# REQUIREMENTS AND RISKS

Eunsuk Kang

Required reading: "The World and the Machine" M. Jackson (1995).

# LEARNING GOALS

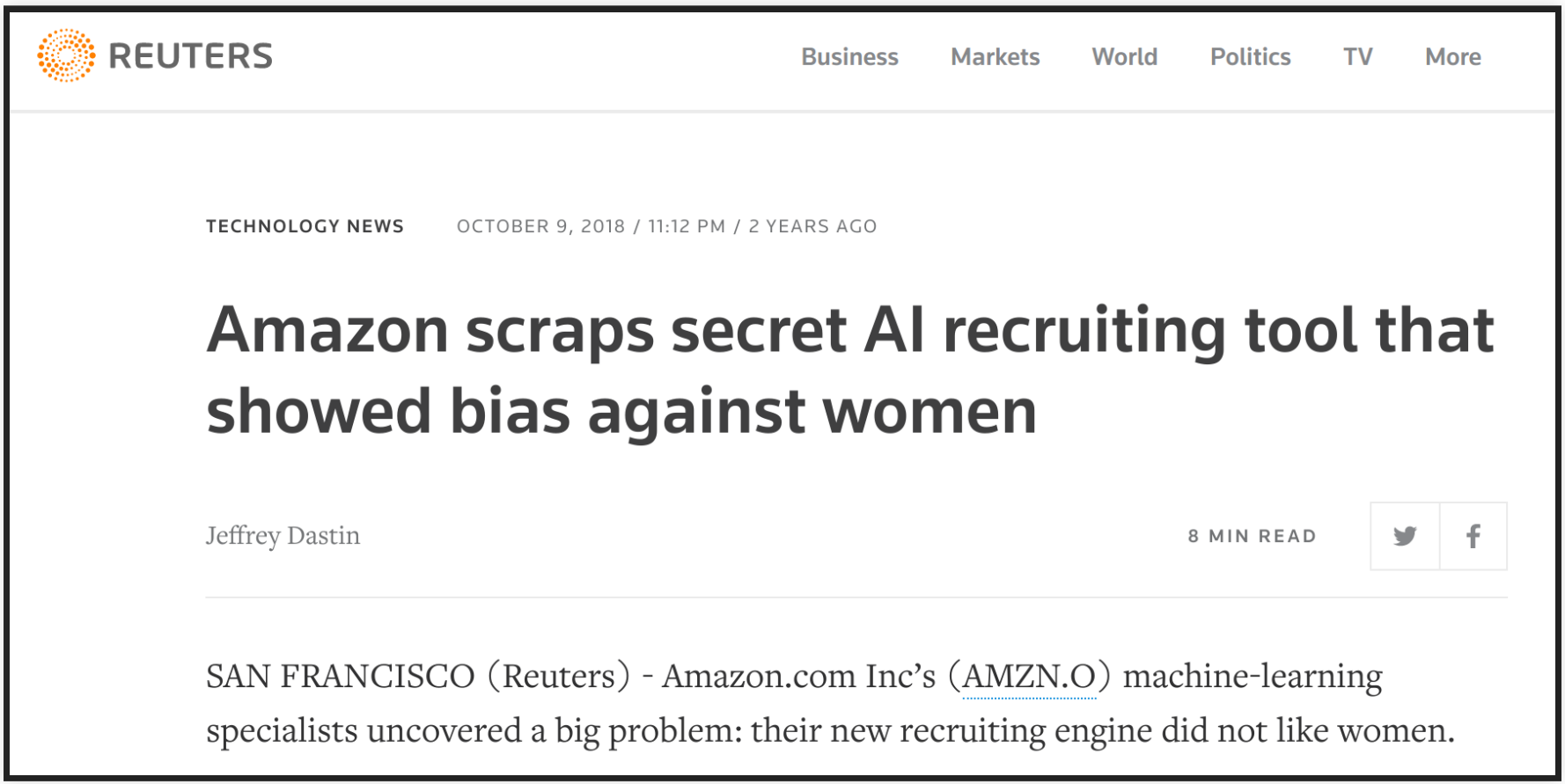
- Understand the role of requirements in ML-based systems and their failures
- Understand the distinction between the world and the machine
- Understand the importance of environmental assumptions in establishing system requirements

# FAILURES IN ML-BASED SYSTEMS

# FACIAL RECOGNITION IN ATM

**Q. What went wrong? What is the root cause of the failure?**

# AUTOMATED HIRING



REUTERS

Business Markets World Politics TV More

TECHNOLOGY NEWS OCTOBER 9, 2018 / 11:12 PM / 2 YEARS AGO

## Amazon scraps secret AI recruiting tool that showed bias against women

Jeffrey Dastin

8 MIN READ

SAN FRANCISCO (Reuters) - Amazon.com Inc's (AMZN.O) machine-learning specialists uncovered a big problem: their new recruiting engine did not like women.

**Q. What went wrong? What is the root cause of the failure?**

# AUTOPILOT IN VEHICLES

U.S. NEWS

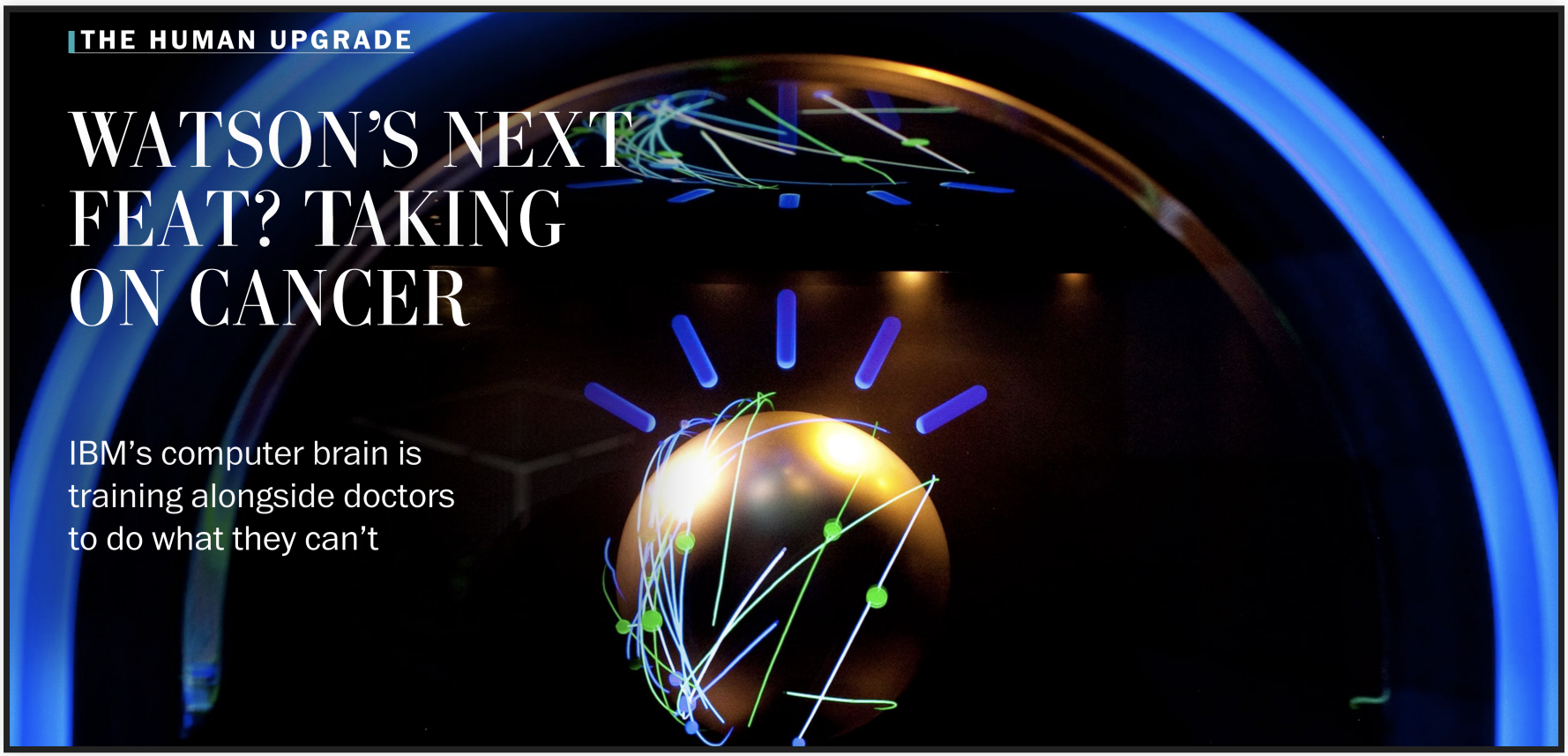
**Tesla driver slept as car was going over 80 mph on Autopilot, Wisconsin officials say**

