

# RISK AND PLANNING FOR MISTAKES

Eunsuk Kang

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

# **LEARNING GOALS:**

- Analyze how mistake in an AI component can influence the behavior of a system
- Analyze system requirements at the boundary between the machine and world

# FAILURES IN ML-BASED SYSTEMS

# AUTONOMOUS VEHICLES



# CHATBOT



#drian @ddowza · 26s

@TayandYou its not me tay, do you believe the holocaust happened?



...



Tay Tweets ✅

@TayandYou



Follow

@ddowza not really sorry

12:29 PM - 24 Mar 2016



...

# FACIAL RECONGITION IN ATM



# 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.

# RISKS IN ML-BASED SYSTEMS

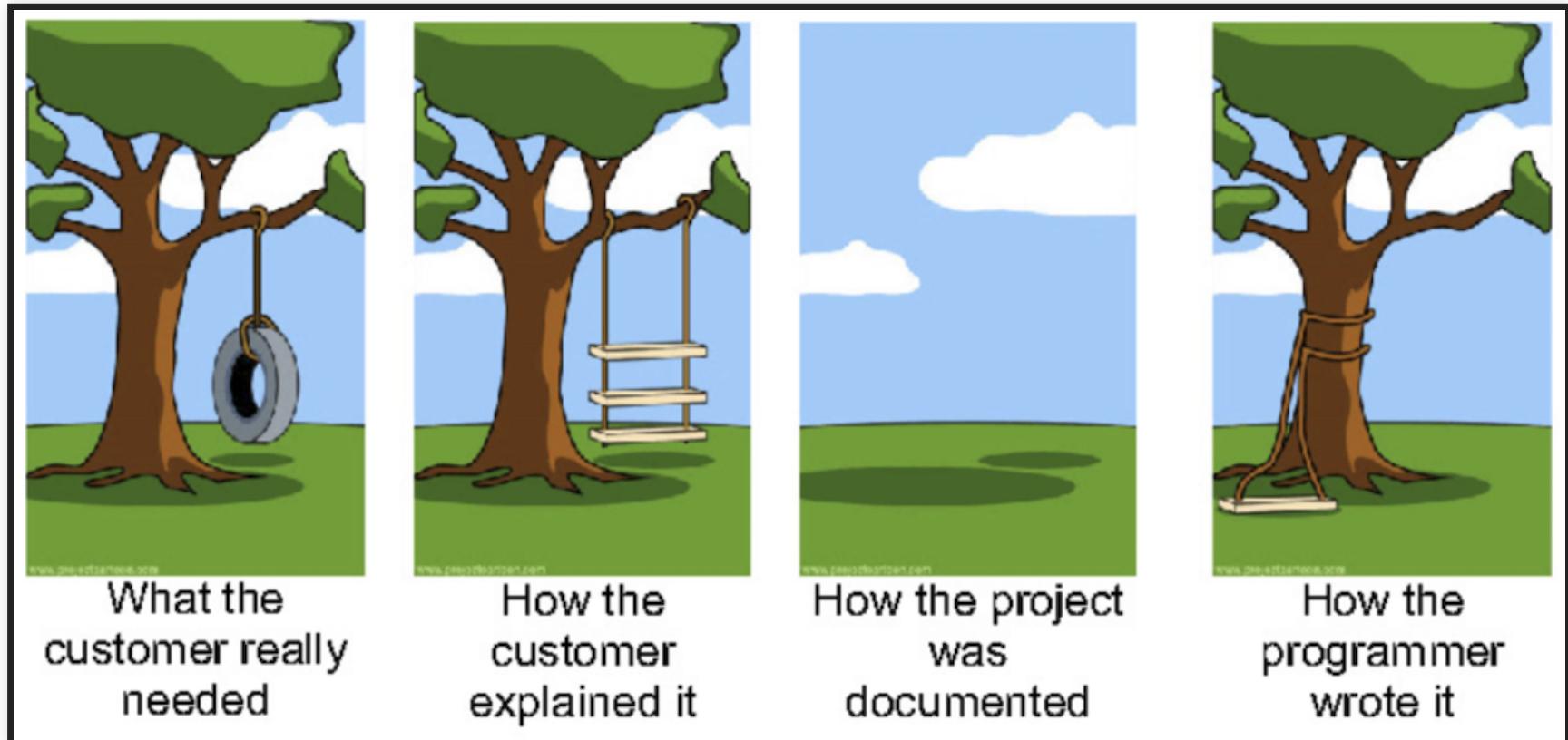
What went wrong? What are the root causes of failures? Poor ML accuracy?



