An error is a manifestation of a fault that can lead to a failure.

Failure Recovery:
- backward recovery
- operation-based (do-undo-redo logs)
- state-based (checkpoints)
- forward recovery

Domino Effect
Cases:
- X fails after \( x_1 \)
- Y fails after sending message \( m \)
- Z fails after sending message \( n \)

Orphan Message
The checkpoint set \( \{ x_1, y_1 \} \) inconsistent

Lost Messages
The checkpoint set \( \{ x_1, y_1 \} \) inconsistent

Strongly Consistent Checkpoints
The set of checkpoints \( \{ x_1, y_1, z_1 \} \) are strongly consistent - no recovery ever needs rollback past this set. The set stops the domino effect. \( \{ x_2, y_2, z_2 \} \) is a consistent set.

Checkpoints Notation
Each node maintains:
- a monotonically increasing counter with which each message from that node is labelled.
- records of the last message from and the first message to all other nodes.

```
last_label_rcvd[x] 
first_label_sent[y]
```

\( m.1 \) (a message \( m \) and its label 1)

Note: "sl" denotes a "smallest label" that is < any other label and "ll" denotes a "largest label" that is > any other label

Checkpoints Algorithm
1. When must I take a checkpoint?
2. Who else has to take a checkpoint when I do?

```
last_label_rcvd[y] >= first_label_sent[x] > sl
```

(2) Any other process from whom I have received messages since my last checkpoint.

\( \text{ckpt cohort} = \{ y \} | \text{last_label_rcvd[y]} > sl \)
**Checkpoint Algorithm**

**Initiator Process** $P_i$:

- for all $p \in \text{ckpt_cohort}(P_i)$,
  - send $\text{TakeTentativeCheckpoint}(P_i, \text{last_label_rcvd}_P_i[p])$;
- if all cohorts replied "yes" then
  - for all $p \in \text{ckpt_cohort}(P_i)$
    - send $\text{MakeCheckpointPermanent}$;
- else
  - for all $p \in \text{ckpt_cohort}(P_i)$
    - send $\text{UndoTentativeCheckpoint}$;