estudios de postgrado en computación

Bases de datos avanzadas

Universidad de Los Andes

Postgrado en Computación

Prof. Isabel M. Besembel Carrera

Unidad II. Sesión 20. Transacciones cooperativas.

UNIVERSIDAD DE LOS ANDES

Introducción

- SMBD convencionales aseguran la atomicidad en concurrencia y en fallas.
- Los ambientes multiusuarios se caracterizan por:
  - ✓ Larga duración: Algunas secuencias de acceso a la BD pueden durar desde minutos hasta meses. Deshacer una Ti es inaceptable debido al costo económico y moral de la pérdida de trabajo realizado.
  - ✓ Control interactivo: Es inconveniente planificar Ti a priori y volver a hacer las actividades automáticamente es imposible.
  - ✓ Cooperación entre los usuarios: Los usuarios comparten los resultados de sus actividades cuando aún están en realización.
- La serialidad es incompatible con la consistencia, cuando hay un esfuerzo concertado para tomar la BD de un estado semánticamente consistente a otro.

UNIVERSIDAD DE LOS ANDES

Granularidad

- *Bloqueo de granularidad múltiple*: Las instancias se bloquean como compartidas (C) o como exclusivas (X), lectura o escritura, respectivamente.
- Las clases se pueden bloquear en cinco modos posibles:
  1. IC: Intención compartida, la Ti bloqueará las instancias como lo necesite.
  2. IX: Intención exclusiva, las instancias serán bloqueadas con C o X.
  3. C: Compartida, la clase con C y todas sus instancias con C.
  4. CIX: Compartida intención exclusiva, clase e instancias con C y la Ti bloquea algunas de ellas
  5. X: Exclusiva, la clase y todas sus instancias con X.

UNIVERSIDAD DE LOS ANDES

Protocolo

- Protocolo de bloqueo con granularidad múltiple garantiza la serialidad según:
  1. Se debe usar la matriz de compatibilidad

	IC	IX	C	CIX	X
IC	si	si	si	si	no
IX	si	si	no	no	no
C	si	no	si	no	no
CIX	si	no	no	no	no
X	no	no	no	no	no

  2. Para colocar C o IC en un nodo P, la Ti debe primero colocar un IX o IS en el padre de P
  3. Para colocar X, CIX o IX en P, la Ti debe primero colocar IX o CIX en su padre

UNIVERSIDAD DE LOS ANDES

Protocolo

4. Todas las Ti deben ser dos fases
5. Una Ti puede desbloquear un nodo P solo si ninguno de sus hijos está bloqueado por Ti.

- Las reglas 2 y 3 indican un bloqueo de la raíz hacia las hojas y la 5 un desbloqueo de las hojas hacia la raíz.
- *Planificación optimista*:
  - ✓ Las Ti tienen acceso libre a los objetos, pero cualquier actualización es asignada a una copia local o sombra del objeto.
  - ✓ Los detalles de todas las lecturas y escrituras se registran en el sistema.
  - ✓ Antes de que cualquier actualización sea hecha a la BD, la Ti pasa por una etapa de prueba donde se chequea si hubo alguna interacción no serial con otra Tj que ya haya pasado, si es así Ti es desecha y reiniciada, sino Ti es validada

UNIVERSIDAD DE LOS ANDES

Planificación optimista

- ✓ El proceso de deshacer se simplifica pues los cambios fueron hechos sobre las copias o sombras.
- ✓ Su principal problema es la cantidad de información que hay que mantener.
- ✓ Su ventaja es que asegura que las Ti de solo lectura nunca entran en conflicto con otra Tj. Ejm: GemStone

➢ *Transacciones anidadas*:

- ✓ Las Ti forman una jerarquía donde aquellas que tienen la misma madre deben ser seriales.
- ✓ Una Ti pasa solamente cuando su madre pasa y por ello, las modificaciones se hacen en la BD cuando pasa la raíz.
- ✓ Este modelo no permite Ti cooperativas.

