www.cartagena99.com no se hace responsable de la información contenida en el presente documento en virtud al
Artículo 17.1 de la Ley de Servicios de la Sociedad de la Información y de Comercio Electrónico, de 11 de julio de 2002.
Si la información contenida en el documento es ilícita o lesiona bienes o derechos de un tercero háganoslo saber y será retirada.

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70
- - -
ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

mas Distribuidos

Memoria compartida distribuida

(DSM, *Distributed Shared Memory*)

Índice

- Introducción
- Estrategias de implementación
  - Mediante memoria virtual (VM-DSM)
  - Por compilador y entorno en tiempo de ejecución (RT-DSM)
- Aspectos de diseño de DSM
  - Esquema de gestión y política de actualización de copias
  - Sincronización
  - Localización de copias
  - *False-sharing* y *thrashing*
- Modelos de coherencia de memoria
- DSM basada en espacios de tuplas

Sistemas Distribuidos 2 — Fernando Pérez Costoya

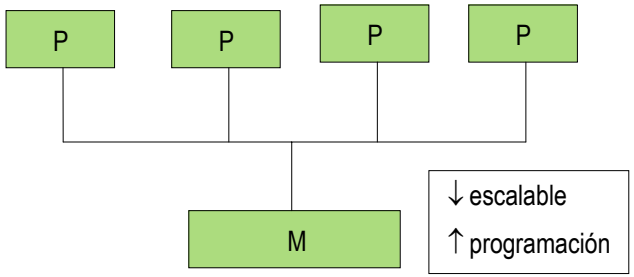
lasificación de sis. paralelos

ada *versus* distribuida
versus compartida

ada y compartida → multiprocesadores
da y privada → sistemas distribuidos
da y compartida → *Distributed Shared Memory* (DSM)
red Memory (DSM)
ware:
adores NUMA (acceso a memoria no uniforme)
vare → Objetivo de la presentación (DSM Software)
are en SD para ofrecer DSM

Fernando Pérez Costoya

Memoria centralizada y compartida

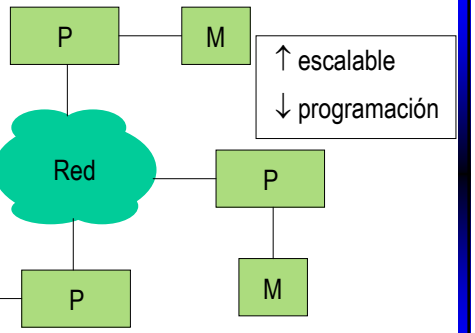


Sistemas Distribuidos 4 — Fernando Pérez Costoya

www.cartagena99.com no se hace responsable de la información contenida en el presente documento en virtud al
Artículo 17.1 de la Ley de Servicios de la Sociedad de la Información y de Comercio Electrónico, de 11 de julio de 2002.
Si la información contenida en el documento es ilícita o lesiona bienes o derechos de un tercero háganoslo saber y será retirada.

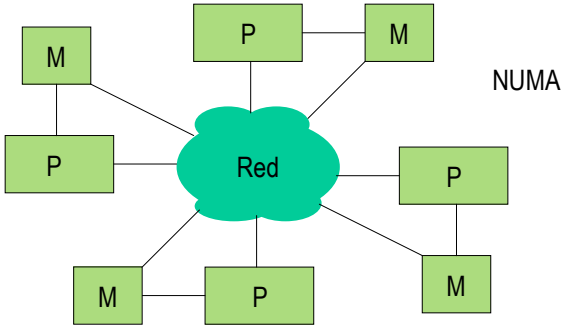
CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70
- - -
ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

oria distribuida y privada



Fernando Pérez Costoya

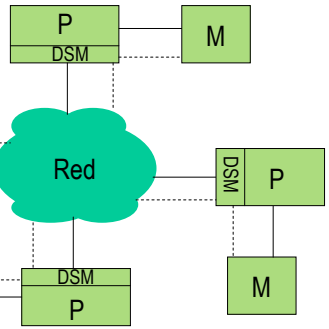
DSM HW: Memoria distribuida y compartida



