New Technical Notes

Macintosh ®

Developer Support

NW 7 - Avoid Use of Network Events

Networking

Revised by: March 1988
Written by: Bryan Stearns July 1987

Future System software enhancements will not support network events. This note gives hints on weaning your application from the use of network events.

What are network events?

When the Event Manager was designed, an event number was reserved for future support of "network events". Later, when the AppleTalk Pascal Interfaces were written, a completion routine was created that, when an asynchronous AppleTalk operation finished, would post an event using `networkEvt` in the `evtNum` field.

Only the AppleTalk Pascal Interfaces generate network events. Assembly-language users of the AppleTalk drivers (and those who called the AppleTalk drivers directly from high-level languages, using `PBControl` calls) either provide a completion routine of their own, or poll the `ioResult` field of the parameter block passed with the call (when `ioResult` became negative or zero, the call is complete).

Why not use network events?

In some cases, network events can be lost. If the Event Manager finds that the queue is full while posting an event, it discards the oldest event. In a situation (such as a server) where multiple asynchronous ATP requests may complete at once, there is a chance that events may be dropped off the end of the queue. This is more likely if the same machine is also handling user-interface events (like keypresses and mouse actions).

Also, in developing improvements to our operating system, it has become apparent that to continue support of network events, we would have to compromise future enhancements to our system. So, future versions of the Macintosh operating system may ignore network events instead of passing them to the application.

How can I tell that my calls have completed without using network events?

As described on page II-275 of *Inside Macintosh*, you can poll the `abResult` field of the call's `ABusRecord`; when this value becomes negative or zero, the call has completed. You can do this in your main event loop.

With this technique, you can ignore any network events returned by `GetNextEvent`, since the AppleTalk Pascal Interfaces will be posting events anyway. If your application starts enough asynchronous operations, it's possible that their network events will cause other non-network events to be lost. To prevent this, you should call `FlushEvents(networkMask,0)` frequently to purge any accumulated network events from the event queue.

You may also consider using the new preferred high-level interface calls; see Technical Note NW 2 - AppleTalk Interface Update for more information.

Further Reference:

- AppleTalk Manager
- Technical Note NW 2 - AppleTalk Interface Update