UNIVERSIDAD DE LOS ANDES

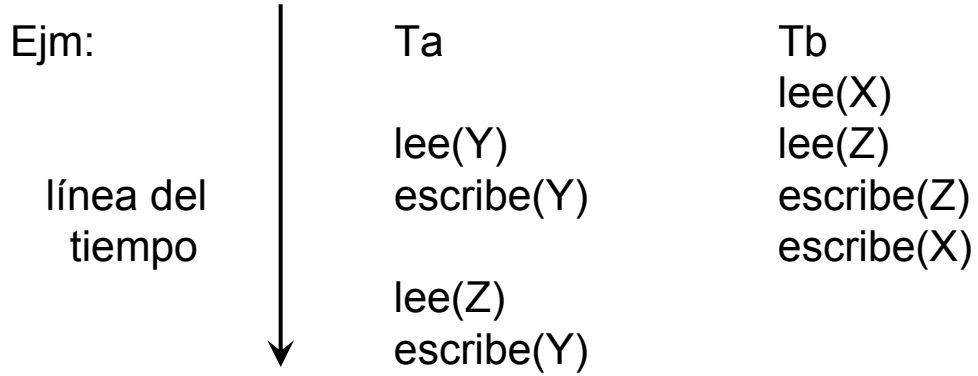
Modelo de chequeo

- *Modelo de chequeo a la salida*:
  - ✓ El enfoque más aceptado para actividades largas e interactivas, donde los usuarios copian los objetos desde una BD compartida para manipularlos en sus áreas de trabajo.
  - ✓ Un usuario puede reservar esos objetos hasta un chequeo posterior, o múltiples usuarios pueden mezclar sus actualizaciones con los objetos semánticamente relacionados.
  - ✓ Hay actividades cooperativas donde los usuarios se comunican fuera del sistema de control

UNIVERSIDAD DE LOS ANDES

Configuraciones

- Agrupamiento de versiones que son consistentes entre ellas.
- Dominios de direccionamiento relativo:
  - ✓ Un dominio es una configuración que consiste de las versiones seleccionadas de cada conjunto de objetos.
  - ✓ Dichos dominios son consistentes, por definición, ya que cada Ti lee las versiones de su dominio de entrada y escribe nuevas versiones generando su dominio de salida



UNIVERSIDAD DE LOS ANDES

Configuraciones

- El manejo de configuraciones asume que las versiones de los objetos son semánticamente consistentes, como el modelo clásico transaccional que lee de un estado consistente de la BD y pasa a otro estado semánticamente consistente, pero no asegura ninguna restricción de integridad particular, porque ellas son específicas de la aplicación

Asegurando la consistencia

Coordinación semántica

- *Smile:*
  - ✓ Ambiente de desarrollo de software multiusuario, utiliza el modelo reservación/depósito con una variante que asegura la consistencia semántica.
  - ✓ Este sistema usa un área de trabajo privada denominada BD experimental (BDE) y no soporta versiones ni configuraciones.
  - ✓ Tiene una configuración pública en la BD principal (BDP) y varias configuraciones privadas donde están las versiones de los objetos que aún no han sido depositados.
  - ✓ Utiliza el módulo como unidad de chequeo y cualquier usuario puede agregar módulos a su BD experimental.
  - ✓ Cuando un usuario requiere depositar su BD experimental, el sistema analiza todos sus módulos, los recompila y los enlaza con los módulos en la BD principal.
  - ✓ Si se detectan errores, la operación se aborta y el usuario debe arreglar los errores y probar de nuevo

UNIVERSIDAD DE LOS ANDES

Coordinación semántica

- Ejm:

Ta	Tb	
	lee(X)	
	lee(Z)	BDEb
lee(Y)	modifica(Z)	
modifica(Y)	modifica(X)	

- Luego de depositar BDEb puede seguir siendo utilizada, pero no puede obtener ningún módulo nuevo de la BDP, ni ser depositada
- Asegura la consistencia entre los objetos accedidos dentro de la misma Ti larga y la consistencia global con respecto a los otros objetos.
- Para asegurar los depósitos inválidos, un usuario no puede modificar la interfaz de un módulo si no ha reservado todos los módulos que dependen de dicha interfaz o la porción de dicha interfaz que será cambiada.

Ejm: la función f de Z es llamada por la función g de Y, f puede ser cambiada luego de reservar Z

Tb debe reservar también Y si va a cambiar la firma de f en Z

Si Ta ha reservado Y en BDEa, la operación de editar la firma de f es abortada y Tb debe esperar por Ta

UNIVERSIDAD DE LOS ANDES

Consistencia multinivel

- *Infuse*
  - ✓ Ambiente de desarrollo de software multiusuario que usa la reservación en la BDE pero ella no es necesariamente privada para el usuario.
  - ✓ Soporta una jerarquía multinivel donde todos los módulos reservados en la BDP se colocan en la misma BDE de alto nivel que es compartida por todos los usuarios, desde la cual se pueden reservar módulos para BDEs hijas que representan a grupos de usuarios, que a su vez pueden tener BDEs hijas para los subgrupos, y así hasta llegar a un usuario particular.
  - ✓ El nivel más alto se corresponde aproximadamente a una Ti de alto nivel en Ti anidadas
  - ✓ En Smile el chequeo de consistencia es estático mientras que en Infuse es dinámico
  - ✓ El chequeo se inicia una vez que todas la BDE hijas han sido depositadas.
  - ✓ Los módulos reservados se compilan y prueban, no son depositados en la BDE madre hasta que no hayan pasado las pruebas.