Sistemas Distribuidos
6

Fernando Pérez Costoya

oria distribuida y compartida



Fernando Pérez Costoya

Introducción (1/2)

- Multiprocesadores con mem. compartida vs. s. distribuidos:
  - HW más complejo y difícilmente ampliable
  - SW más sencillo y mejor conocido
- Modelo de programación en sistemas con mem. compartida
  - Llamadas a procedimiento (o invocación de métodos)
  - Comunicación mediante datos compartidos
  - Sincronización mediante semáforos o mecanismos equivalentes
- Modelo de programación tradicional en s. distribuidos
  - Paso de mensajes para comunicación y sincronización
- Querría ejecutar en SD aplicación paralela basada en m.comp.
- Nuevo modelo de programación en s. distribuidos
  - RPC (o RMI) + DSM

Sistemas Distribuidos
8

Fernando Pérez Costoya

ntroducción (2/2)

artida distribuida (DSM)
artida implementada por software
iante paso de mensajes
con datos compartidos y sincronización con semáforos

es de construir y programar
adecuada para aplicaciones paralelas
oncurrente con estructuras de datos compartidas
ación para cliente-servidor
consumidor de servicios no deberían compartir memoria

ceptable requiere múltiples copias en distintos nodos
coherencia: Similar a multiprocesadores pero por SW
heterogeneidad

Fernando Pérez Costoya

Implementación VM-DSM

- Extensión de m. virtual que permite solicitar páginas remotas
  - Accesos a DSM convencionales (LOAD/STORE)
  - Fallo de página no local, se solicita a máquina remota
- Normalmente implementada en modo usuario
  - Se impide acceso a página no presente en máquina local (mprotect)
  - En tratamiento de excepción (SEGV), solicita página remota
  - Se "instala" la página en mapa de proceso con permiso de acceso
  - Proceso reanuda ejecución de forma transparente
- Características:
  - Dificulta tratamiento de heterogeneidad
  - No sobrecarga en accesos pero sí por tratamiento de fallos
  - *False sharing*
- Ejemplos:
  - Ivy, TreadMarks, Munin

Sistemas Distribuidos
10
Fernando Pérez Costoya

lementación RT-DSM

ompilador + biblioteca DSM en t. ejecución
mpartido, compilador traduce en LOAD/STORE
en llamada a DSM
ompartido, el compilador asigna espacio extra:
its de modificado y validez, nº de versión, *timestamp*, ...

ento de heterogeneidad
cada acceso
ng

Fernando Pérez Costoya

Gestión y actualización de copias

- Una sola copia fija o migratoria: mal rendimiento
  - Uso de múltiples copias
- Alternativas en la actualización de copias:
  - *Write-invalidate*:
    - Actualización genera invalidación de copias
    - Múltiples copias de lectura (*read-replication*)
      - Múltiples lectores/único escritor
    - Ivy, Treadmarks, Munin (configurable)
  - *Write-update*:
    - Actualización se propaga a copias
    - Múltiples copias tanto de lectura como de escritura (*full-replication*)
      - Múltiples lectores/múltiples escritores
    - Midway, Munin (configurable)
  - En ambas opciones puede usarse *multicast*

Sistemas Distribuidos
12
Fernando Pérez Costoya

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70
- - -
ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

www.cartagena99.com no se hace responsable de la información contenida en el presente documento en virtud al Artículo 17.1 de la Ley de Servicios de la Sociedad de la Información y de Comercio Electrónico, de 11 de julio de 2002. Si la información contenida en el documento es ilícita o lesiona bienes o derechos de un tercero háganoslo saber y será retirada.