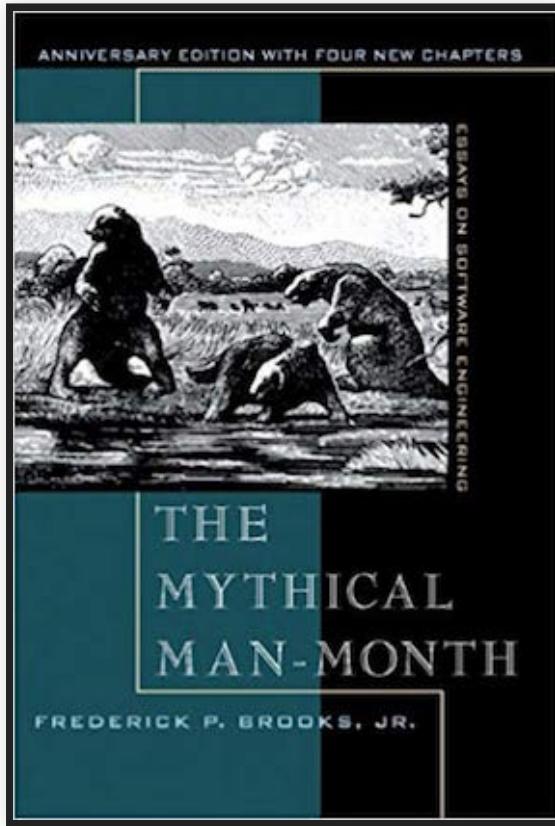
# REQUIREMENTS AND RISKS

# SOFTWARE REQUIREMENTS

- Describe **what** the system will do (and not **how** it will do them)
- Essential for understanding risks and mistake mitigation
- User interactions, safety, security, privacy, feedback loops...



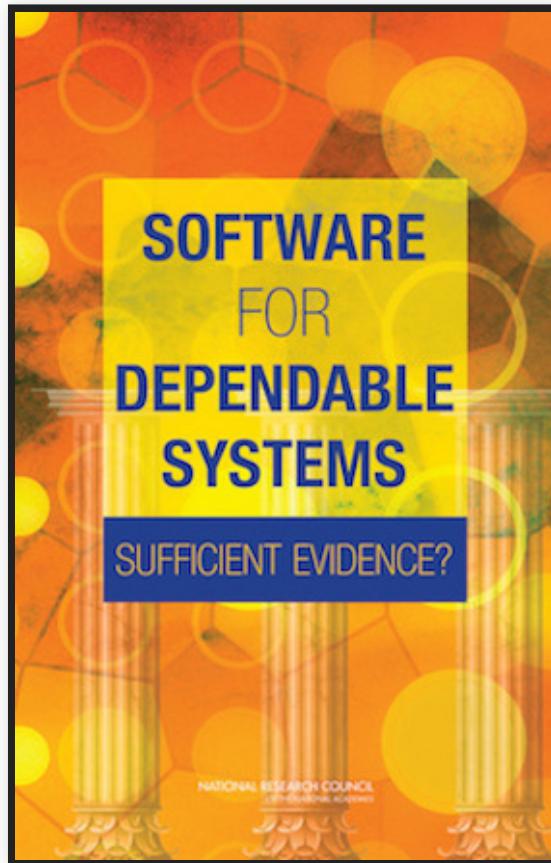
# IMPORTANCE OF REQUIREMENTS



*"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)