UNIVERSIDAD DE LOS ANDES

Consistencia multinivel

- Ejemplo:
  - ✓ Ta y Tb crean una $BDE_{X,Y}$ donde X y Y están reservadas
  - ✓ Tb crea una BDE_X hija donde reserva X
  - ✓ Ta crea su BDE_Y hija donde reserva Y
  - ✓ Tb anexa Z a $BDE_{X,Y}$ y a BDE_X obteniendo $BDE_{X,Y,Z}$ y $BDE_{X,Z}$
  - ✓ Tb cambia la interfaz de f y deposita $BDE_{X,Z}$ en $BDE_{X,Y,Z}$
  - ✓ Ta modifica Y y deposita BDE_Y en $BDE_{X,Y,Z}$
  - ✓ Ta y Tb tratan de depositar $BDE_{X,Y,Z}$ en la BDP pero no es consistente
  - ✓ Ta o Tb deben crear una $BDE_{Y'}$ para arreglar el problema
- Permite mayor concurrencia que Smile, pero el costo es mayor inconsistencia.
- Soporta Ti *trascendentes* para usuarios autorizados donde hay solamente consultas de lectura y que pueden contener cualquier subconjunto de BDE de la jerarquía.
- Posibilidad de definir áreas de trabajo creadas y eliminadas al vuelo, para contener BDE de la jerarquía para un grupo temporal, con el propósito de chequear la consistencia antes de depositarlas

UNIVERSIDAD DE LOS ANDES

Coordinación optimista

- Enfoque *copiar/modificar/mezclar o coordinación optimista:*
  - ✓ Los usuarios de un mismo grupo necesitan cambiar concurrentemente el mismo objeto y normalmente, ellos conocen los planes de cada uno con anterioridad y luego de sus cambios, mezclarán sus versiones para obtener una versión consistente.
  - ✓ No hay lectura de una versión para escribir nuevas versiones diferentes
- NSE
  - ✓ opera sobre una BD jerárquica, donde cada nodo es un *ambiente* que aquí representa una vista de sólo lectura del sistema de archivos, pero que actualiza copias privadas de los archivos
  - ✓ Control de concurrencia: permite varios ambientes hermanos para reservar simultáneamente la misma versión de un objeto y cuando los conflictos se descubran en la etapa de depósito, se mezclan sus actualizaciones en el sistema de archivo compartido

UNIVERSIDAD DE LOS ANDES

Coordinación optimista

- El primer ambiente que termine, deposita su copia como una nueva versión del archivo en su ambiente padre, el segundo mezcla su copia con la creada por el primero, creando una nueva versión, y así sucesivamente.
- Ejemplo:

Ta y Tb crean $AMB_{X,Y}$ para X y Y.

Tb crea su AMB_X y luego anexa Z resultando $AMB_{X,Y,Z}$ y $AMB_{X,Z}$

Ta crea su AMB_Y y compila/encadena con la versión vieja de Z en AMB $_{X,Y,Z}$

El que deposite su ambiente de último deberá mezclarlo con el del otro

UNIVERSIDAD DE LOS ANDES

Modelado de la coordinación

- Los modelos anteriores trabajan sobre objetos individuales sin mantener una noción global de consistencia sobre varios objetos.
- El modelo de Walter-84 propone una relación más flexible entre Ti madres e hijas.
  - ✓ La interfaz entre una Ti madre y una hija puede ser: una *petición simple*, donde la madre hace una consulta o modificación a la hija y espera hasta que ella responda, o *conversacional*, donde el control se alterna entre madre e hija, la madre hace peticiones y la hija responde cada una individualmente
  - ✓ La interfaz conversacional necesita el agrupamiento de las Ti madre y sus hijas en una *esfera trasera* ya que si la Ti hija es abortada en el medio de la conversación, el sistema debe deshacer la Ti hija y la Ti madre hasta el punto donde se inició la conversación