ión y localización de copias

san mecanismos "tradicionales"
rreras, semáforos, ...
neficiente implementación basada en *TestAndSet*
cho tráfico de accesos e invalidaciones superfluas
n basada directamente en mensajes explícitos

copias
dcast
ble
he información sobre copias de todos los datos
otella", falta de *escalabilidad* y punto único de fallo
s: cada uno guarda info. sobre subconjunto de datos
a en páginas: Reparto dependiendo de número de página
adena de dueños probables

Fernando Pérez Costoya

False-sharing y thrashing

- *False sharing*
  - Puede ocurrir si tamaño(unidad de transferencia) > tamaño(dato)
    - Sí en VM-DSM; No en RT-DSM
    - Página puede contener datos independientes que se escriben a la vez
    - Se puede reducir con asignación "inteligente" de datos
      - Solución extrema: dato ocupa nº entero de páginas (fragmentación)
  - Mayor tráfico en la red
  - *Write-invalidate* invalidaciones innecesarias
  - *Write-update* posible sobrescritura errónea de datos
  - Uso de "twins" y "diffs" para paliar problemas (TreadMarks, Munin)
- *Thrashing*
  - Procesos compitiendo por dato (realmente o por *false-sharing*)
  - Página "viaja" continuamente entre procesadores ("ping-pong")
    - Suele existir un intervalo mínimo de permanencia en un nodo

Sistemas Distribuidos
14
Fernando Pérez Costoya

e coherencia de memoria

l s. memoria respecto a accesos de programa
s. memoria compartida HW o SW a las aplicaciones
uelve cada acceso de lectura

ogramador querría coherencia estricta:
ve valor de escritura "más reciente"

so en sistemas de memoria compartida
ue HW de memoria la asegure hay otros factores:
y compilador reordenan instrucciones
guarda variables en registros
moria de Java no es estricto

horia: compromiso coherencia y rendimiento
coherencia menos estricta → mayor rendimiento
izaciones (p.e. retardar propagación de cambios)
nodelo de programación menos intuitivo

Fernando Pérez Costoya

Modelos de coherencia uniformes

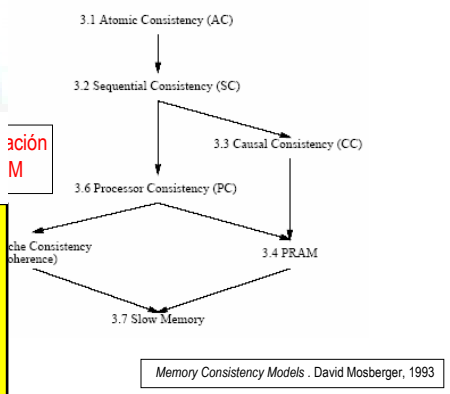
- Coherencia secuencial (Lamport 1979):
  - "Resultado equivalente a que los accesos a memoria de procesadores se ejecutaran en algún orden secuencial apareciendo las operaciones de cada uno en el orden especificado por su programa".
  - Definición similar a la "seriabilidad" de las transacciones
  - Todos los procesadores ven accesos a memoria en el mismo orden
    - Aunque puede ser ≠ del tiempo real exacto de emisión de los accesos
- Coherencia secuencial permite muy pocas optimizaciones
  - Escritura debe esperar hasta que se complete propagación
  - No escalable
- Uso de modelos de coherencia más "relajados" (véase figura)
- Modelos presentados hasta ahora son de tipo uniforme
  - Tratan igual todos los accesos a memoria
  - No suficientemente relajados para DSM

Sistemas Distribuidos
16
Fernando Pérez Costoya

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70
- - -
ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

www.cartagena99.com no se hace responsable de la información contenida en el presente documento en virtud al Artículo 17.1 de la Ley de Servicios de la Sociedad de la Información y de Comercio Electrónico, de 11 de julio de 2002. Si la información contenida en el documento es ilícita o lesiona bienes o derechos de un tercero háganoslo saber y será retirada.

