

Removing Unreferenced Entities

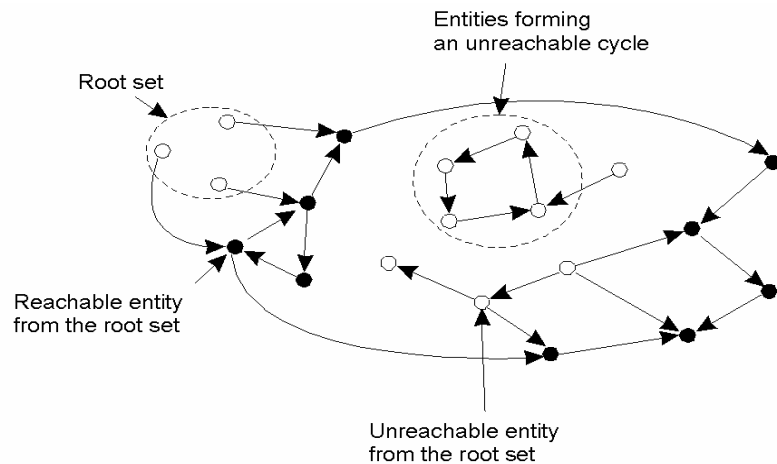
Unreferenced Objects: Problems

- Assumption: Objects may exist only if it is known that they can be contacted:
 - Each object should be named
 - Each object can be located
 - A reference can be resolved to client–object communication
- Problem: Removing unreferenced objects:
 - How do we know when an object is no longer referenced (think of cyclic references)?
 - Who is responsible for (deciding on) removing an object?

7/12/2005

44

The Problem of Unreferenced Objects



An example of a graph representing objects containing references to each other.

7/12/2005

45

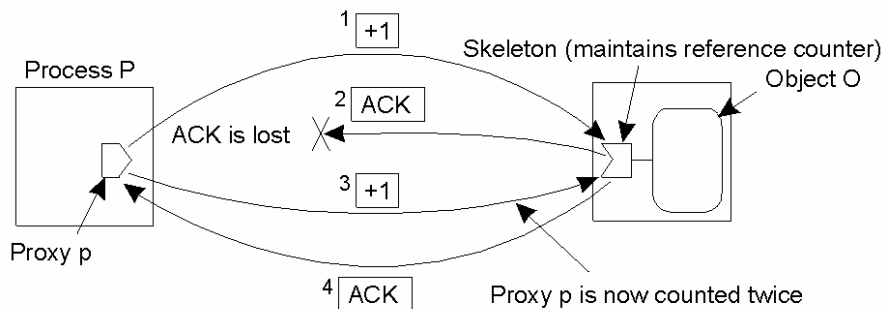
Reference Counting (1)

- Each time a client creates (removes) a reference to an object O, a reference counter local to O is incremented (decremented)
- Problem 1: Dealing with lost (and duplicated) messages:
 - An increment is lost so that the object may be prematurely removed
 - A decrement is lost so that the object is never removed
 - An acknowledgement is lost, so that the increment/decrement is resent
- Solution: Keep track of duplicate requests.

7/12/2005

46

Reference Counting (2)



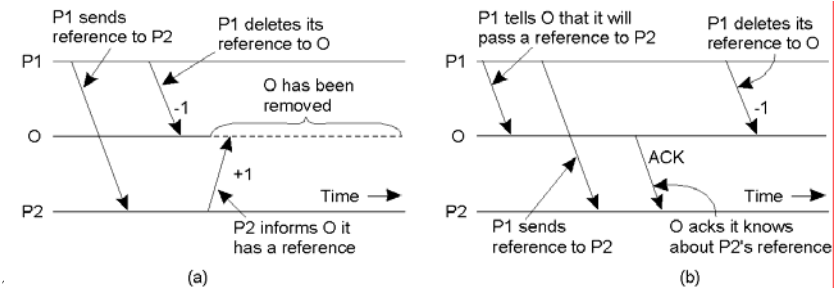
The problem of maintaining a proper reference count in the presence of unreliable communication.