**Q. What went wrong? What is the root cause of the failure?**

# IBM WATSON


PIENSE
THINK
DENKE
PENSER
$300,000
Who is Stoker?
(( FOR ONE WELCOME OUR NEW COMPUTER OVERLORDS))
$ 1,000
$1,000,000
Who is Bram Stoker?
$ 17,973
$200,000
WHO IS BRAM STOKER?
$5600

# IBM WATSON




Washington Post, 06/2015

# IBM WATSON


*"We got concerns from them that the recommendations that it was giving were just not relevant...it would suggest a particular kind of treatment that wasn't available in the locality in which it was making the recommendation, or the recommendation did not at all square with the treatment protocols that were in use at the local institution..."*

Slate, 01/2022

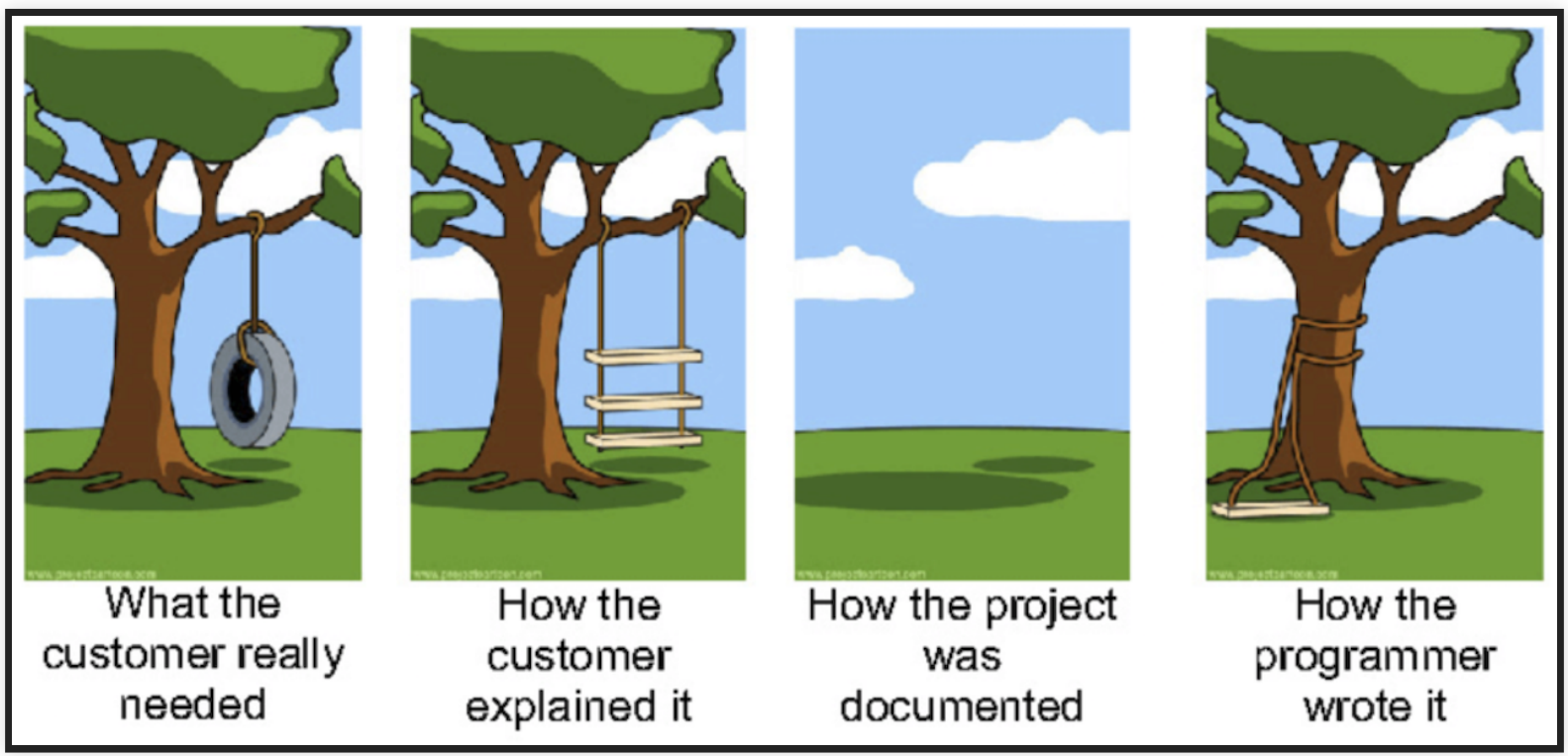
# RISKS IN ML-BASED SYSTEMS

**What went wrong? What were the root causes of failures in these systems? Was the quality of an ML model to blame?**

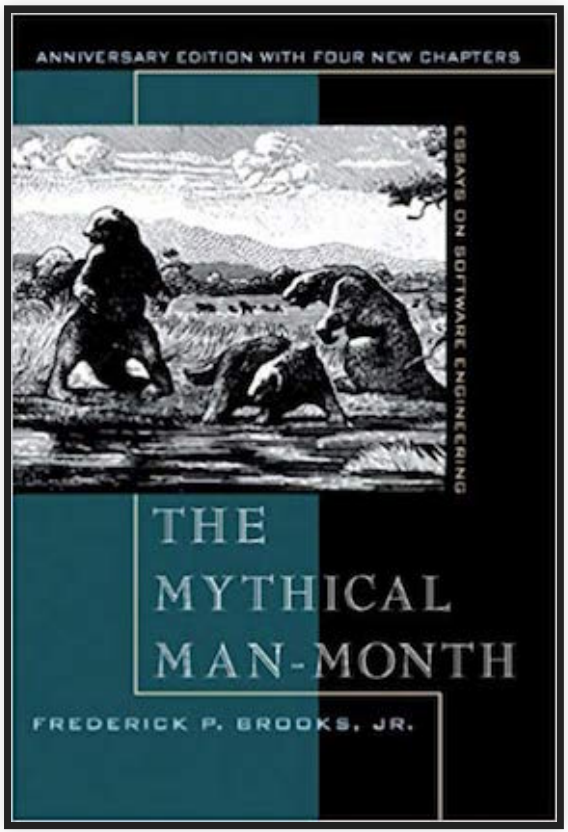
# SOFTWARE REQUIREMENTS

# SOFTWARE REQUIREMENTS

- Describe what the system should do, in terms of the services that it provides and their qualities (safety, reliability, performance...)
- Gathered through discussions with stakeholders (customers, domain experts, marketing team, industry regulators...)