coherencia más relajados



Memory Consistency Models . David Mosberger, 1993

Modelos de coherencia híbridos

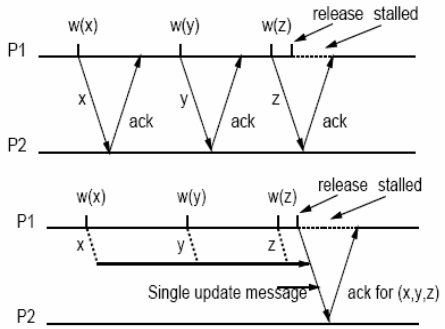
- Programa bien construido → no condiciones de carrera
  - Accesos conflictivos a datos deben usar sección crítica (SC)
- Modelos de coherencia híbridos
  - Distinguen operaciones de accesos a datos y de sincronización
  - Tienen en cuenta sincronización para mantener coherencia de datos
  - No se requiere propagar escrituras sobre datos inmediatamente
  - Necesario gestionar un *bit de modificado*
    - Por página en VM-DSM; Por variable en RT-DSM
  - Si no condiciones de carrera → equivalencia a c. secuencial
  - Accesos conflictivos sin sincronización: resultado impredecible
  - Varios modelos: de liberación, de entrada, de ámbito, ...
  - Distinguir dos tipos de operaciones de sincronización:
    - Entrada en SC (*acquire*)
    - Salida de SC (*release*)

encia de liberación (RC)

ción
mbios locales

alizados durante SC se propagan a copias
(si *write-invalidate*) o se envían cambios (si *write-update*)
os multiprocesadores m. compartida (DASH)
validate
en Munin (VM DSM)
i *write-invalidate* o si *write-update*
r cada página modificada durante SC
direcciones de máquinas con copia
ualiza las copias

Propagación de cambios con RC



Design of the Munin Distributed Shared Memory System. John B. Carter

www.cartagena99.com no se hace responsable de la información contenida en el presente documento en virtud al
Artículo 17.1 de la Ley de Servicios de la Sociedad de la Información y de Comercio Electrónico, de 11 de julio de 2002.
Si la información contenida en el documento es ilícita o lesiona bienes o derechos de un tercero háganoslo saber y será retirada.

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70
- - -
ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

iberación perezosa (LRC)

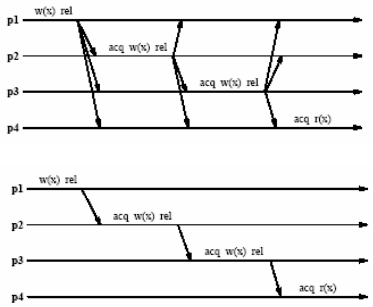
do "impaciente" (*eager* → ERC)
ciada a cerrojo C propaga cambios a todas las copias
que adquiera C necesita ver cambios inmediatamente

iberación perezosa (LRC)
ambios hasta que otro proceso los necesite
C: pide cambios a último proceso Q que tuvo C
cambios hechos por Q y previos que tengan relación causal
n de SC sobre un cerrojo establecen un orden causal

M DSM) sigue este modelo (con invalidación)
de relojes lógicos para mantener causalidad de cambios
C, mensaje a Q con vec. relojes de su última SC con C
or calcula qué cambios no ha visto todavía P
n lista de páginas a invalidar

Fernando Pérez Costoya

ERC vs. LRC

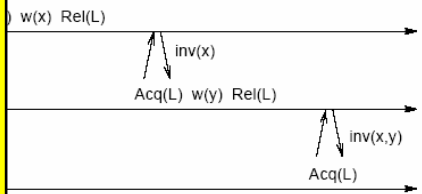


Lazy Release Consistency for Distributed Shared Memory.
Tesis de Peter Keleher, 1995

Sistemas Distribuidos
22

Fernando Pérez Costoya

den causal en LRC



Distributed Shared Memory.
Keleher, 1995

Fernando Pérez Costoya

Coherencia de entrada (EC)