7/12/2005

47

Reference Counting (3)

- Problem 2: Dealing with duplicated references – client P1 tells client P2 about object O:
 - Client P2 creates a reference to O, but dereferencing (communicating with O) may take a long time
 - If the last reference known to O is removed before P2 talks to O, the object is removed prematurely
- Solution 1: Ensure that P2 talks to O on time:
 - Let P1 tell O it will pass a reference to P2
 - Let O contact P2 immediately
 - A reference may never be removed before O has acknowledged that reference to the holder



7/1:

48

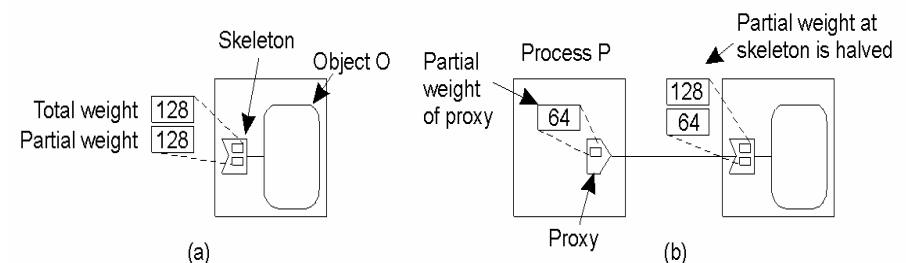
Weighted Reference Counting

- Solution 2: Avoid increment and decrement messages:
 - Let O allow a maximum M of references
 - Client P1 creates reference, implies grant it M/2 credit
 - Client P1 tells P2 about O, it passes half of its credit grant to P2
 - Pass *current* credit grant back to O upon reference deletion

7/12/2005

49

Advanced Referencing Counting (1)

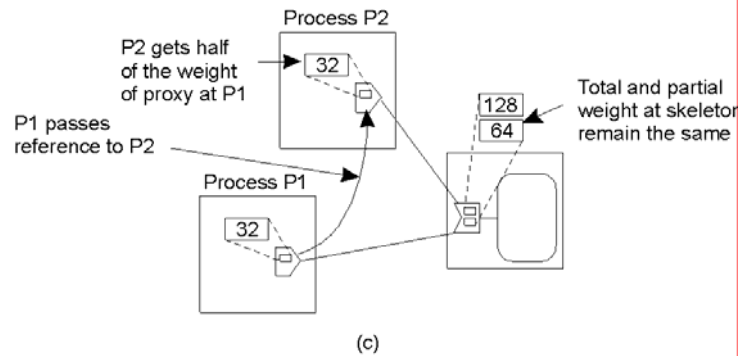


- The initial assignment of weights in weighted reference counting
- Weight assignment when creating a new reference.

7/12/2005

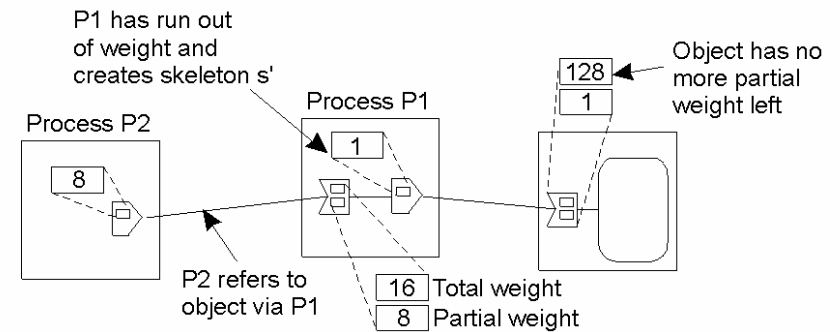
50

Advanced Referencing Counting (2)



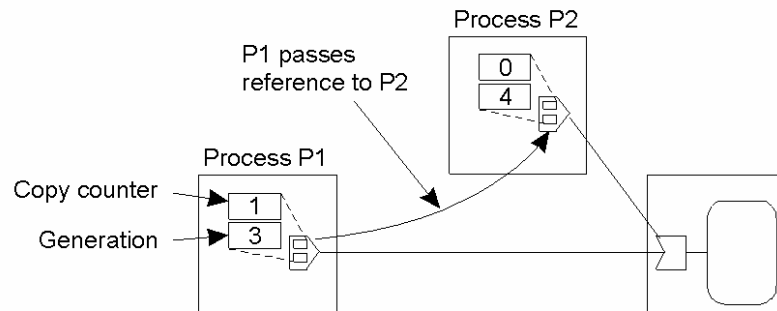
c) Weight assignment when copying a reference.

Advanced Referencing Counting (3)



Creating an indirection when the partial weight of a reference has reached 1.

Generation Referencing Counting



Creating and copying a remote reference in generation reference counting.

Reference Listing

- Observation: We can avoid many problems if we can tolerate message loss and duplication
- Reference listing: Let an object keep a list of its clients:
 - Increment operation is replaced by an (*idempotent*) insert
 - Decrement operation is replaced by an (*idempotent*) remove
- There are still some problems to be solved:
 - **Passing references:** client *B* has to be listed at *O* before last reference at *O* is removed (or keep a chain of references)
 - **Client crashes:** we need to remove outdated registrations (e.g., by combining reference listing with leases)