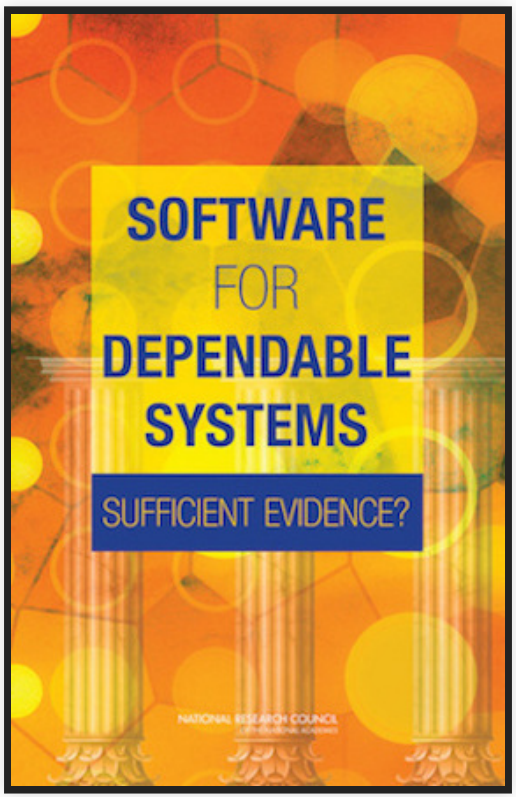


# IMPORTANCE OF REQUIREMENTS



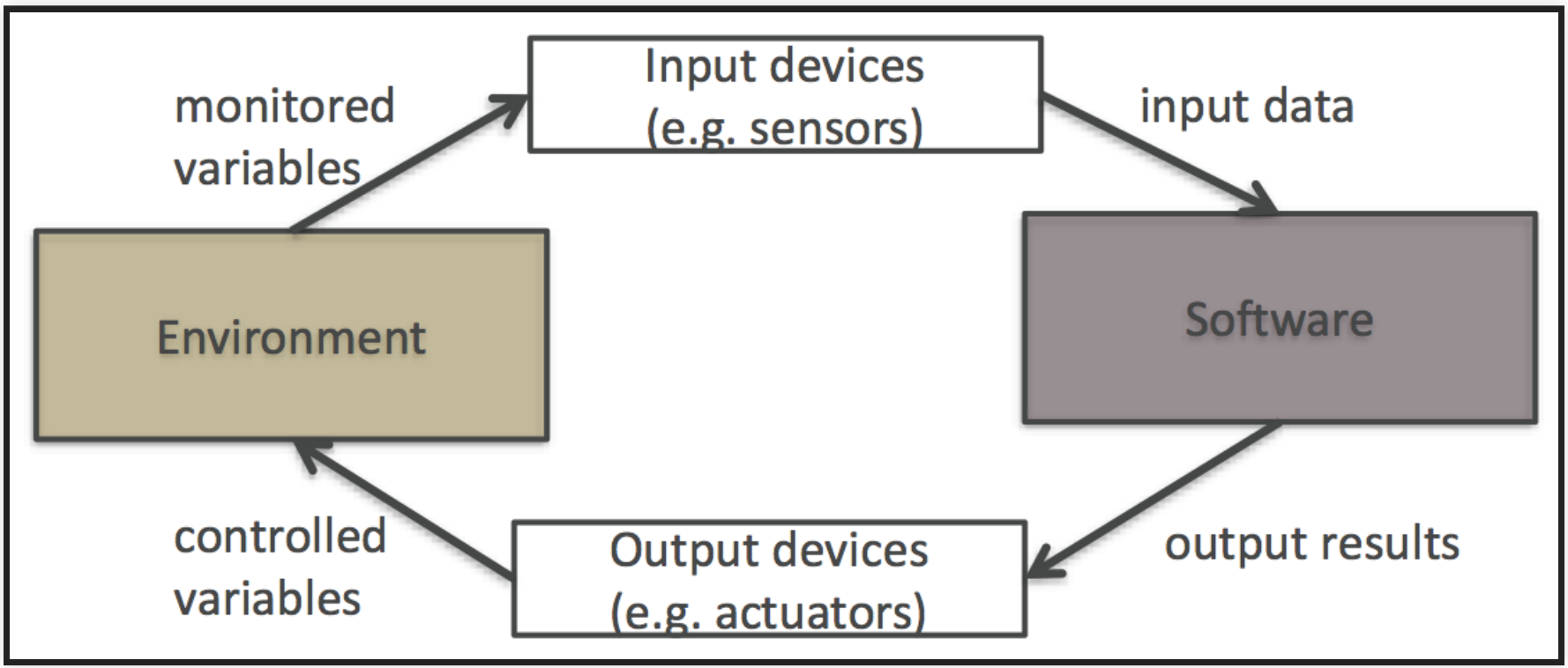

*"The hardest single part of building a software system is deciding precisely what to build...No other part of the work so cripples the resulting system if done wrong."* -- Fred Brooks, Mythical Man Month (1975)

# IMPORTANCE OF REQUIREMENTS

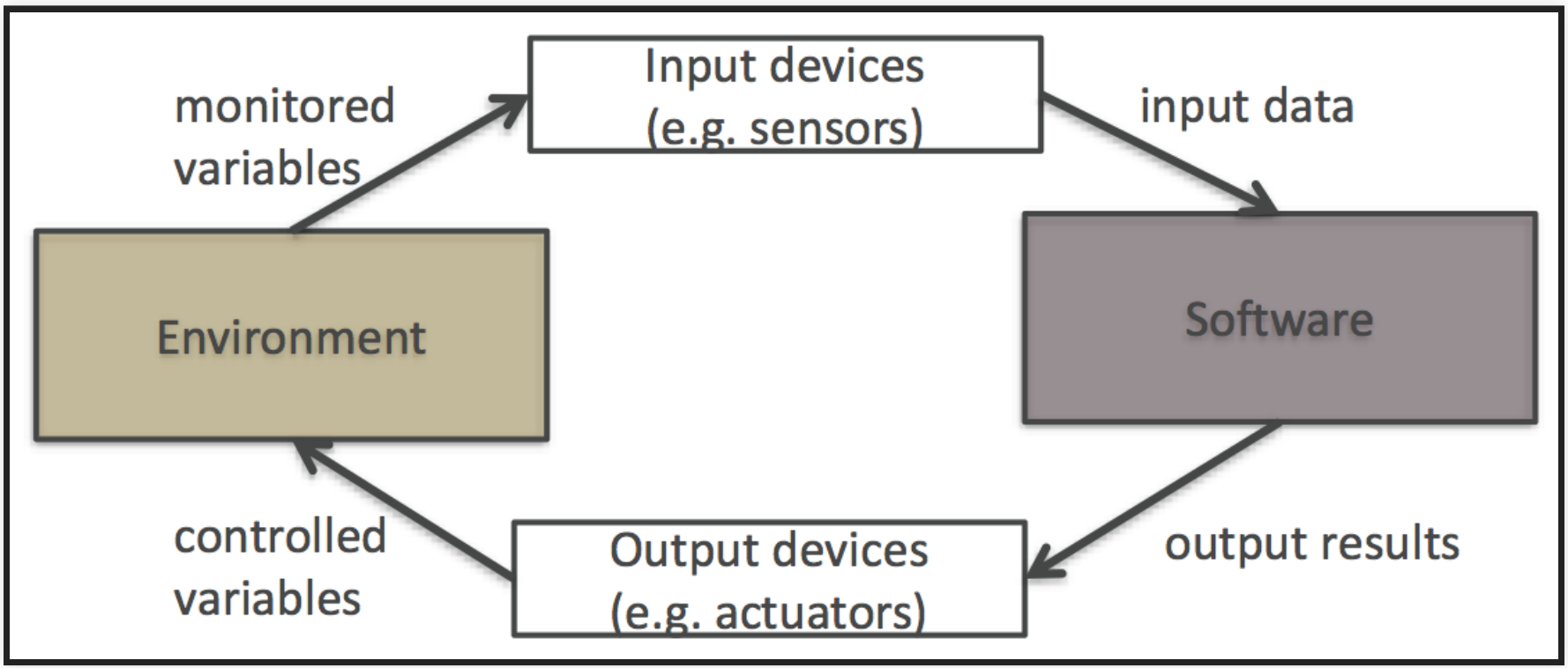



- An investigation of software-related failures by the National Research Council in the US (2007)
- Bugs in code account only for 3% of fatal software accidents
- Most failures due to **poor understanding of requirements** or usability issues

