HW #2 Software Safety Requirements & Design Analysis

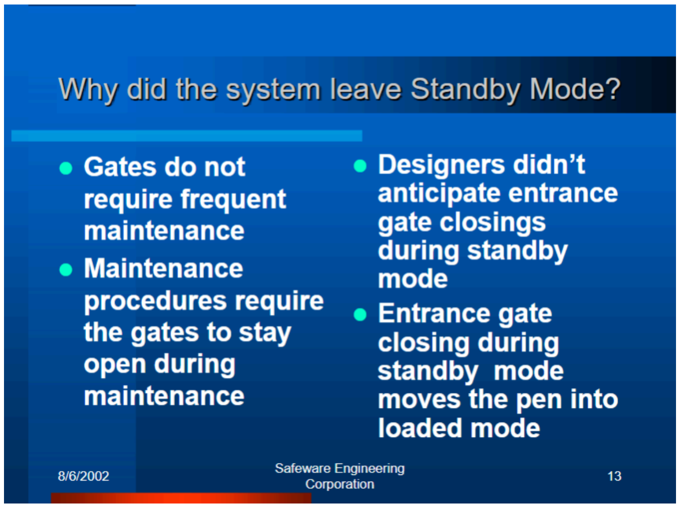
Com S 4150/5150, Spring, 2025
Due by 10 pm, Thursday, Feb. 27; turn in on Gradescope as a pdf
Benjamin Smith, Jaret Van Zee, Gurumanie Singh Dhiman, Jonathan Tan

Re-read the Homework Policy at the top of Homework 1, and make sure that you understand it before beginning this assignment.

Reading assignment:
- 7.3 in textbook
- Chapter 15 of Leveson, Safeware: System Safety and Computers (posted below Lecture 5)
- For Problem 1: Howard and Anderson slides (posted on Canvas)
- For Problem 2: Appendix D (posted on Canvas) of Safeware

1. Checking Requirements Completeness (50 pts.) Read Chap. 15, posted on Canvas. Consider the specification of the system described in the Electroshear presentation by Howard and Anderson, 2002 (posted on Canvas). For each requirement completeness problem they find in the Electroshear (e.g., those described on slides 13, 16, 19, 22, 24, 26, 28), state the "rule" in Chap. 15 that it violates, and briefly justify your answer. (By "rule", I mean the italicized statements prefixed in Chap. 15 by a ☐.)

Black is used for the main answer, other colors are additional inputs on the same data and may not be the perfect answer.

- 

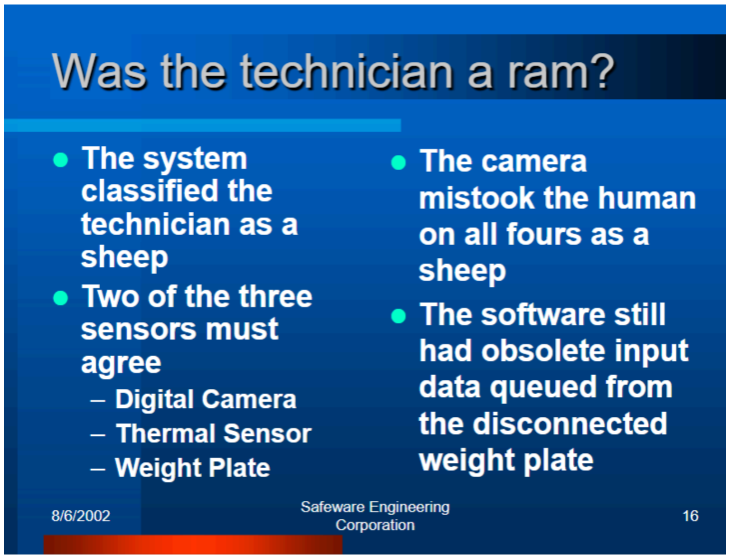
  Why did the system leave Standby Mode?

