L15: Transactions

Sam Madden
6.033, Spring 2014
Bank account transfer

\[
xfer(bank, a, b, amt):
    \begin{align*}
    bank[a] &= bank[a] - amt \\
    bank[b] &= bank[b] + amt
    \end{align*}
\]
Bank account transfer

xfer(bank, a, b, amt):

bank[a] = bank[a] – amt

bank[b] = bank[b] + amt

Crash!
Bank account transfer

\[
xfer(bank, a, b, amt):
\]
\[
\text{bank}[a] = \text{bank}[a] - \text{amt}
\]
\[
\text{bank}[b] = \text{bank}[b] + \text{amt}
\]

Crash!

Lost $\text{amt}$
Transfer w/ Audit

xfer(bank, a, b, amt):
    bank[a] = bank[a] – amt
    bank[b] = bank[b] + amt

audit(bank):
    sum = 0
    for acct in bank:
        sum = sum + bank[acct]
    return sum

A = 100
B = 100
xfer(A,B,50)
Transfer w/ Audit

xfer(bank, a, b, amt):
    bank[a] = bank[a] – amt
    bank[b] = bank[b] + amt

audit(bank):
    sum = 0
    for acct in bank:
        sum = sum + bank[acct]
    return sum

A = 100
B = 100
xfer(A,B,50)

audit(bank):

  sum=200
Transfer w/ Audit

```python
A = 100
B = 100
xfer(A,B,50)
```

```python
xfer(bank, a, b, amt):
    bank[a] = bank[a] – amt
    bank[b] = bank[b] + amt

audit(bank):
    sum = 0
    for acct in bank:
        sum = sum + bank[acct]
    return sum
```
Transfer w/ Audit

xfer(bank, a, b, amt):
    bank[a] = bank[a] – amt
    bank[b] = bank[b] + amt

audit(bank):
    sum = 0
    for acct in bank:
        sum = sum + bank[acct]
    return sum

A = 100
B = 100
xfer(A,B,50)

audit(bank):
    sum=200
    sum=150
    sum=200
Transfer w/ Audit

\[
\text{xfer(bank, a, b, amt):} \\
\quad \text{bank}[a] = \text{bank}[a] - \text{amt} \\
\quad \text{bank}[b] = \text{bank}[b] + \text{amt}
\]

\[
\text{audit(bank):} \\
\quad \text{sum} = 0 \\
\quad \text{for acct in bank:} \\
\quad \quad \text{sum} = \text{sum} + \text{bank}[acct] \\
\quad \text{return sum}
\]
Two goals

• Want “all or nothing” atomicity for complex operations in the presence of crashes

• Want serial equivalence
  • Concurrent operations are isolated from each other
  • Never see each other’s intermediate state
Abstraction: transactions
all-or-nothing & isolation

T1:

begin
transfer(A,B,20)
debit(B,10)
...
end

T2:

begin
transfer(B,C,5)
deposit(A,5)
...
end
Atomic Transfer: Strawman

\[
xfer(\text{bankfile, a, b, amt}): \\
\text{bank} = \text{read}\_\text{accounts(\text{bankfile})} \\
\text{bank}[a] = \text{bank}[a] - \text{amt} \\
\text{bank}[b] = \text{bank}[b] + \text{amt} \\
\text{write}\_\text{accounts(\text{bankfile})}
\]
Atomic Transfer: Shadow Copy

\[
xfer(bankfile, a, b, amt):
\]
\[
  \text{bank} = \text{read\_accounts}(bankfile)
\]
\[
  \text{bank}[a] = \text{bank}[a] - \text{amt}
\]
\[
  \text{bank}[b] = \text{bank}[b] + \text{amt}
\]
\[
  \text{write\_accounts}("#".bankfile) // = \text{concat}
\]
\[
  \text{rename}("#".bankfile, bankfile)
\]
File system data structures

directory entries (aka dirents):
  filename “bank” → inode 12
  filename “#bank” → inode 13

inode 12:
  data blocks: 3, 4, 5
  refcount: 1

inode 13:
  data blocks: 6, 7, 8
  refcount: 1
rename(“#bank”, “bank”)

directory entries:
  filename “bank” → inode 12
  filename “#bank” → inode 13

inode 12:
  data blocks: 3, 4, 5
  refcount: 1

inode 13:
  data blocks: 6, 7, 8
  refcount: 1
rename(“#bank”, “bank”)

directory entries:
    filename “bank” → inode 12
    filename “#bank” → inode 13

inode 12:
    data blocks: 3, 4, 5
    refcount: 1

inode 13:
    data blocks: 6, 7, 8
    refcount: 2
rename("#bank", "bank")

directory entries:
  filename "bank" → inode 13
  filename "#bank" → inode 13

inode 12:
  data blocks: 3, 4, 5
  refcount: 1

inode 13:
  data blocks: 6, 7, 8
  refcount: 2
rename("#bank", "bank")

directory entries:
    filename "bank" → inode 13
    filename "#bank" → inode 13

inode 12:
    data blocks: 3, 4, 5
    refcount: 0

inode 13:
    data blocks: 6, 7, 8
    refcount: 2
rename(“#bank”, “bank”)
rename("#bank", "bank")

directory entries:
    filename "bank" → inode 13
    filename "#bank" → inode 13

inode 12:
    data blocks: 3, 4, 5
    refcount: 0

inode 13:
    data blocks: 6, 7, 8
    refcount: 1
Recovery after crash

```
salvage(disk):
    for inode in disk.inodes:
        inode.refcnt =
            find_all.refs(disk.root_dir, inode)

if exists("#bank"):
    unlink("#bank")
```