- LRC en salir SC requiere revisar todo el espacio compartido
- Coherencia de entrada (EC)
  - Todo dato compartido está asociado al menos a una v. sincro (cerrojo)
  - P entra SC de C: pide cambios a último proceso Q que tuvo C
    - P debe ver cambios hechos por Q y previos que tengan relación causal
    - Pero sólo cambios de los datos asociados a C
- LRC versus EC
  - EC puede ser más eficiente pero requiere asociar variables y cerrojos
- Midway (RT DSM) sigue este modelo
  - Usa relojes lógicos escalares para mantener causalidad

Sistemas Distribuidos
24

Fernando Pérez Costoya

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70
- - -
ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

www.cartagena99.com no se hace responsable de la información contenida en el presente documento en virtud al Artículo 17.1 de la Ley de Servicios de la Sociedad de la Información y de Comercio Electrónico, de 11 de julio de 2002. Si la información contenida en el documento es ilícita o lesiona bienes o derechos de un tercero háganoslo saber y será retirada.

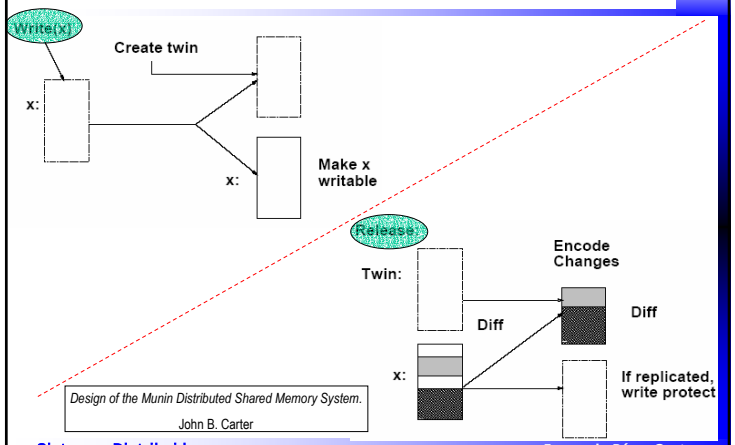
www.cartagena99.com no se hace responsable de la información contenida en el presente documento en virtud al
Artículo 17.1 de la Ley de Servicios de la Sociedad de la Información y de Comercio Electrónico, de 11 de julio de 2002.
Si la información contenida en el documento es ilícita o lesiona bienes o derechos de un tercero háganoslo saber y será retirada.

y en VM-DSM con m. híbridos

- en acceder a datos disjuntos en misma página
- SC entre ellos ya que no hay condición de carrera
- pias se sobrescriben cambios
- " y "diffs"
  - dificar la página
  - duce una excepción (STORE)
  - a copia de trabajo (*twin*)
  - nin) o cuando otro proc. pida cambios (ThreadMarks)
  - iferencias (*diff*)
  - (mediante invalidación o actualización) sólo *diff*
  - mezcla *diffs* con su copia de la página
  - ma de sobrescritura y reduce tráfico
  - pias y cálculo de diferencias

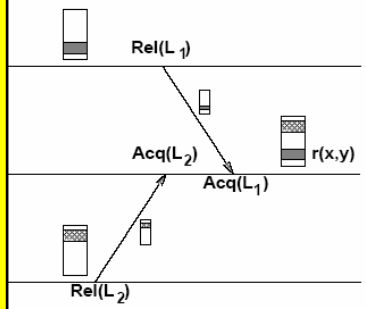
Fernando Pérez Costoya

Twins y Diffs en Munin



Sistemas Distribuidos
26
Fernando Pérez Costoya

critores sobre misma página



Distributed Shared Memory.
Keleher, 1995

Fernando Pérez Costoya

M. Compartida vs. Paso de mensajes

- M. compartida → comunicación desacoplada:
  - Generador dato (*G*) y consumidor (*C*) totalmente desacoplados:
    - En nombrado: *G* y *C* no se conocen; sólo comparten nombre de dato
    - En el tiempo: no necesitan coincidir
    - Si M. Compartida respaldo en disco → comunicación persistente
- Paso de mensajes → comunicación acoplada:
  - *G* y *C* deben conocerse y coincidir en el tiempo
  - No en el caso de sistemas de colas de mensajes (MOM)
    - Precisamente ese es punto fuerte de MOM
- P. mensajes ofrece sincronización implícita en comunicación:
  - Consumidor se bloquea hasta que se genere dato
- P. mensajes puede manejar heterogeneidad (MPI)
- ¿M. compartida con sincro implícita y manejo heterogeneidad?
  - Espacio de tuplas