  - Gates do not require frequent maintenance
  - Maintenance procedures require the gates to stay open during maintenance
  - Designers didn't anticipate entrance gate closings during standby mode
  - Entrance gate closing during standby mode moves the pen into loaded mode

  8/6/2002 Safeware Engineering Corporation 13

  - Rule:
    - *The behavior of the state machine should be deterministic (only one possible transition out of a state is applicable at any time)*
    - *Spontaneous receipt of a nonspontaneous input must be detected and responded to as an abnormal condition*
    - *Automatic update and deletion requirements for information in the human-computer interface must be specified.*

        - *Every state must have a behavior (transition) defined for every possible input (15.4.4.1)*
    - Justification:
        - The rule defines nondeterminism as a case where a condition can cause the transition into 2 different states. This is displayed most accurately in the slide above because, in our case, the standby signal was high (meaning set to standby state) + entry gate was closed (meaning set to loaded state). This could have been avoided if both of those conditions were disjoint to ensure that 2 transition conditions can never be true at the same time.
        - Skipped loading mode and went straight to loaded
        - Even if the designers were unable to anticipate that entrance gates could close during standby mode, there must still be some form of check conducted for the weight plate that was disconnected. This violates the rule that states that automatic update and deletion requirements for information must be specified because even though the entrance gate closing caused the pen to move into loaded mode, there should have been subsequent checks to ensure that all other inputs were in a normal condition before leaving the standby mode. The rule further elaborates on this by stating that events placed in queues may be negated by subsequent events and that the requirements should specify the condition under which this can occur. Since in this case the subsequent event was the gate closing, the requirements need to state specifically why/when the standby mode will move to loaded mode even though the weight plate is disconnected.
        - Gate close + standby mode is not properly defined in the system, causing the system to change to an unexpected state.

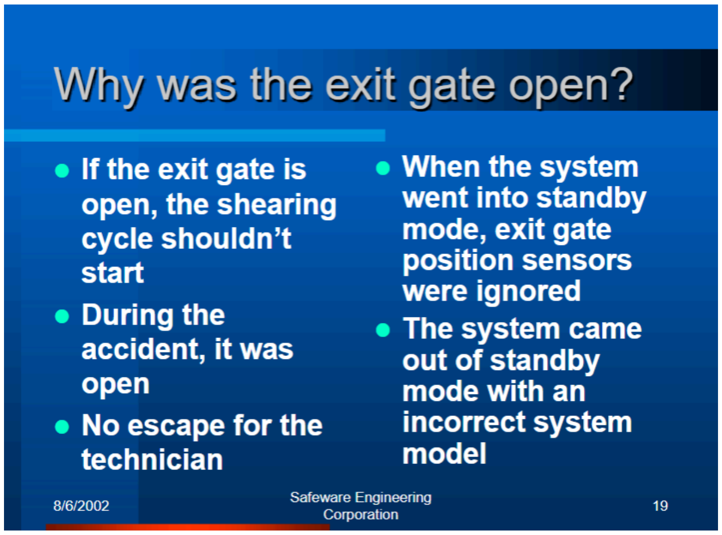
- 

Was the technician a ram?

- The system classified the technician as a sheep
- Two of the three sensors must agree
  - Digital Camera
  - Thermal Sensor
  - Weight Plate
- The camera mistook the human on all fours as a sheep
- The software still had obsolete input data queued from the disconnected weight plate

8/6/2002 Safeware Engineering Corporation 16

  - Rule:
    - *All inputs used in specifying output events must be properly limited in the time they can be used (data age). Output commands that may not be able to be executed immediately must be limited in the time they are valid.*
    - *All information from the sensors should be used somewhere in the specification*
  - Justification:
    - The software used obsolete data (data from when the weight plate was still active) and used it for an output command. The obsolete data needs to be flagged and discarded after a validity time that needs to be set in the requirements.
    - The entrance gate closing during standby mode moved the pen into loaded mode but did not consider the exit gate, or the weight plate as inputs before doing so. Since the weight plate was disconnected and the exit gate was open, if they were also used as inputs to determine the state change from standby to load, the incident would not have taken place.