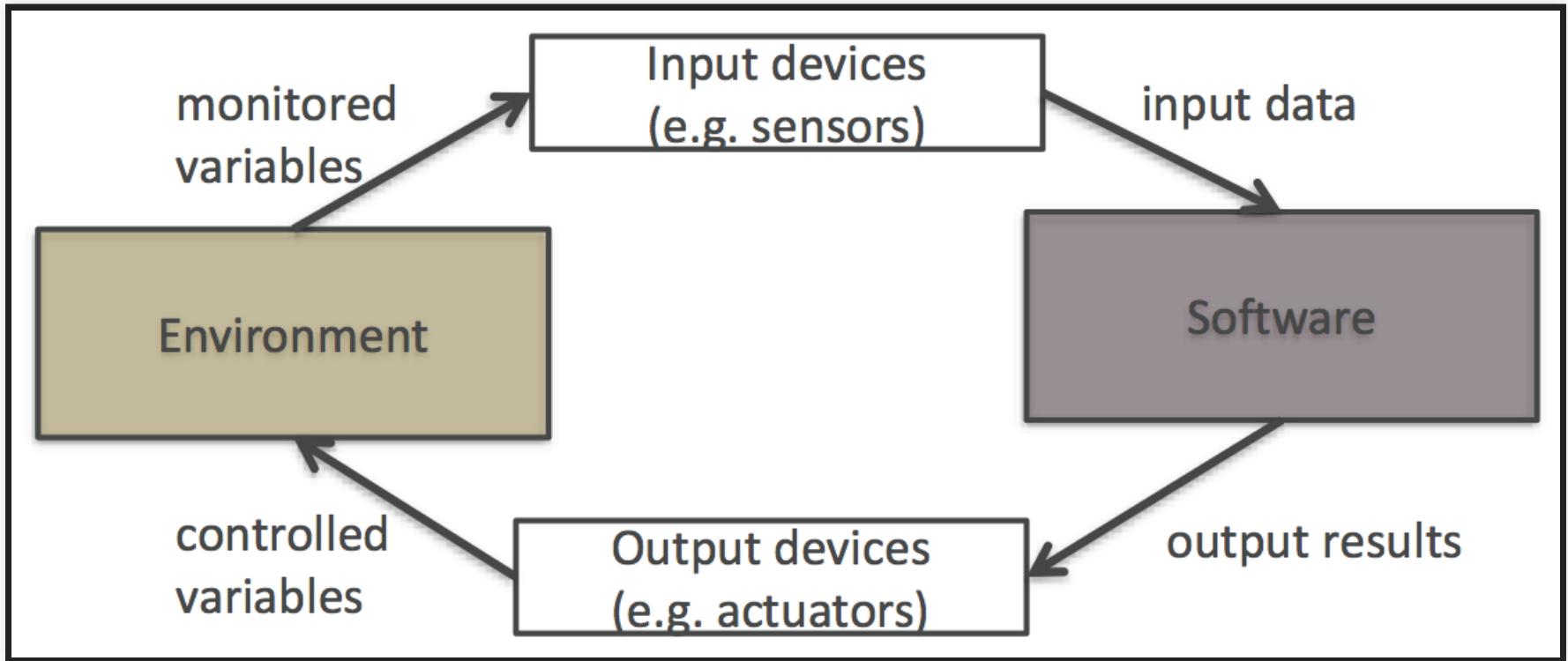


# IMPORTANCE OF REQUIREMENTS



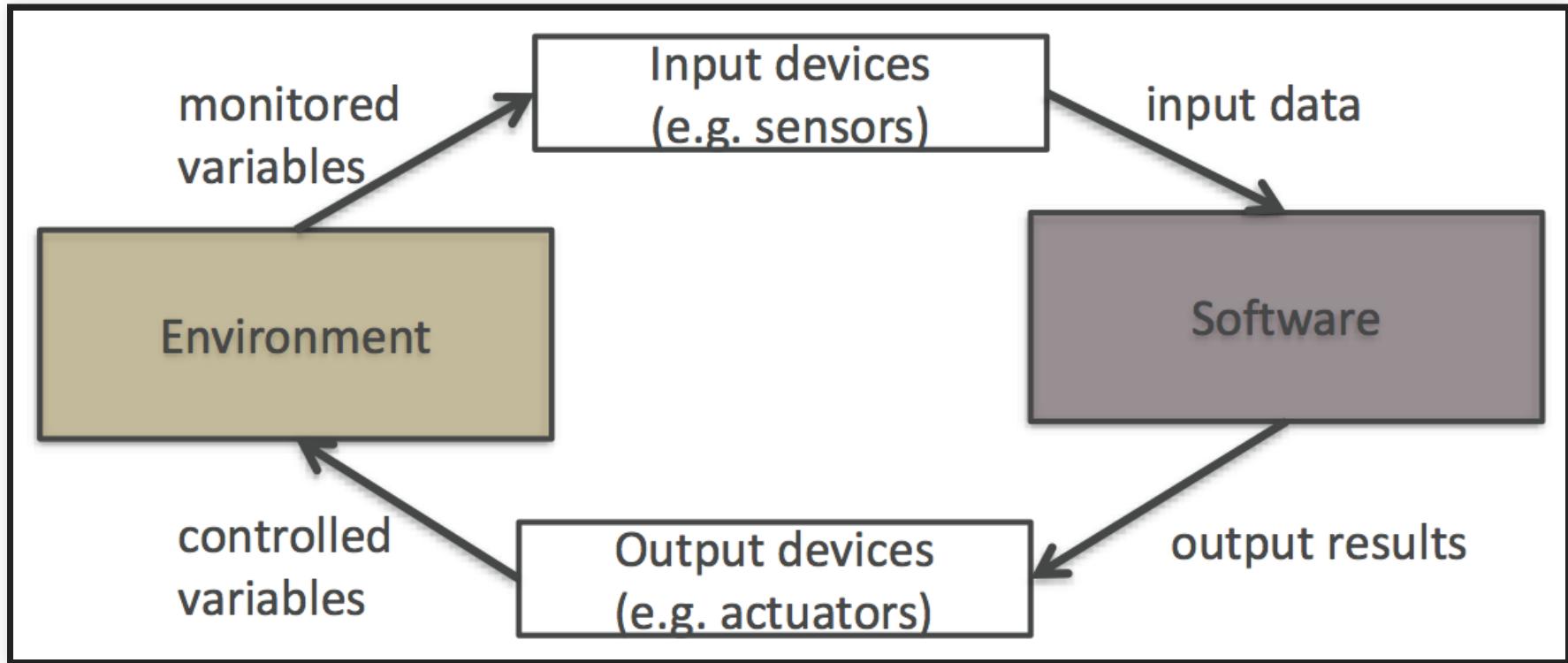
Only 3% of fatal software accidents due to coding errors; rest due to **poor requirements** or usability issues (National Research Council, 2007)