UNIVERSIDAD DE LOS ANDES

Modelado de la coordinación

- La interfaz de petición simple es soportada por el modelo de Ti anidadas, donde no hace falta deshacer la madre si la hija no pasa.
- Adicionalmente, este modelo propone una *esfera de validación*, donde cualquier Ti puede validarse sólo si todas la Ti en la esfera se validan.
- La sincronización entre madres e hijas se hace a través de la petición simple o conversacional con bloqueos u otro esquema
- Con los tres aspectos: interfaz, dependencia y sincronización, Walter presenta su modelo de Ti anidadas, donde cada Ti tiene esos tres atributos
- Interfaz {BACKOUT, NOBACKOUT}: BACKOUT: Ti hija está asociada con su propia esfera trasera y NOBACKOUT la esfera trasera de la hija es la de su mamá.
- Dependencia: {COMMIT, NOCOMMIT}
- Sincronización: {SYNC, NOSYNC}

UNIVERSIDAD DE LOS ANDES

Modelado de la coordinación

- Ejemplo:
  - ✓ Una Ti hija con {BACKOUT, COMMIT, SYNC} es independiente de su mamá y
  - ✓ necesita el uso de bloqueos para sincronizar su acceso con los de su mamá.
- El modelo clásico de Ti anidadas se obtiene colocándole a la Ti hijas {BACKOUT, NOCOMMIT, SYNC}
- Infuse y NSE serian {BACKOUT, NOCOMMIT, NOSYNC} pero este modelo, con esos tres aspectos, no logra diferenciar realmente lo que soportan ambos

UNIVERSIDAD DE LOS ANDES

Cooperación

- Existen grupos pequeños donde se puede asumir una cooperación frecuente fuera del sistema.
- Los grupos grandes por lo general se dividen en grupos pequeños.
- Es necesaria una coordinación entre los miembros de un grupo y una entre los grupos
- **Primitivas de cooperación**
  - ✓ *Notificación*: Alerta al usuario sobre los intentos de bloqueos de otras Ti o de la creación de nuevas versiones de los objetos. Junto con ella hay una resolución interactiva de conflictos, en vez de asegurar la serialidad. Si se usan versiones y bloqueos no exclusivos, se tiene dos posibles notificaciones
    - o Notificación inmediata: Alerta a los usuarios afectados sobre cualquier intento de acceso conflictivo tan pronto como él ocurre

UNIVERSIDAD DE LOS ANDES

Cooperación

- Notificación tardía: Notifica a los usuarios de los conflictos que han ocurrido cuando una de las Ti conflictivas trata de pasar.

- ✓ Los conflictos se resuelven vía llamada telefónica al usuario.
- ✓ Estas políticas incluyen la intervención humana como parte del algoritmo de resolución de los conflictos.
- ✓ Ventaja: Mejora la concurrencia cuando hay muchas tareas interactivas.
- ✓ Desventaja: Puede degradar la consistencia porque los humanos no siempre resolverán correctamente los conflictos
- ✓ *Reestructuración dinámica*: Incluye dos nuevas operaciones: fisión de Ti y acoplamiento de Ti
  - Idea básica: el plan de ejecución es serial hasta que las Ti pasen.
  - El plan puede incluir nuevas Ti producto de fisiones de viejas Ti o de acoplamiento de Ti

UNIVERSIDAD DE LOS ANDES

Cooperación

Ta	Tb
	inicio
inicio	lee(X)
lee(Y)	lee(Z)
escribe(Y)	escribe(Z)
	escribe(X)
pide leer(Z)	notificación (Z)
	rompe ((Z), (X))
	pasa(Z)
lee(Z)	
escribe(Y)	
pasa(Y, Z)	
	tiempo

- Una fisión de Ti divide una Ti ejecutándose en 2 o más Ti seriales por la división de sus operaciones y de sus recursos.
- Las Ti resultantes proceden independientemente y la Ti original desaparece como si ella nunca hubiera existido.
- Solo se hace si es posible obtener las nuevas Ti seriales entre ellas y con las otras Ti
- La fisión normalmente se da, cuando dentro de una Ti en ejecución se conoce algún dato que hace que sea posible romperla

UNIVERSIDAD DE LOS ANDES

Cooperación

- Acoplamiento de Ti: operación inversa de la anterior, donde se mezclan dos o más Ti independientes en ejecución como si ellas nunca hubieran existido, en una única Ti.

✓ *Ti de grupos*: Las Ti en un grupo pueden cooperar en formas no permitidas para las Ti fuera del grupo o entre los grupos.