# MACHINE VS WORLD

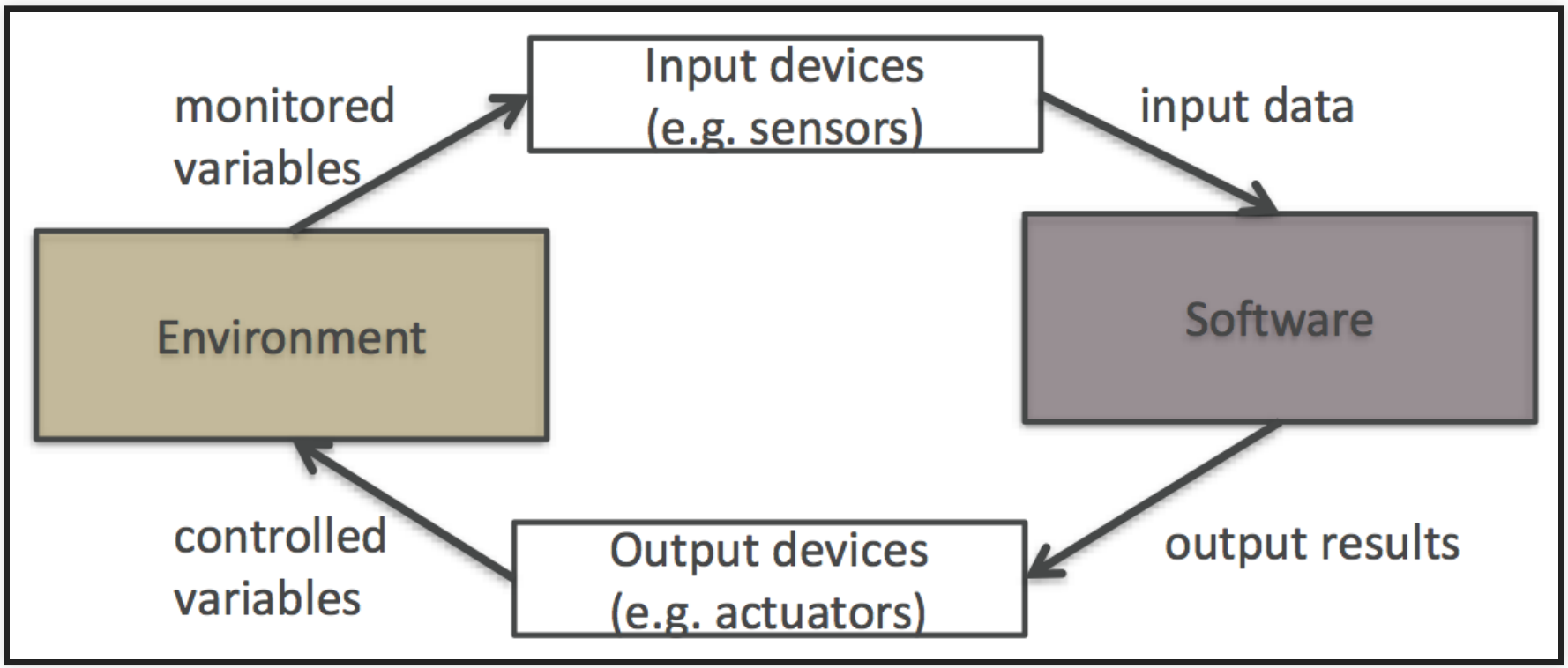


# MACHINE VS WORLD



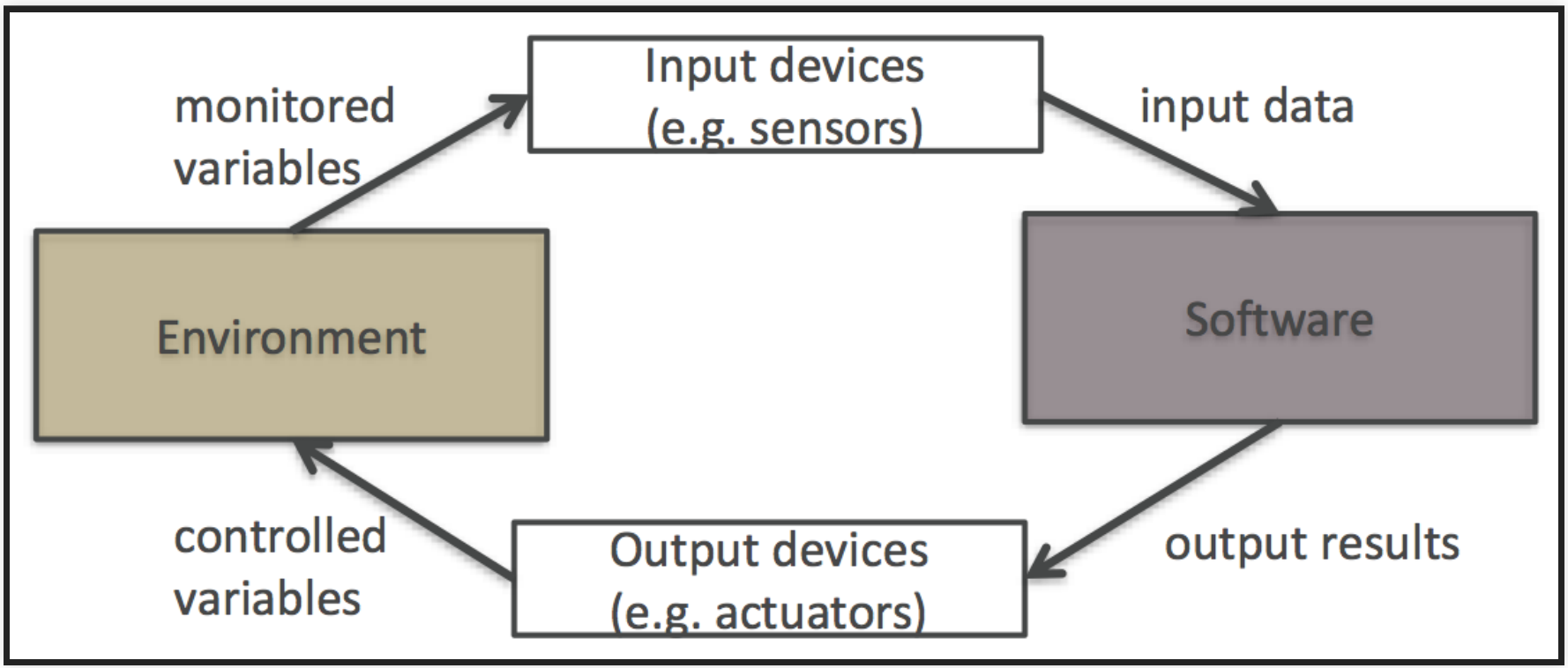

- No software lives in vacuum; every system is deployed as part of the world

# MACHINE VS WORLD



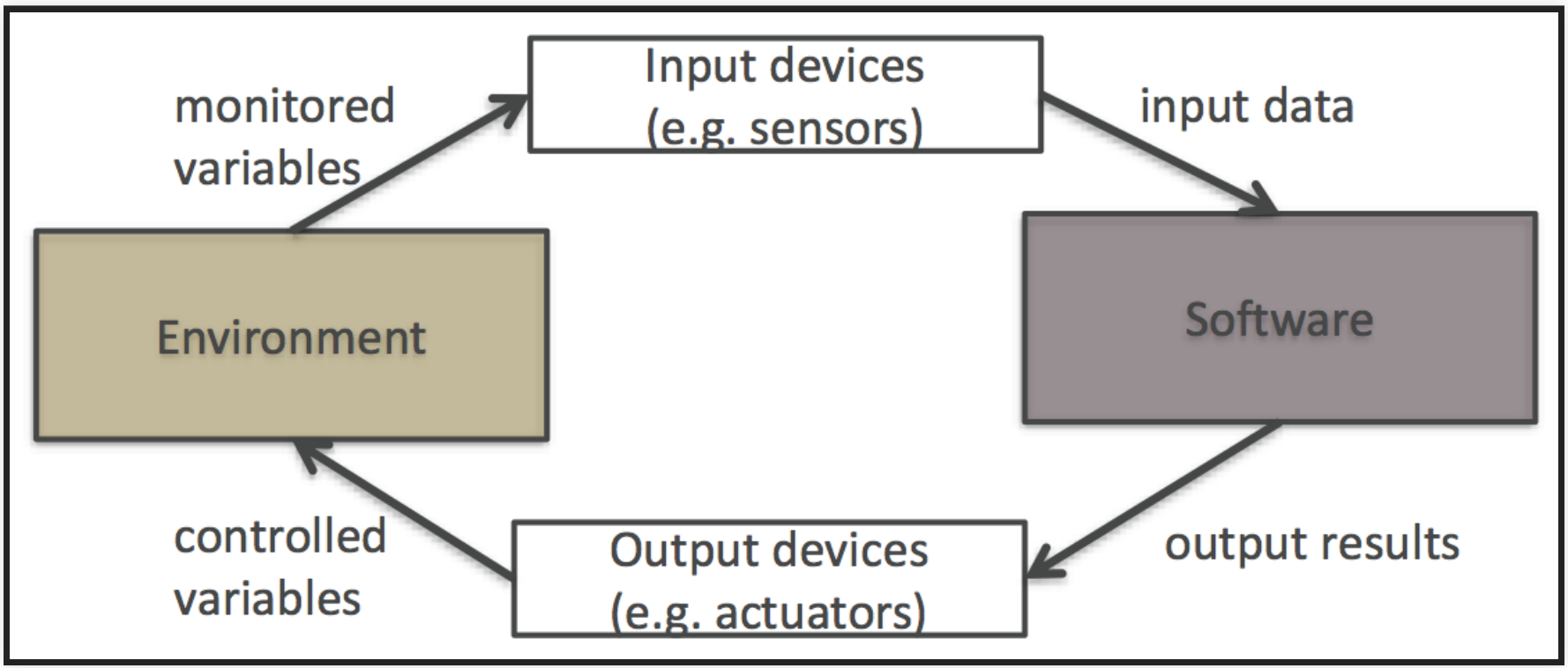

- No software lives in vacuum; every system is deployed as part of the world
- A requirement describes a desired state of the world (i.e., environment)

# MACHINE VS WORLD




- No software lives in vacuum; every system is deployed as part of the world
- A requirement describes a desired state of the world (i.e., environment)
- Machine (software) is designed to sense and manipulate the environment into this desired state using input & output devices

# MACHINE VS WORLD




- Q. In the following systems, what does the environment consist of?
  - Smart home thermostat: ??
  - Movie recommendation system: ??