Sistemas Distribuidos
28
Fernando Pérez Costoya

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70
- - -
ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

M con espacios de tuplas

Ahuja, Carriero y Gelernter (≈1985)
del lenguaje, SO y HW (M. compartida y SD)
stá basado en esta idea

ompartida en Linda: espacio de tuplas "vivo"
+ sincronización + procesamiento

a de campos con tipo *("hola", 7,3.14, "adios")*
na independiente a sus creadores
ociativamente

a comunicación y creación de procesos:
de carácter pasivo (base de la DSM)
so: de carácter activo (creación de procesos)
ión en alguno de los campos
se genera proceso que ejecuta función
tupla de datos donde resultado función sustituye a función

Fernando Pérez Costoya

Linda: Operaciones sobre tuplas

- *out(tupla)*: Añade *tupla* al espacio de tuplas; no bloqueante
- *in(patrón)*: Busca tupla que encaje en *patrón* y la extrae
  - si varias → cualquiera; si ninguna → se bloquea
  - Ejemplo: *in("hola", ?i, ?f, "adios")* → *("hola", 7, 3.14, "adios")*
  - *inp*: versión no bloqueante
- *rd(patrón)*: Igual que *in* pero la tupla no se extrae
  - *rdp*: versión no bloqueante
- *eval(tupla)*: Añade *tupla* de proceso al espacio de tuplas
  - Al insertar se activa proceso que la convertirá en tupla de datos
- Extensión a propuesta original:
  - Múltiples espacio de tuplas
    - Con operaciones para mover/copiar tuplas entre espacios

Sistemas Distribuidos
30
Fernando Pérez Costoya

da: Ejemplos de uso

o: *in("sem")*
o: *out("sem")*
:

```
lice);

 indice, ?pet);

sta",indice++,res);
```

```
cliente(){
   int indice;
   . . . . . . . . .
   in("turno", ?indice);
   out("turno", indice+1);
   out("peticion", indice, pet);
   in("respuesta",indice,?res);
}
```

Fernando Pérez Costoya

Ejemplos vectores y matrices en Linda

- Iniciar matriz "M" (MxN) con diagonal a 1 y resto a 0

```
for (i=1; i<=M; i++)
   for (j=1; j<=N; j++)
      out("M", i, j, (i==j));
```

- Cálculo vector "V" suma de filas de matriz "M" (MxN)

```
for (i=1; i<=M; i++) {
   suma = 0;
   for (j=1; j<=N; j++) {
      rd("M", i,  j, ? v);
      suma += v;
   }
   out("V", i, suma);
}
```

Sistemas Distribuidos
32
Fernando Pérez Costoya

www.cartagena99.com no se hace responsable de la información contenida en el presente documento en virtud al
Artículo 17.1 de la Ley de Servicios de la Sociedad de la Información y de Comercio Electrónico, de 11 de julio de 2002.
Si la información contenida en el documento es ilícita o lesiona bienes o derechos de un tercero háganoslo saber y será retirada.

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70
- - -
ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70
- - -
ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

www.cartagena99.com no se hace responsable de la información contenida en el presente documento en virtud al Artículo 17.1 de la Ley de Servicios de la Sociedad de la Información y de Comercio Electrónico, de 11 de julio de 2002. Si la información contenida en el documento es ilícita o lesiona bienes o derechos de un tercero háganoslo saber y será retirada.