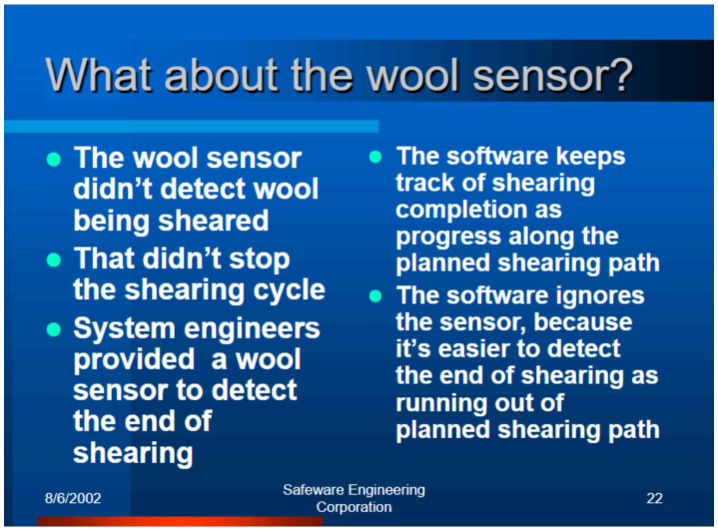
- 

Why was the exit gate open?

- If the exit gate is open, the shearing cycle shouldn't start
- During the accident, it was open
- No escape for the technician
- When the system went into standby mode, exit gate position sensors were ignored
- The system came out of standby mode with an incorrect system model

8/6/2002 Safeware Engineering Corporation 19

  - Rule:
    - *The internal software model of the process must be updated to reflect the actual process state at internal startup and after temporary shutdown (15.4.2)*
    - *The behaviour of the software with respect to inputs received before startup, after shutdown, or when the computer is temporarily disconnected from the process (off-line) must be specified, or it must be determined that this information can be safely ignored, and this conclusion must be documented.*
    - *The logical OR of the conditions on every transition out of any state must form a tautology.*
  - Justification:
    - The rule states that it is important to consider that the process continues to change state even when the computer is not executing. In the slide above, the internal software model was not made aware/updated to reflect the actual process after standby mode (exit gate position sensors were ignored) and the system exited standby mode with an incorrect system model. In summary, after the unexpected state change from standby, the system failed to recognize that the exit gate was open.
    - In the above case, exit gate position sensors were ignored, which violates the above rule that states that the behavior of the software needs to be specified, or it must be determined that the information can be safely ignored. The exit door being open is not an input that can be safely ignored and furthermore, the weight plate was also disconnected, which is another input that was ignored. Both of these, in conjunction, cannot possibly justify that the information can be ignored safely.
    - In this case, because the exit gate position sensors were ignored, but the system entered sheer mode, it shows that the logical OR of the conditions on the transition out of standby mode didn't form a tautology.

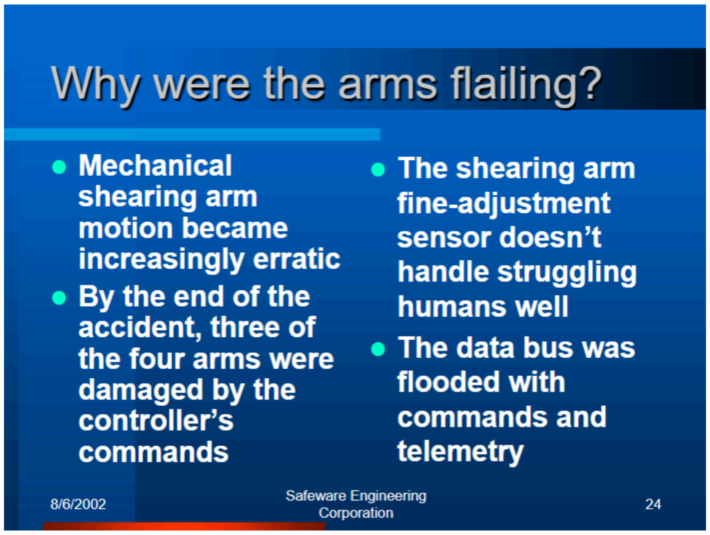


What about the wool sensor?