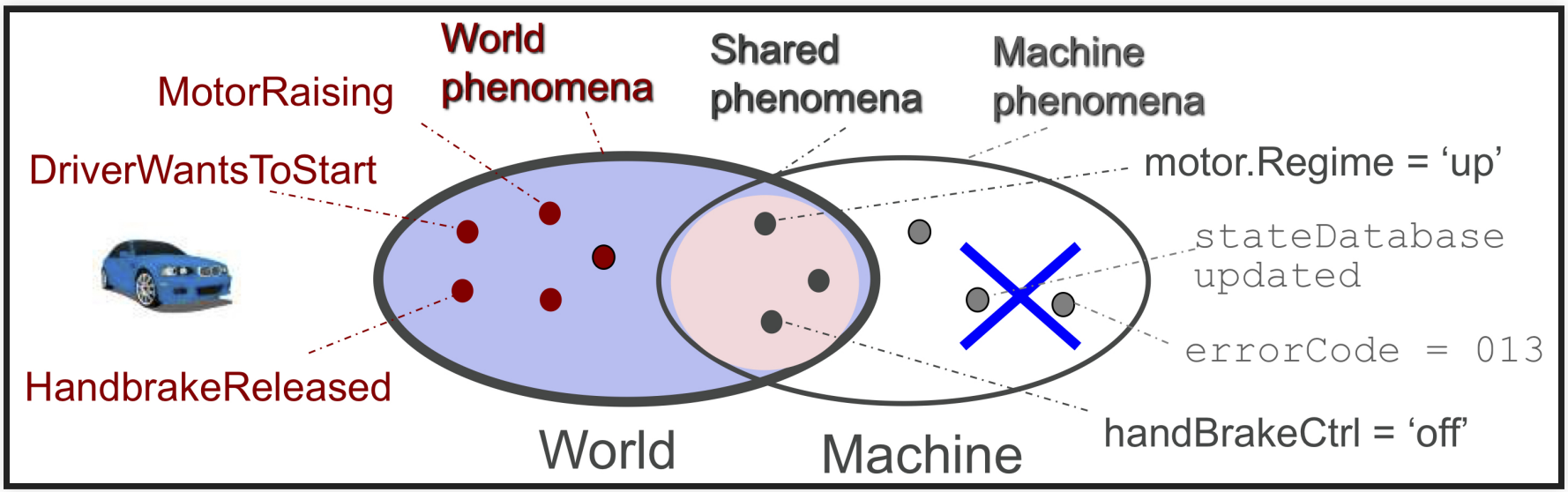
# EXAMPLE: LANE KEEPING ASSIST


- Q. What is the requirement of this system?

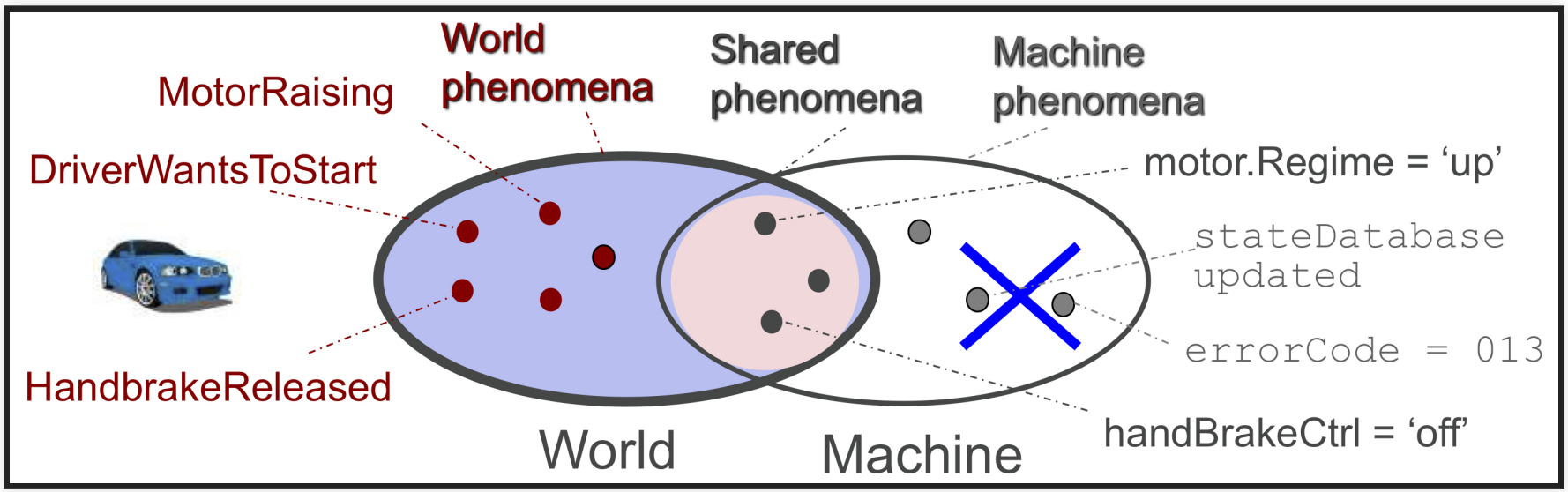
# EXAMPLE: LANE KEEPING ASSIST


- Requirement: The vehicle must be prevented from veering off the lane.
- Q. What does the environment consist of?

# SHARED PHENOMENA

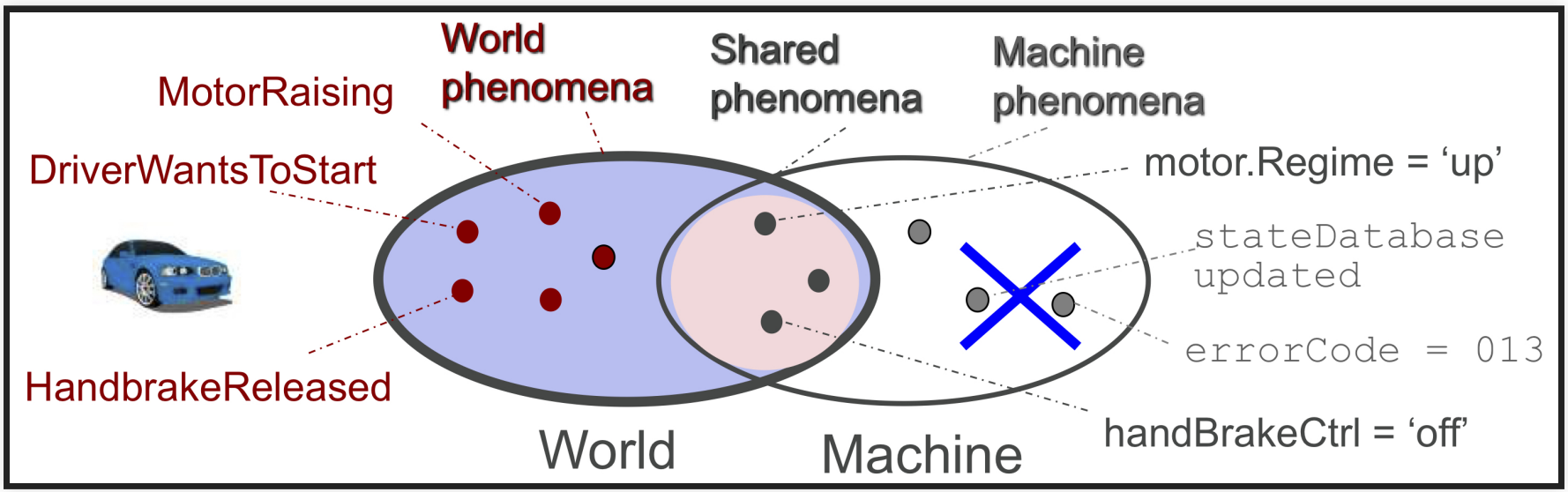


# SHARED PHENOMENA



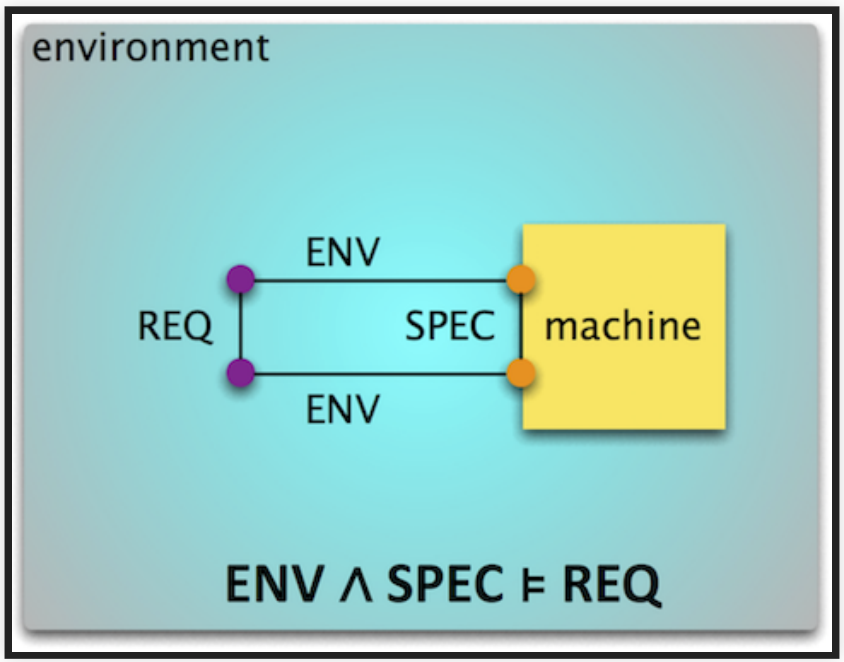

- Shared phenomena: Interface between the environment & software
  - Input: Lidar, camera, pressure sensors, GPS
  - Output: Signals generated & sent to the engine or brake control

