L15: Transactions

Sam Madden
6.033, Spring 2014

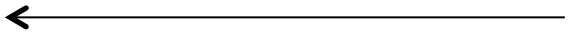
Bank account transfer

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

Bank account transfer

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

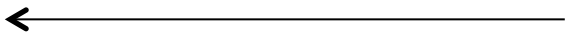
Crash!



Bank account transfer

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

Crash!



Lost $amt

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)

Transfer w/ Audit

audit(bank):

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

⟵ *sum=200*

```
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

audit(bank):

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

⟵ *sum=200*

⟵ *sum=200*

```
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

audit(bank):

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

← *sum=200*
← *sum=150*
← *sum=200*

```
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

audit(bank):

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

← *sum=200*
← *sum=150*
← *sum=200*

```
audit(bank):
    sum = 0
    for acct in bank:
        sum = sum + bank[acct]
    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(bankfile, a, b, amt):
    bank = read_accounts(bankfile)
    bank[a] = bank[a] – amt
    bank[b] = bank[b] + amt
    write_accounts(bankfile)
```

Atomic Transfer: Shadow Copy

```
xfer(bankfile, a, b, amt):
    bank = read_accounts(bankfile)
    bank[a] = bank[a] – amt
    bank[b] = bank[b] + amt
    write_accounts("#".bankfile) //. = concat
    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”)

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”)

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”)
```