- The wool sensor didn't detect wool being sheared
- That didn't stop the shearing cycle
- System engineers provided a wool sensor to detect the end of shearing
- The software keeps track of shearing completion as progress along the planned shearing path
- The software ignores the sensor, because it's easier to detect the end of shearing as running out of planned shearing path

8/6/2002 Safeware Engineering Corporation 22

- Rule:
  - *All information from the sensors should be used somewhere in the specification*
  - *The behaviour of the software with respect to inputs received before startup, after shutdown, or when the computer is temporarily disconnected from the process (off-line) must be specified, or it must be determined that this information can be safely ignored, and this conclusion must be documented.*
- Justification:
  - The rule states that if an input can be sent to the computer, there should be some specification of what should be done with it. In the above slide, the wool sensor did not detect wool but its input was not taken into consideration and the process continued anyway.
  - The wool sensor did not detect wool being sheared, but the process continued because the software chose to ignore the sensor. The sensor is there for a specific reason, if the software chooses to ignore the sensor, it must ensure that it can be ignored safely and documented. The rule mentioned above was not conformed with in this case because the software had access to the data from the wool sensor, yet chose to ignore this information without confirmation that it was safe.

- 

Why were the arms flailing?

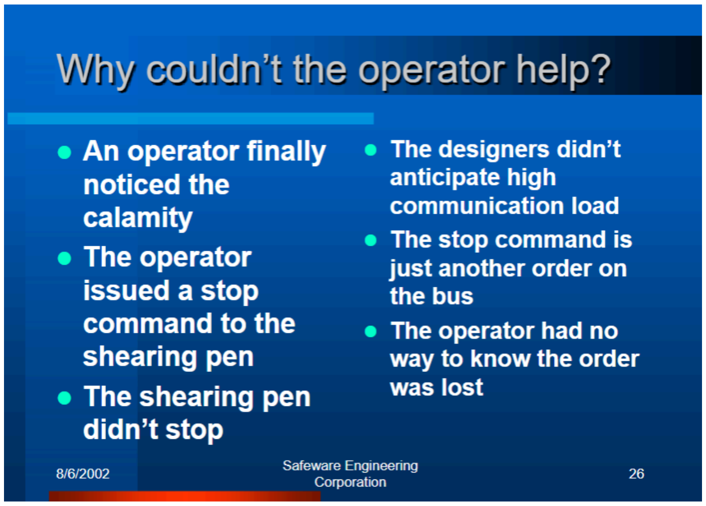
- Mechanical shearing arm motion became increasingly erratic
- By the end of the accident, three of the four arms were damaged by the controller's commands
- The shearing arm fine-adjustment sensor doesn't handle struggling humans well
- The data bus was flooded with commands and telemetry

8/6/2002 Safeware Engineering Corporation 24

  - Rule:
    - *For the largest interval in which both input and output loads are assumed and specified, the absorption rate of the output environment must equal or exceed the input arrival rate.*
    - *Safety-critical outputs should be checked for reasonableness and for hazardous values and timing. (15.4.5)*
    - *Incomplete hazardous action sequences (transactions) should have a finite time specified after which the software should be required to cancel the sequence automatically and inform the operator. (15.4.5)*
    - *Basic feedback loops, as defined by the process control function, must be included in the software requirements. That is, there should be an input that requirements must include appropriate checks on these inputs in order to detect internal or external failures or errors. (15.4.6)*
  - Justification:
    - As specified in the rule above, the environmental capacity problem allowed the actuators to be overwhelmed with data on the command bus and damage themselves.
    - As mentioned in the slide above, the shearing arm sensor does not handle struggling humans well, which indicates that the values must have been erratic and highly inconsistent, yet the software did not flag this as unreasonable or hazardous. This violates the rule above because it clearly indicates that even though the data bus was flooded with commands and inconsistent values, the software did not check for reasonableness.
    - As mentioned above in the last rule, there must be a finite time specified, after which the software should be required to cancel the sequence automatically and inform the operator. This was not the case in the slide above because the data bus was flooded with all sorts of commands and telemetry in a short period of time, yet the software took no action to inform the operator or cancel the sequence. This violates the rule

because it indicates that there is a flaw in the software that is unable to comprehend that all the information flooding the data bus is incorrect and erratic and to take countermeasures.

