiasmo o las presiones de los defensores del software libre que buscan proteger una actividad deliberadamente desviada de los principios de mercado.

Sin embargo, en el pasado reciente se han aprobado y puesto en marcha iniciativas en España que buscan promover de forma exclusiva el software libre frente al resto de alternativas a partir de motivaciones de carácter ideológico y mal fundamentadas teórica y técnicamente. Esta preocupante línea de actuación se aleja del consenso de los países más avanzados



## Referencias bibliográficas

- Bote, V. (2006), *Microsoft, Apple y la burocracia europea*. Nota ENTER n° 11.
- Economides, N. (1996), 'The Economics of Networks', *International Journal of Industrial Organization*, vol. 14, no. 2.
- Economides, N. (2000), 'Durable Goods Monopoly with Network Externalities with Application to the PC Operating Systems Market', *Quarterly Journal of Electronic Commerce*, vol. 1, no. 3.
- Economides, N. (2006), 'Public Policy in Network Industries', en P. Buccirossi (ed), *Handbook of Antitrust Economics*, Cambridge, The MIT Press.
- Economides, N. y E. Katsamakas (2006), 'Two-Sided Competition of Proprietary vs. Open Source Technology Platforms and the Implications for the Software Industry', *Management Science*, vol. 52, no. 7.
- Economides, N. y E. Katsamakas (2006), 'Linux vs. Windows: A Comparison of Application and Platform Innovation Incentives for Open Source and Proprietary Software Platforms', en Bitzer, J. y P. J. H. Schröder (eds.) (2006), *The Economics of Open Source Software Development*. Elsevier B. V.
- EICTA (2004), *EICTA Interoperability White Paper*. 21 de junio.
- ENTER (2006), *Capital intelectual y competitividad: un reto decisivo*. Informes ENTER, noviembre.
- Evans, D. S., A. Hagiu y R. Schmalensee (2006), *Invisible Engines. How Software Platforms Drive Innovation and Transform Industries*. The MIT Press. Cambridge, Massachusetts.
- Gantz, J. F., A. Gillen y M. Warmerdam (2006a), *The Economic Impact of Microsoft Windows Vista*. IDC White Paper (September).
- Gantz, J. F., A. Gillen y M. Warmerdam (2006b), *The Economic Impact of Microsoft Windows Vista in the United States*. IDC White Paper (December).
- Goto, M., I. Kimura y H. Sakai (2002), 'Macroeconomic Impact of IT Adoption and Diffusion', *Japan Bank for International Cooperation Review*, n° 8, pp. 101-129.
- Government of the Hong Kong Special Administrative Region (2003), *Is open source software secure?*, texto utilizado disponible en la siguiente url: <http://www.info.gov.hk/digital21/eng/knowledge/open-source1.html>
- Hahn, R. W. (ed.) (2002), *Government Policy toward Open Source Software*, AEI-Brookings Joint Center for Regulatory Studies, Washington D. C.
- IDC (2002), *Windows 2000 Versus Linux in Enterprise Computing. An Assessment of Business Value for Selected Workloads*. IDC White Paper.
- Jorgenson, D. W. (2001), 'Information Technology and the U.S. Economy', *American Economic Review*, 90(1).
- Jorgenson, D. W. (2005), 'Accounting for Growth in the Information Age', en Aghion, P. y S. Durlauf, (eds.), *Handbook of Growth Economics*, Vol. 1A, Amsterdam, North-Holland, pp. 743-815.
- KBSt (2005), *Migration Guide. A guide to migrating the basic software components on server and workstation computers*. KBSt Publication Series, Vol. 72.
- Kooths, S., M. Langenfurth y N. Kalwey (2003), *Open Source Software. An Economic Assessment*. MICE Economic Research Studies, vol. 4.
- Mäkelä, M. M. (2005). *Software Business: Position as a Field of Research and Avenues for Scholarly Contributions*. 14<sup>th</sup> International Conference of the

- International Association for Management of Technology (IAMOT). Viena, Austria.
- Messerschmitt, D. G. y C. Szyperski (2000). *Industrial and Economic Properties of Software*.
- Messerschmitt, D. G. y C. Szyperski (2003). *Software Ecosystem. Understanding an Indispensable Technology and Industry*. Cambridge, Massachusetts: The MIT Press.
- Padilla, A. J., P. L. Sánchez, G. Lozano y E. Cañizares (2004), *El papel del sector público en la industria del software: comentarios a las iniciativas en materia de uso y promoción del 'software libre'*. NERA Economic Consulting, marzo.
- Peeling, N. y J. Satchell (2001), *Analysis of the Impact of Open Source Software*. QinetiQ Ltd.
- Smith, D. M., R. Simpson, M. A. Silver y L. Fiering (2003), *Linux on the Desktop: The Whole Story*. Gartner Research AV-20-6574.
- Schmidt, K. M. y M. Schnitzer (2003), 'Public Subsidies for Open Source? Some Economic Policy Issues of the Software Market'. *Harvard Journal of Law and Technology*, Vol. 16, No. 2
- Shy, O. (2001), *The Economics of Network Industries*. Cambridge University Press.
- Thompson, H. H. (2005), *Reliability: Analyzing Solution Uptime as Business Needs Change*. Security Innovation White Paper.
- Van Ark, B. y R. Inklaar (2005). *Catching Up or Getting Stuck? Europe's Troubles to Exploit ICT's Productivity Potential*. Groningen Growth and Development Centre. Research Memorandum GD-79.
- Van Reeden, J. y R. Sadun (2005), *Information Technology and Productivity: It ain't what you do it's the way that you do I.T.*, London School of Economics, Centre for Economic Performance Discussion Paper Series No. 002.



[www.enter.es](http://www.enter.es)

**enter**  **ie**