# SHARED PHENOMENA



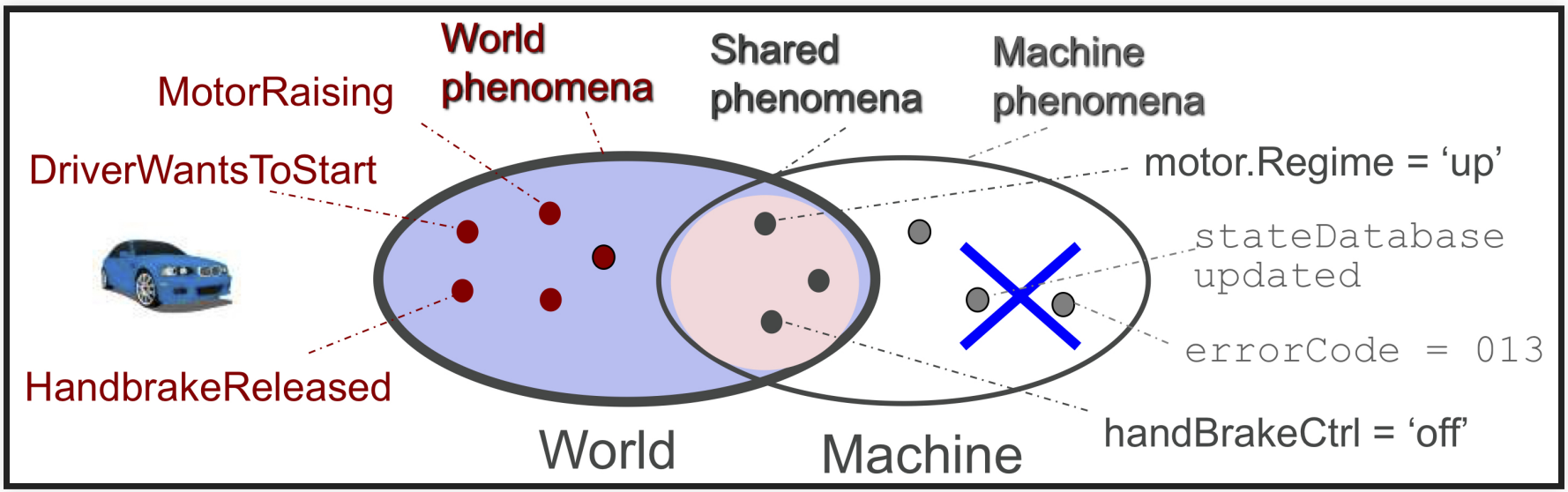

- Shared phenomena: Interface between the environment & software
  - Input: Lidar, camera, pressure sensors, GPS
  - Output: Signals generated & sent to the engine or brake control
- Software can influence the environment **only** through the shared interface
  - Unshared parts of the environment are beyond software's control
  - We can only **assume** how these parts will behave

# REQUIREMENT VS SPECIFICATION




- Requirement (REQ): What the system must ensure, in terms of desired effects on the environment
- Specification (SPEC): What software must implement, expressed over the shared phenomena
- Assumptions (ENV): What's assumed about the behavior/properties of the environment; bridges the gap between REQ and SPEC

# SHARED PHENOMENA




- Requirements (REQ) are expressed only in terms of world phenomena
- Assumptions (ENV) are expressed in terms of world & shared phenomena
- Specifications (SPEC) are expressed in terms of shared phenomena

**Software cannot directly satisfy a requirement on its own; it relies on assumptions about the environment!**

# EXAMPLE: LANE ASSIST


- Requirement (REQ): The vehicle must be prevented from veering off the lane.
- Specification (SPEC): ??

# EXAMPLE: LANE ASSIST

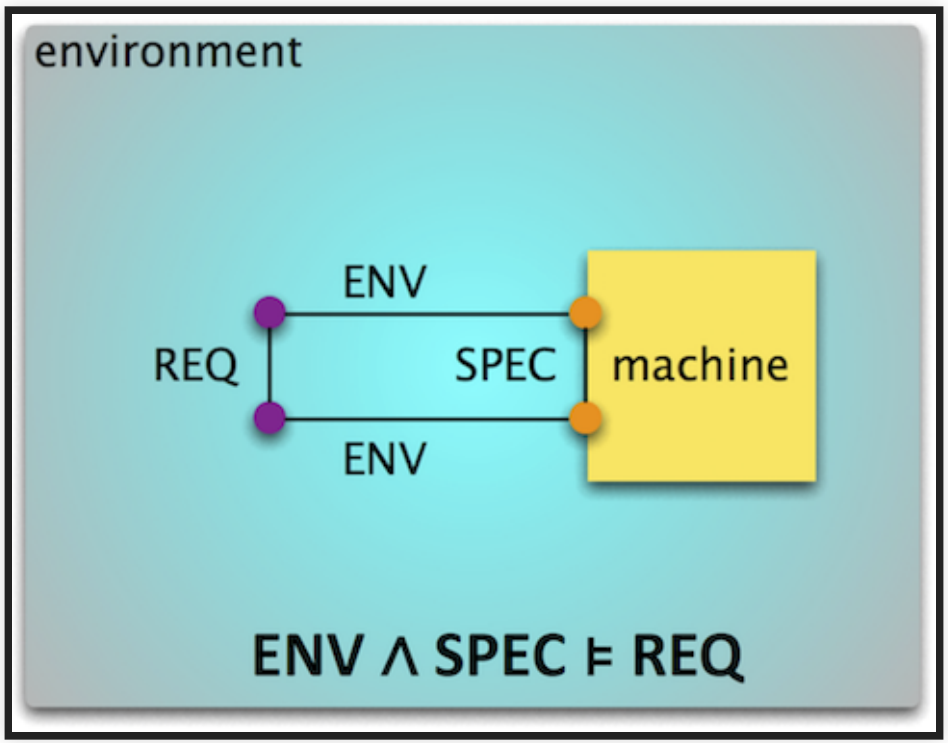

- REQ: The vehicle must be prevented from veering off the lane.
- SPEC: Lane detector accurately identifies lane markings in the input image; the controller generates correct steering commands
- Assumptions (ENV): ??

**Discuss with the person next to you for 1 min & type into Slack**

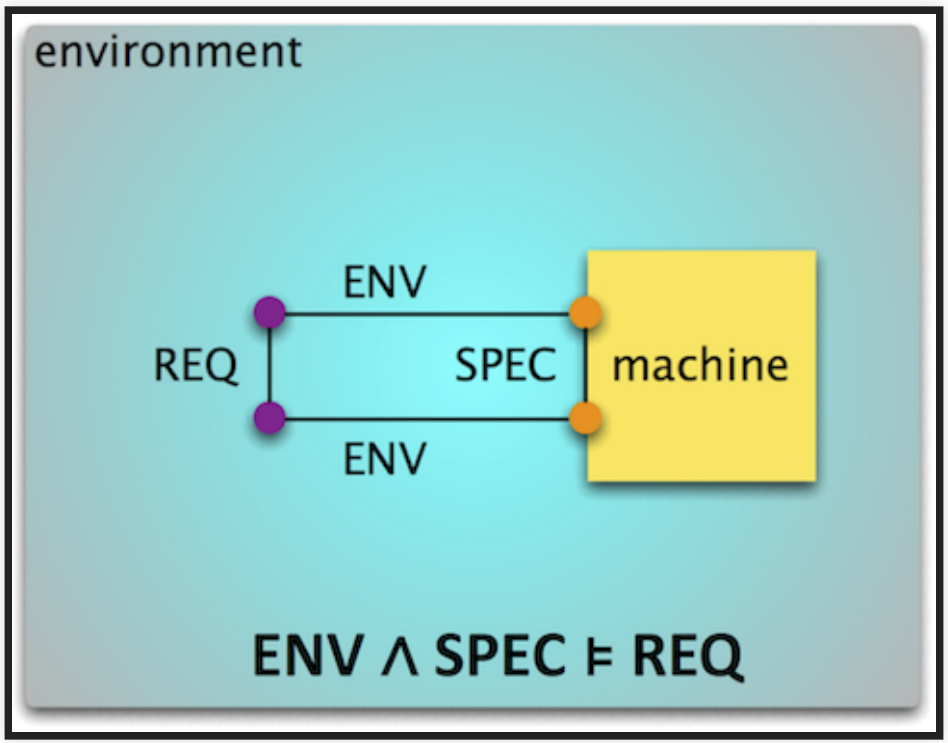
# EXAMPLE: LANE ASSIST


- REQ: The vehicle must be prevented from veering off the lane.
- SPEC: Lane detector accurately identifies lane markings in the input image; the controller generates correct steering commands
- ENV: Sensors are providing accurate information about the lane; driver responses when given warning; steering wheel is functional