- If the shearing arms had basic feedback loops, it would have detected that the shearing arm's fine-adjustment sensors were detecting irregular shearing patterns.

- 

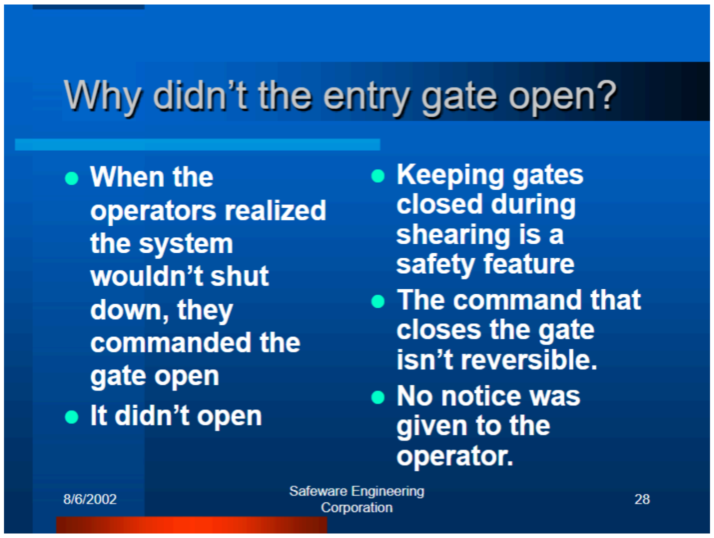
Why couldn't the operator help?

- An operator finally noticed the calamity
- The operator issued a stop command to the shearing pen
- The shearing pen didn't stop
- The designers didn't anticipate high communication load
- The stop command is just another order on the bus
- The operator had no way to know the order was lost

8/6/2002 Safeware Engineering Corporation 26

  - Rule:
    - Basic feedback loops, as defined by the process control function, must be included in the software requirements. That is, there should be an input that the software can use to detect the effect of any output on the process. The requirements must include appropriate checks on these inputs in order to detect internal or external failures or errors. (15.4.6)
    - *Multiple paths should be provided for state changes that maintain or enhance safety. Multiple inputs or triggers should be required for paths from safe to hazardous states.*
    - *The response to excessive inputs (violations of load assumptions) must be specified*
    - *Soft and hard failure modes should be eliminated for all hazard-reducing outputs. Hazard-increasing outputs should have both soft and hard failure modes.*
    - *Transitions must satisfy software system safety requirements and constraints.*
    - *There must be a response specified for the arrival of an input in any state, including indeterminate states. (15.4.2)*
  - Justification:
    - The output feedback problem kept the operators unaware that their stop command was being ignored. The operator's commands to stop the process had no special priority, and were susceptible to being dropped in the high traffic caused due to the telemetry and movement command problem.

    - There should have been multiple inputs or triggers to get from the safe state (standby) to the loaded state, which ended up being hazardous, but there was only one (the entry gate closing)
    - There was no designated response to an overloaded bus of information. There should have been checks in place to make sure that if a stop command was entered, no matter how loaded the bus, it should be able to receive it or know it possibly couldn't receive one in a reasonable amount of time.
    - The shearing state should have multiple logical ways to issue the commands needed to maintain the safety of the system so that a single hardware failure cannot prevent the operator from taking action to avoid a hazard. This was not the case in our situation, as the operator issued a command, but it did nothing, and there were no backup ways to get into a hard failure mode.
    - As mentioned on page 35, after the italicized point (*Transitions must satisfy software system safety requirements and constraints*), it states that every hazardous state must have a path to a safe state. Furthermore, it also states that if a safe state cannot be reached there from a hazardous state, all paths from that state must lead to a minimum risk state and that at least one such path should exist. In our case, there was one path that existed, but when the operator tried to issue a stop command, it did not work. This violates the above rule because there must be a way to reach a safe state from a hazardous state.
    - The slide shown above is an example of a missing response for the input "stop" in any state. Showing that the "stop" input doesn't have the right priority and/or that there exists a state that has no appropriate response for the input.