- Categoriza las Ti en: Ti de grupo (TG) y Ti de usuarios (TU). Cada TU es una sub-Ti en TG
- Una TG reserva sus objetos de la BD pública en la BD del grupo, donde los usuarios reservan sus objetos en la BD del usuario.
- Los grupos están aislados unos de otros.
- Las TG son seriales con respecto de unas y otras.
- Dentro de las TG, las TU pueden ser seriales o cooperativas siguiendo un esquema no serial.
- El mecanismo básico provee una relajación de la serialidad si se usan versiones y notificación

UNIVERSIDAD DE LOS ANDES

Cooperación

- Soporta 5 formas de bloqueo: Solo-lectura, lectura/derivación, derivación compartida, derivación exclusiva y bloqueo exclusivo.
- Cada Ti es dos fases. Si una Ti desea bloquear una gránula en un modo incompatible con otra, ésta es informada de la imposibilidad de bloqueo y luego se le notifica cuando la gránula esté disponible

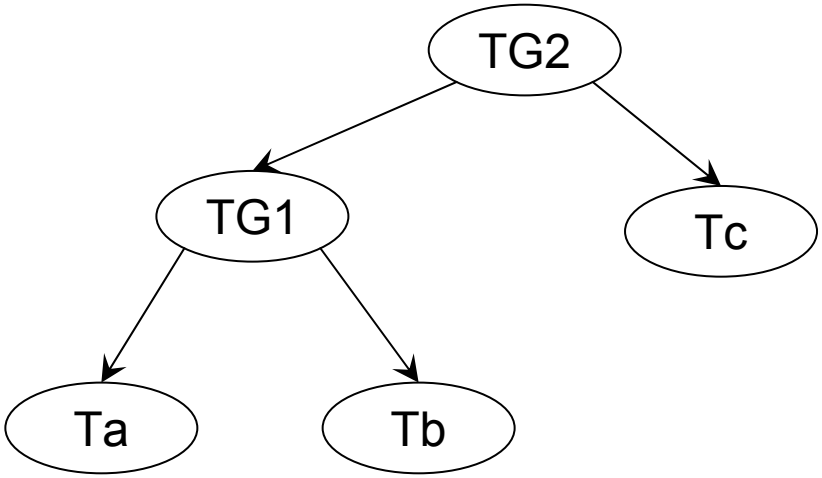
✓ Grupos de Ti: Un grupo de Ti es un proceso que controla el acceso a la BD por el conjunto de Ti cooperantes y subgrupos de Ti (miembros del grupo).

- Dentro del grupo de Ti, sus Ti miembros y sus subgrupos se sincronizan en base a un protocolo de entrada que define algún criterio de correctitud semántica apropiado para la aplicación.
- Criterio: se especifica con patrones semánticos y se asegura con el reconocedor y detector de conflictos.
- Reconocedor: asegura que los bloqueos estén de acuerdo al conjunto de ellos definido para los miembros del grupo.

UNIVERSIDAD DE LOS ANDES

Cooperación

- Detector: asegura que los bloqueos hechos a un objeto no estén en conflicto con la petición de bloqueo hecha
- Las transformaciones se hacen con un protocolo de salida que consulta la tabla de bloqueos para determinar como transformar una petición de bloqueo de un subgrupo en un bloqueo de grupo aceptado.
- Ejemplo: Carlos es responsable de la documentación del proyecto y de su actualización.



UNIVERSIDAD DE LOS ANDES

Cooperación

Ta	Tb
	inicio(D)
inicio(D)	lee(X)
lee(Y)	lee(Z)
escribe(Y)	escribe(Z)
	escribe(X)
lee(Z)	escribe(X)
escribe(Y)	
pasa(Y,Z)	
	lee(Y)
	escribe(X)
	pasa(X, Y, Z)
	tiempo

- ✓ Ti participantes: Define cada Ti como participante en un dominio específico, donde las Ti participantes no necesitan ser hechas en un orden serial.
  - Un dominio está representado por un grupo de Ti controlado por los usuarios que colaboran en una tarea común.
  - No hay la restricción de que todas las Ti del dominio deben pasar juntas o de que todas ellas deben pasar.
  - Aquellas Ti que no participan en un dominio se consideran observadoras y ellas deben ser seriales con respecto a las del dominio

UNIVERSIDAD DE LOS ANDES

Cooperación

- **Conclusión**
  - ✓ No hay consenso sobre cual de estos modelos es el mejor, ni siquiera si hay un enfoque superior para todos los ambientes de aplicación.