# WHAT COULD GO WRONG?

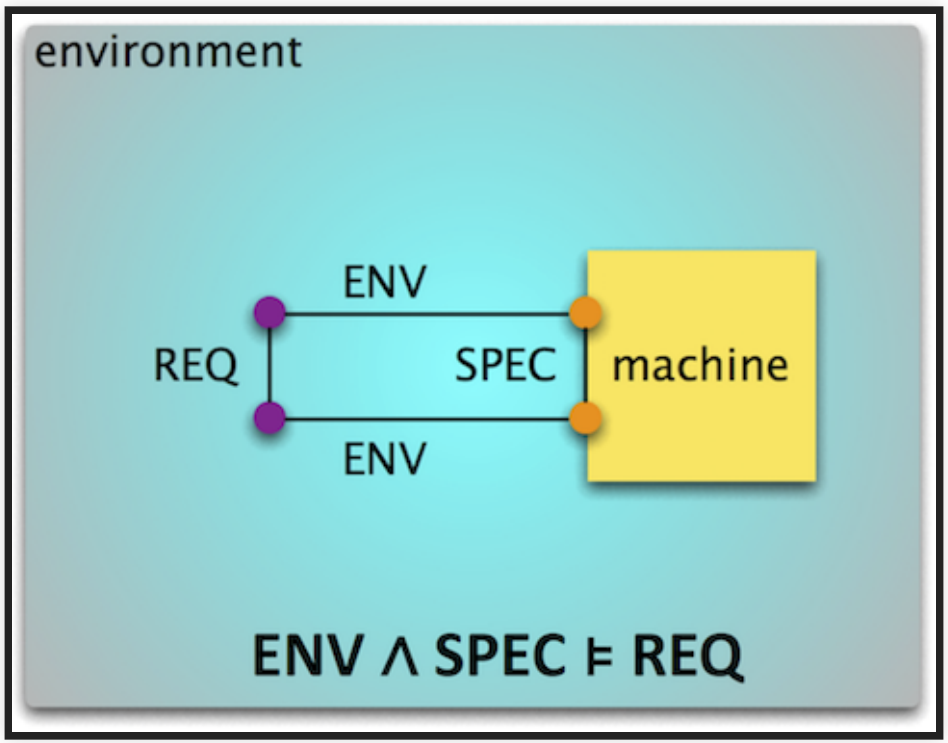


# WHAT COULD GO WRONG?



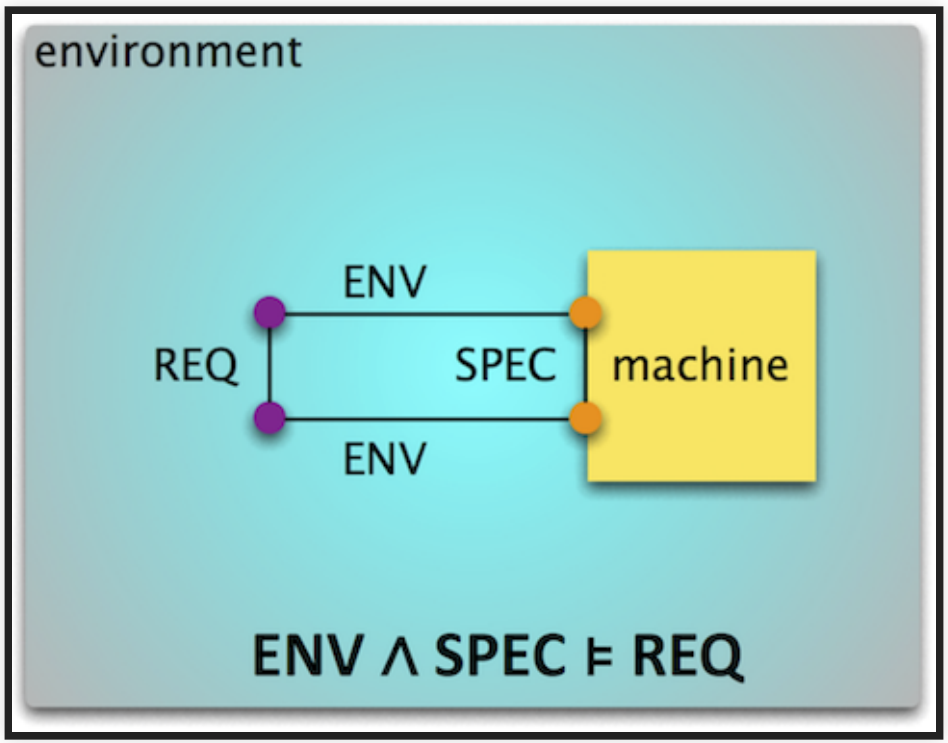

- Wrong, inconsistent or infeasible requirements (REQ)

# WHAT COULD GO WRONG?



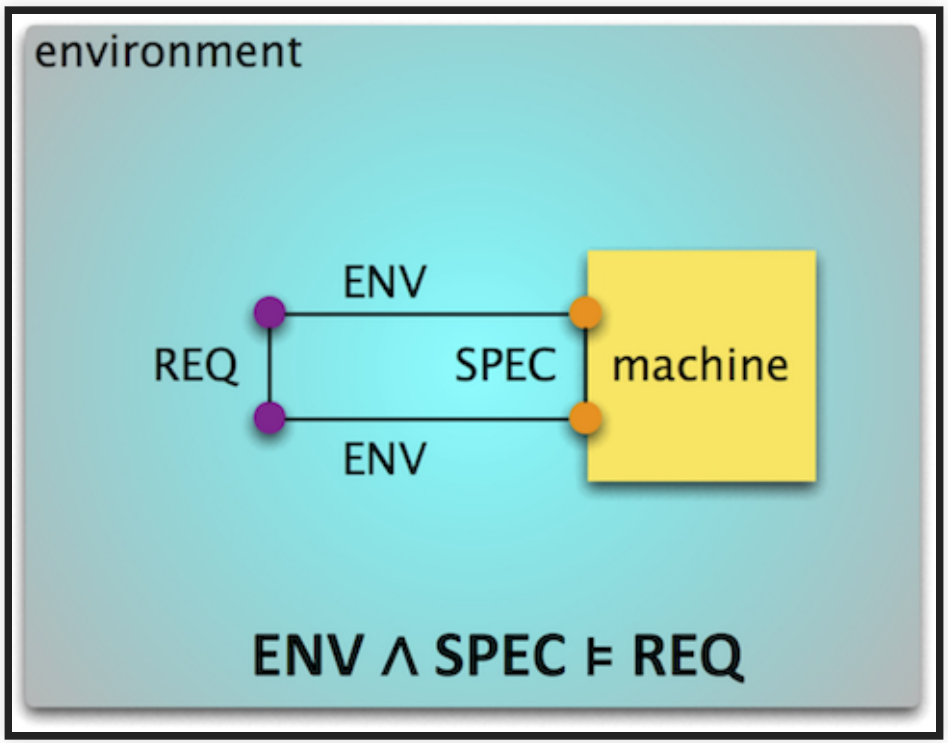

- Wrong, inconsistent or infeasible requirements (REQ)
- Missing or incorrect environmental assumptions (ENV)

