```
separate (Dining_Philosophers)
task body Waiter is
  entry Wait(ID);     -- used to requeue philosophers
  eating : array (ID) of Boolean; -- who is eating
  want : array (ID) of Boolean;   -- who wants to eat
  go : array(ID) of Boolean;      -- who can go now
begin
  for j in ID loop           -- initialize the arrays
    eating(j) := False; want(j) := False;
  end loop;
  loop             -- basic server loop
   select
     accept Pickup(i : in ID) do  -- DP(i) needs forks
       if not(eating(left(i)) or eating(right(i))) then
         eating(i) := True;
       else
         want(i) := True; requeue Wait(i);
       end if;
     end;
   or
     accept Putdown(i : in ID) do  -- DP(i) is done
       eating(i) := False;
     end;
     -- check neighbors to see if they can eat now
     if want(left(i)) and not eating(left(left(i))) then
       accept Wait(left(i));
       eating(left(i)) := True; want(left(i)) := False;
     end if;
     if want(right(i)) and not eating(right(right(i)))
       then  accept Wait(right(i));
       eating(right(i)) := True; want(right(i)) := False;
     end if;
   or
     terminate;  -- quit when philosophers have quit
   end select;
  end loop;
end Waiter;
```

Figure 8.19 Dining philosophers in Ada: Waiter task.