7.5 Error recovery

All of the events in the following table cause one or more of the target's task sets to be aborted; see ANSI NCITS 325-1998 for details. Unless otherwise noted, a target shall preserve execution context for active ORBs across these events.

Event	AGENT_STATE.*st*	Comment
Unrecoverable transaction errors	DEAD	Store status block TRANSPORT FAILURE for faulted ORB, if possible.
ABORT TASK SET		
~~LOGICAL UNIT RESET~~		~~Target support for LOGICAL UNIT RESET is optional~~
Fetch agent reset (write to AGENT_RESET)	RESET	
TARGET RESET	DEAD	
Bus reset	RESET	For each login, a target shall retain, for at least *reconnect_hold* + 1 seconds after the bus reset, sufficient information to permit initiators to reconnect their logins. After this time, a target shall discard execution context for the task set of any initiator that failed to reconnect.
Command reset (write to RESET_START)		These events are equivalent; execution context for all ORBs in all task sets shall be discarded. Device operations should be halted and the device restored to an idle condition.
Power reset		

Unrecoverable transaction errors may be caused by a missing acknowledgement packet, a split transaction timeout, a data error or a retry limit exceeded. A missing acknowledgment by itself is not necessarily an unrecoverable error; the target shall wait a split timeout period before further action. If a transaction response is received within the split timeout period, there is no error. Otherwise, a split transaction timeout has occurred. In the case of a data error or a split transaction timeout, if the request was not addressed to an initiator *status_FIFO,* target may retry the transaction up to some implementation-dependent limit. Once the target deems a transaction error unrecoverable, it shall set the *resp* field in completion status for the faulted ORB to TRANSPORT FAILURE and transition the fetch agent to the dead state.

When an unrecoverable transaction errors occur for a request that does not address an initiator *status_FIFO*, the target should attempt to store status for the faulted ORB before transitioning the fetch agent to the dead state. This notifies the initiator that error recovery is necessary. If an unrecoverable transaction error occurs for a write request addressed to an initiator's *status_FIFO*, the target shall take no additional action. It is the initiator's responsibility to detect such an error, usually by means of a timeout.

After a task set has been aborted, an initiator's client applications and services may resume data transfer with the target's services and client applications on a queue by queue basis.

Data transfer between a client application and a service may have caused device operations to commence even if not all the data had been transferred before the task set was aborted. For this reason, it is essential for each queue to be resumed by ~~one of two methods. The simplest is to abandon any operations in progress, flush initiator and target buffers as necessary and return both endpoints of the queue to a known state – at which point the abandoned operation(s) may be reinitiated. The other approach is more efficient and uses~~ using execution context for active ORBs to permit resumption of data transfer at the point at which it was interrupted.

NOTE – A prerequisite to the resumption of data transfer is the existence of a login (the initiator reconnects to the target if there was a bus reset) and a reset fetch agent (the initiator writes to AGENT_RESET if the target's fetch agent had been left in the dead state after the task set was aborted).

~~An initiator may implement simple recovery for a queue by signaling an ORB whose *signature* differs from those of all ORBs active at the time the task set was aborted.~~

~~Although this method is robust, it may be improved upon.~~ If the client application and service can reliably resume data transfer from the point it was interrupted, it may be unnecessary to cancel operations and flush buffers. In order for this method to work, the transport must be able to recognize resumption of an ORB active at the time the task set was aborted. The *signature* field in a transport flow ORB (see 5.1) provides a method by which previously active ORBs may be recognized if they are resubmitted after a task set abort.

~~If the initiator elects not to reset the queue by signaling an ORB with a fresh *signature* value, data~~ Data transfer may be safely resumed if initiator and target can identify, for each queue, the ORBs active at the time the task set was aborted. For a particular queue, an initiator considers an ORB to be active if no status has been received from the target while a target considers an ORB active until positive acknowledgment of the receipt of status is signaled by the initiator. When an initiator wishes to resume data transfer for a particular queue from the point at which it was interrupted, it shall perform the following steps:

a) If there were no active ORBs in the task set for the queue to be resumed, no action is necessary and the initiator may resume data transfer for the queue;

b) Otherwise, for each previously active ORB for the queue, the initiator shall signal an equivalent ORB to the target fetch agent. Certain parts of the ORB shall remain unchanged: the *direction*, *special* and *end_of_message* bits and the *queue* and *signature* fields shall have the same values both before and after the task set abort. The *data_descriptor*, and *data_size* fields and the *page_table_present* may have different values but they shall describe a buffer of the same size and whose contents are identical to the buffer described by the ORB at the time it was aborted. If a bus reset caused the task set to be aborted, the *spd* and *max_payload* bits may differ as a result of changed topology between the initiator and target. An initiator shall signal equivalent ORBs in the same relative order within a queue as they had been prior to the task set abort.

c) Once all the previously active ORBs for a particular queue have been signaled, the initiator may signal new ORBs in any order; these shall be interpreted by the target as if they are new; there are no restrictions on their field values.

When the target fetches an ORB, the action taken depends upon the value of the *queue* and *signature* field, which together uniquely identify an execution context for the initiator. If the value of *signature* is equal to the signature of an ORB active for the queue at the time the task set was aborted, the target shall discard execution context information for any older, previously active ORBs for the same queue. An ORB is older than another ORB if it was signaled before the other ORB. If the value of the *signature* field is not equal to any previously active ORB for the queue, the target shall discard all execution context information for that queue.

When the *signature* field identifies execution context for a previously active ORB, the target operations are determined by the data transfer state at the time the task set was aborted. If the data transfer had completed, successfully or in error, and completion status had been written to the initiator's *status_FIFO* (but no response had been received from the initiator), the target simply stores the same completion status again. The target shall maintain context information for the ORB until the conditions specified by 7.4 are met. Otherwise, when data transfer had been in progress, the target shall resume data transfer from the point specified by the execution context for the ORB.