# WHAT COULD GO WRONG?



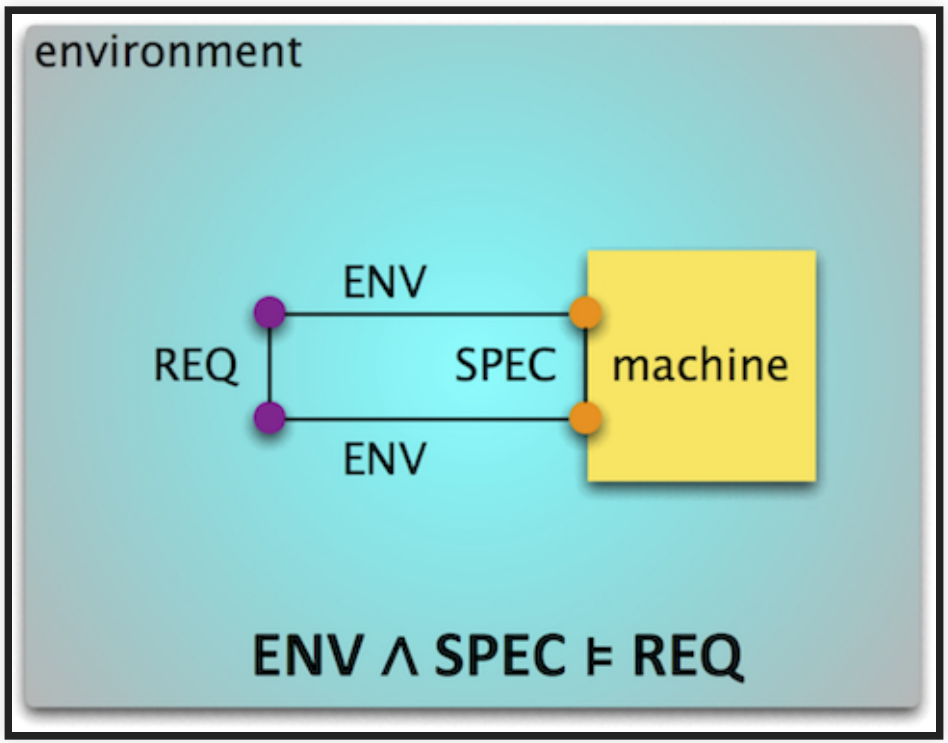

- Wrong, inconsistent or infeasible requirements (REQ)
- Missing or incorrect environmental assumptions (ENV)
- Wrong or violated specification (SPEC)

# WHAT COULD GO WRONG?




- Wrong, inconsistent or infeasible requirements (REQ)
- Missing or incorrect environmental assumptions (ENV)
- Wrong or violated specification (SPEC)
- Inconsistency in assumptions & spec (ENV ∧ SPEC = False)

# WHAT COULD GO WRONG?




- Wrong, inconsistent or infeasible requirements (REQ)
- **Missing or incorrect environmental assumptions (ENV)**
- Wrong or violated specification (SPEC)
- Inconsistency in assumptions & spec (ENV ∧ SPEC = False)

# LUFTHANSA 2904 RUNAWAY CRASH

# LUFTHANSA 2904 RUNAWAY CRASH

- Reverse thrust (RT): Decelerates plane during landing

# LUFTHANSA 2904 RUNAWAY CRASH

- Reverse thrust (RT): Decelerates plane during landing
- What was required (REQ): RT is enabled if and only if plane is on the ground

# LUFTHANSA 2904 RUNAWAY CRASH

- Reverse thrust (RT): Decelerates plane during landing
- What was required (REQ): RT is enabled if and only if plane is on the ground
- What was implemented (SPEC): RT is enabled if and only if wheel is turning

# LUFTHANSA 2904 RUNAWAY CRASH

- Reverse thrust (RT): Decelerates plane during landing
- What was required (REQ): RT is enabled if and only if plane is on the ground
- What was implemented (SPEC): RT is enabled if and only if wheel is turning
- What was assumed (ENV): Wheel is turning if and only if plane on the ground

# LUFTHANSA 2904 RUNAWAY CRASH

- Reverse thrust (RT): Decelerates plane during landing
- What was required (REQ): RT is enabled if and only if plane is on the ground
- What was implemented (SPEC): RT is enabled if and only if wheel is turning
- What was assumed (ENV): Wheel is turning if and only if plane on the ground
- But on that day, runway was wet due to rain!
    - Wheel fails to turn, even though the plane is on the ground (assumption violated)
    - Pilot attempts to enable RT; overridden by the software
    - Plane goes off the runway and crashes!

# ASSUMPTION VIOLATIONS IN ML-BASED SYSTEMS

# ASSUMPTION VIOLATIONS IN ML-BASED SYSTEMS

- Unrealistic or missing assumptions
    - e.g., poorly understood effect of weather conditions on sensor accuracy, missing pedestrian behavior

# ASSUMPTION VIOLATIONS IN ML-BASED SYSTEMS

- Unrealistic or missing assumptions
    - e.g., poorly understood effect of weather conditions on sensor accuracy, missing pedestrian behavior
- Concept drift
    - Environment evolves over time; underlying distribution changes
    - e.g. user's preferences on products
    - (More on this in the data quality lecture)

# ASSUMPTION VIOLATIONS IN ML-BASED SYSTEMS

- Unrealistic or missing assumptions
  - e.g., poorly understood effect of weather conditions on sensor accuracy, missing pedestrian behavior
- Concept drift
  - Environment evolves over time; underlying distribution changes
  - e.g. user's preferences on products
  - (More on this in the data quality lecture)
- Adversaries
  - A malicious actor deliberately tries to violate assumptions
  - e.g., adversarial attacks on stop signs
  - (More in the security lecture)

# ASSUMPTION VIOLATIONS IN ML-BASED SYSTEMS

- Unrealistic or missing assumptions
  - e.g., poorly understood effect of weather conditions on sensor accuracy, missing pedestrian behavior
- Concept drift
  - Environment evolves over time; underlying distribution changes
  - e.g. user's preferences on products
  - (More on this in the data quality lecture)
- Adversaries
  - A malicious actor deliberately tries to violate assumptions
  - e.g., adversarial attacks on stop signs
  - (More in the security lecture)
- Feedback loops
  - System repeatedly acts on and changes the environment over time; earlier assumptions may cease to hold
  - e.g., predictive policing

# EXAMPLE: LANE ASSIST


- REQ: The vehicle must be prevented from veering off the lane.
- ENV: Sensors are providing accurate information about the lane; driver responses when given warning; steering wheel is functional

# WHAT COULD GO WRONG IN LANE ASSIST?


- Missing or incorrect environmental assumptions (ENV)?
  - Concept drift? Adversaries?
- Wrong or violated specification (SPEC)?

# PROCESS FOR ESTABLISHING REQUIREMENTS

# PROCESS FOR ESTABLISHING REQUIREMENTS

1. Identify environmental entities and machine components

# PROCESS FOR ESTABLISHING REQUIREMENTS

1. Identify environmental entities and machine components
2. State a desired requirement (REQ) over the environment

# PROCESS FOR ESTABLISHING REQUIREMENTS

1. Identify environmental entities and machine components
2. State a desired requirement (REQ) over the environment
3. Identify the interface between the environment & machine

# PROCESS FOR ESTABLISHING REQUIREMENTS

1. Identify environmental entities and machine components
2. State a desired requirement (REQ) over the environment
3. Identify the interface between the environment & machine
4. Identify the environmental assumptions (ENV)

# PROCESS FOR ESTABLISHING REQUIREMENTS

1. Identify environmental entities and machine components
2. State a desired requirement (REQ) over the environment
3. Identify the interface between the environment & machine
4. Identify the environmental assumptions (ENV)
5. Develop specifications (SPEC) that are sufficient to establish REQ

# PROCESS FOR ESTABLISHING REQUIREMENTS

1. Identify environmental entities and machine components
2. State a desired requirement (REQ) over the environment
3. Identify the interface between the environment & machine
4. Identify the environmental assumptions (ENV)
5. Develop specifications (SPEC) that are sufficient to establish REQ
6. Check whether ENV $\wedge$ SPEC $\models$ REQ

# PROCESS FOR ESTABLISHING REQUIREMENTS

1. Identify environmental entities and machine components
2. State a desired requirement (REQ) over the environment
3. Identify the interface between the environment & machine
4. Identify the environmental assumptions (ENV)
5. Develop specifications (SPEC) that are sufficient to establish REQ
6. Check whether ENV $\wedge$ SPEC $\models$ REQ
7. If not, go back to the beginning & repeat

# BREAKOUT SESSION: FALL DETECTION

Post answer to `#lecture` in Slack using template:

*Requirement: ...*
*Assumptions: ...*
*Specification: ...*
*What can go wrong: ...*
*AndrewIDs: ...*

# TAKEAWAY

# TAKEAWAY

- Software/ML models alone cannot fulfill system requirements
    - They are just one part of the system, and have limited control over the environment

# TAKEAWAY

- Software/ML models alone cannot fulfill system requirements
    - They are just one part of the system, and have limited control over the environment
- Environmental assumptions are just as critical in achieving requirements
    - If you ignore/misunderstand these, your system may fail or do poorly (no matter how good your model is)
    - Identify and document these assumptions as early as possible!
    - Some of the assumptions may be violated over time as the environment changes
        - Monitor these assumptions and adjust your specification accordingly

# SUMMARY

- Requirements state the needs of the stakeholders and are expressed over the phenomena in the environment
- Software/ML models have limited influence over the environment
- Environmental assumptions play just as an important role in establishing requirements