## Writing a Reply Filter Function

If your application calls **AESend** and chooses to yield the processor to other processes while waiting for a reply, you can provide an idle function to process update, null, operating-system, and activate events and, additionally, you can provide a reply filter function to process high-level events. The section **Writing an Idle Function** describes how an idle function processes events.

Your reply filter function can process any high-level events that it is willing to handle while waiting for a reply Apple event. For example, your application can choose to handle Apple events from other processes while waiting. Note, however, that your application must maintain any necessary state information. Your reply filter function must not accept any Apple events that can change the state of your application and make it impossible to return to its previous state.

A reply filter function must use this syntax:

```
Boolean MyWaitReplyFilter (EventRecord *theEventRecord, long returnID,
                           long transactionID, AEAddressDesc *sender);
```

The parameter *theEventRecord* is the event record for a high-level event. The next three parameters contain valid information only if the event is an Apple event. The *transactionID* parameter is the transaction ID for the Apple event. The *returnID* parameter is the return ID for the Apple event. The *sender* parameter contains the address of the application or process that sent the Apple event.

Your reply filter function should return TRUE as the function result if you want to accept the Apple event; otherwise it should return FALSE. If your filter function returns TRUE, the **Apple Event Manager** calls the **AEProcessAppleEvent** function on behalf of your application, and your handler routine is called to process the Apple event.