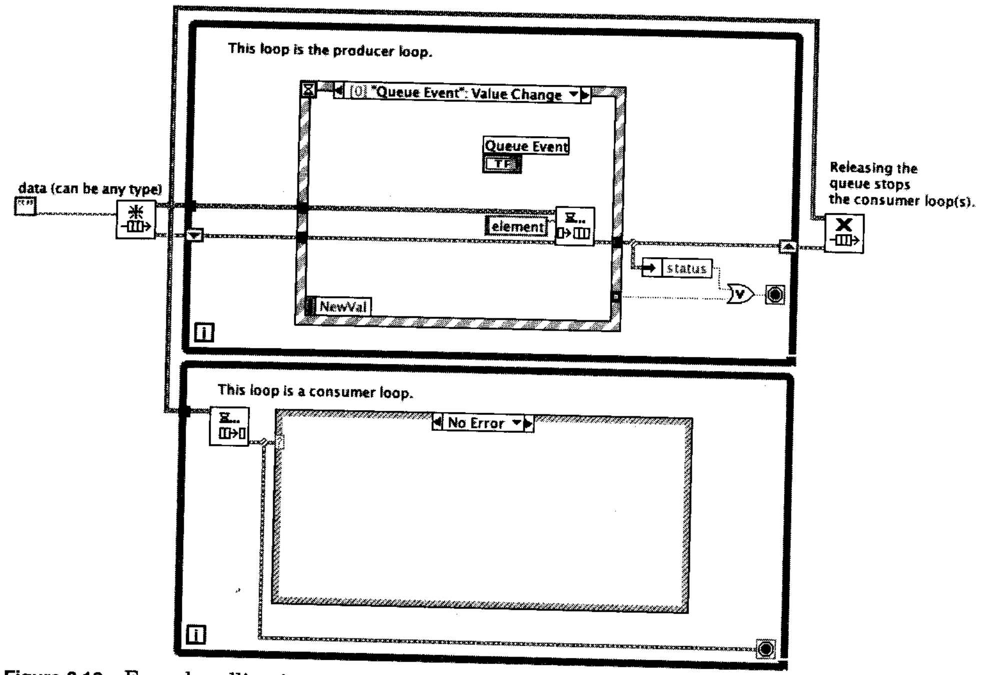

This loop is the producer loop.
[0] "Queue Event": Value Change
Queue Event
data (can be any type)
element
NewVal
status
Releasing the queue stops the consumer loop(s).
This loop is a consumer loop.
No Error

Figure 8.16 Error handling in producer/consumer design pattern. Any error terminates a loop. Is that what you want in a robust application?

The messages are put into a queue (**enqueued**) by the **producer** and removed from the queue (**dequeued**) by the bottom **consumer** loop. Stopping the producer loop releases the queue and stops the consumer loop when the dequeue function generates an error. If any subVIs in the consumer loop were included in the error chain between the dequeue function and the stop condition, it would programmatically alter the behavior by forcing the loop to terminate on any error. The queued message handler is a common design pattern for programming user-interface applications; however, because there is only one-way communication between the loops, the producer loop will never know if the consumer loop terminated—it will just go on putting messages into the queue. Clearly this is not a robust solution!

One flexible error-handling scheme we've used, and seen used by others, is based on the addition of an extra top-level error-handling loop to which all error reports are passed via a **global queue**. Any VI is permitted to deposit an error in the queue at any time, which causes the error handler to wake up and evaluate that error. Figure 8.17 shows the top-level error-handling loop. The reference to **Error Queue** is stored in **ErrorQueue.vi** (Figure 8.18). Any incoming error placed into the queue is handled by the **Simple Error Handler.vi**. Figure 8.19 shows one way to put the code into action.

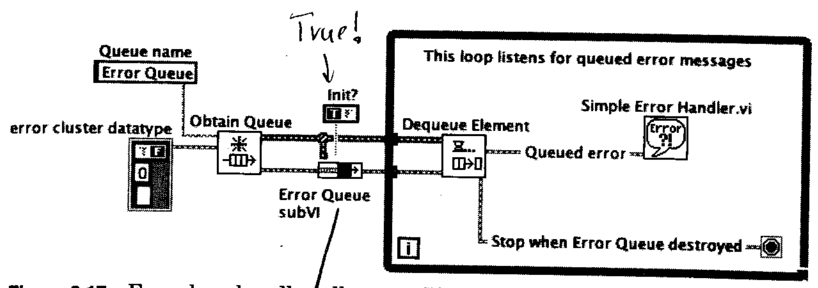


Figure 8.17 Error loop handles all errors. The Error Queue reference is stored in ErrorQueue.vi (Figure 8.18).

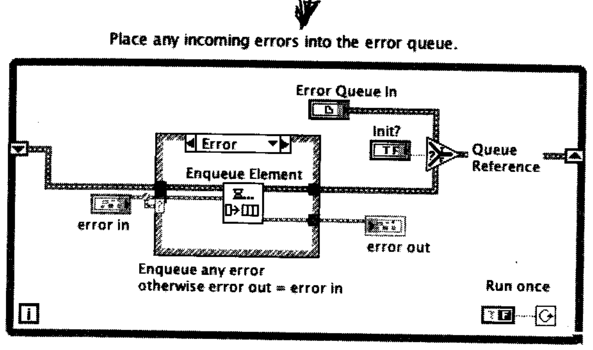
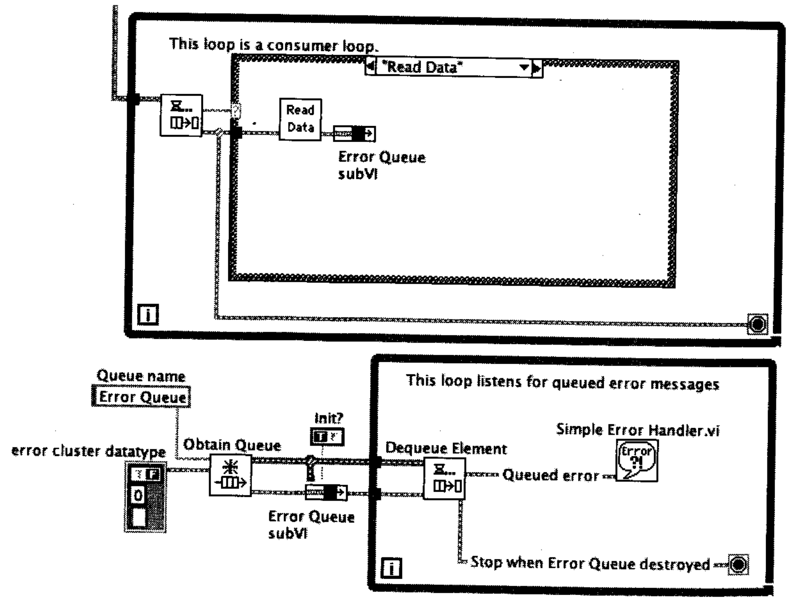

Place any incoming errors into the error queue.
Error Queue In
Init?
Error
Enqueue Element
Queue Reference
error in
error out
Enqueue any error otherwise error out = error in
Run once

Figure 8.18 ErrorQueue.vi enqueues incoming errors into the Error Queue. The queue reference is stored in a shift register and remains valid until the queue is destroyed.



Mie
m

Figure 8.19 ErrorQueue.vi and the error-handling loop in action. All errors are reported via the Error Queue.