15.6a Exception scope

As mentioned in Section 15.6, it is even possible to propagate an exception out of its scope, where it becomes anonymous, and then back in again where it can once more be handled by its proper name. This is illustrated by the following extremely artificial example

```
procedure Main is

   package P is
      procedure F;
      procedure H;
   end P;

   procedure G is
   begin
      P.H;
   exception
      when others =>
         raise;
   end G;

   package body P is
      X: exception;

      procedure F is
      begin
         G;
      exception
         when X =>
            Put("Got it!");
      end F;

      procedure H is
      begin
         raise X;
      end H;

   end P;

begin
   P.F;
end Main;
```

The procedure Main declares the specification of a package P containing procedures F and H, then a procedure G, and finally the body of the package P. The procedure Main calls F in P which calls G outside P which in turn calls H back in P. The procedure H raises the exception X whose scope is the body of P. The procedure H does not handle X, so it is propagated to G which called H. The procedure G is outside the package P, so the exception X is now outside its scope; nevertheless G handles the exception anonymously using **others** and propagates it further by reraising it using a raise statement without any argument. The procedure G was called by F so X is now propagated back into the package and can then be handled by F using its proper name.