- 

Why didn't the entry gate open?

- When the operators realized the system wouldn't shut down, they commanded the gate open
- It didn't open
- Keeping gates closed during shearing is a safety feature
- The command that closes the gate isn't reversible.
- No notice was given to the operator.

8/6/2002 Safeware Engineering Corporation 28

  - Rule:
    - *Output commands should usually be reversible. (15.4.7)*

        - *Every hazardous state must have a path to a safe state. All paths from the hazardous state must lead to a safe state. Time in the hazardous state must be minimized and contingency action may be necessary to reduce risks while in the hazardous state.*
        - *Safety-critical outputs should be checked for reasonableness and for hazardous values and timing.*
    - Justification:
        - The rule clearly states that there needs to be a way to cancel/reverse an "ON" command using an "OFF" command from the state in which the "ON" command was issued. In our case, since there was an emergency, the command sent by the operator was not reversible (opening the gate), which kept the operators from being able to provide an escape route during the accident.
        - Again, this shows that incorrect "stop" command priority and/or that there exists a state that has no appropriate response for it, let to the irreversibility of the shear command.
        - A shutdown command was given, the operator tried commanding the gate to open, and there were hazardous values from within the machine while the person was still inside. All of these should have been considered. Even though keeping the gates closed is a safety feature, there must be a check for reasonableness when there are so many flags being received at once. At least one of the above should have put the machine in a safe state due to the number of hazardous values and the fact that these commands were all given one after another in a short period of time.

2. Safety Requirements for information in the human-computer interface (50 pts.) In Leveson's Safeware, read Appendix D.3., pp. 619-639. Leveson quotes Hoagland: "The fact that a system or process is being controlled automatically does not mean that the pilot's information needs are any different than if being controlled manually." That holds for the operator's information, too.

In the Three Mile Island accident,
(a) Describe what specific information was not available to the operators that they needed.

- Temperatures from the reactor coolant drain tank, while temperature information was presented, it was often deemed to be inaccurate due to known issues of leakage from PORV or another valve
- The twelve-valves were not open when they were supposed to be
- The origin of the water coming into the containment building sump
- Steam and water were getting pumped through the pipes, the operators were not aware that the water was getting boiled into steam.
- The direct source of the hydrogen explosion from building pressure in the tank.
- The control room alarms did not provide potential reasons as to why the alarms were going off
- Water pressure information in the core was not provided to the operators

- Operators were unable to receive updated information at different times due to the computers running behind occurring events.
- Operators were not aware of the poor maintenance of the pressurizer level transmitter, hydrogen recombine, pressurizer heaters, makeup pump switches, and condensate polishers
- The operators did not have information on the condition of the iodine filters, which would prevent radioactive contamination from leaking into the air
- Priority of alarms, the operator can't filter lower priority alarms, causing them to have to face more than 100 alarms in a short period of time.
- No direct indication of coolant water turning into steam.
- Operators were not able to see alarms that were poorly placed

(b) What false assumptions did the operators make, due in part to this missing information, that contributed to the accident?

- Assumption of accurate instrumentation: The operators trusted the control room indicators. Turns out the indicators misrepresented key parameters like the coolant level and the state of the relief valve.
- Assumption of sufficient coolant: Due to the indicators' misrepresentation of the coolant level, the operator overlooked that the coolant was escaping.
- Assumption that the incident is contained and isolated: The operators didn't expect that the failure of the relief valve could cascade down and lead to issues in other systems affecting the coolant levels and the system's cooling.
- Assumption that the high temperature of the drain pipe represented residual heat. This should have been an indication to the operators that there was an open PORV and a loss of coolant.
- The assumption that the hydrogen explosion was the sound of the lid slamming shut.