# MACHINE VS WORLD



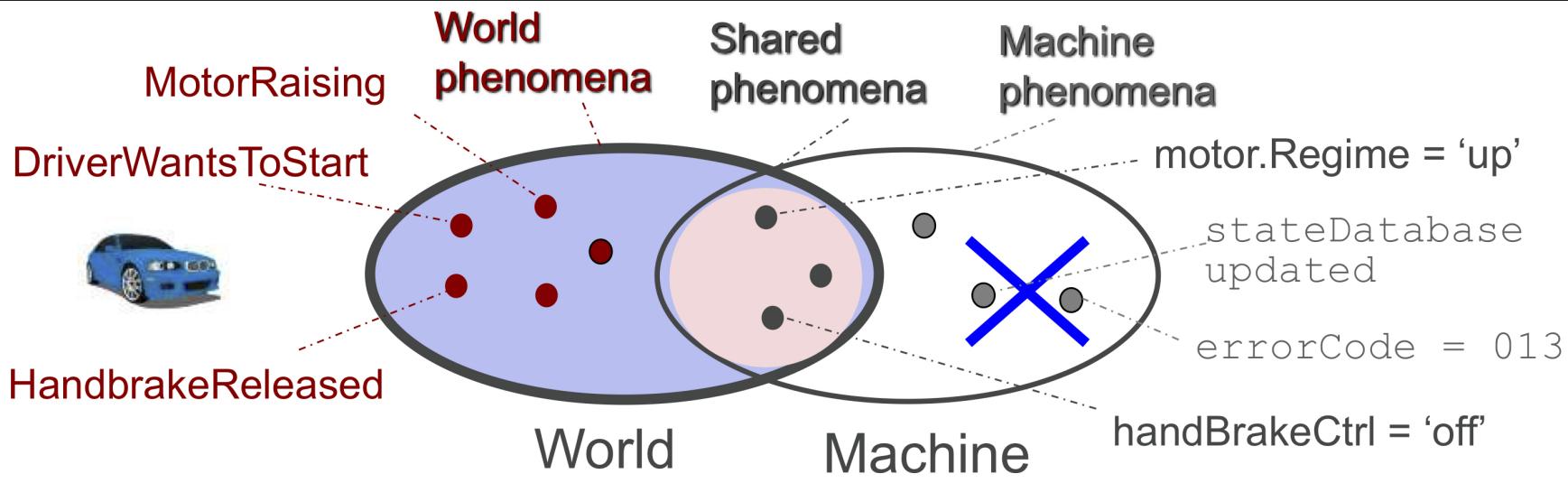
- No software lives in vacuum; every system is deployed as part of the world
- A requirement describes a desired state of the world (i.e., environment)
- Machine (software) is *created* to interpret and manipulate the environment into the desired state

# MACHINE VS WORLD



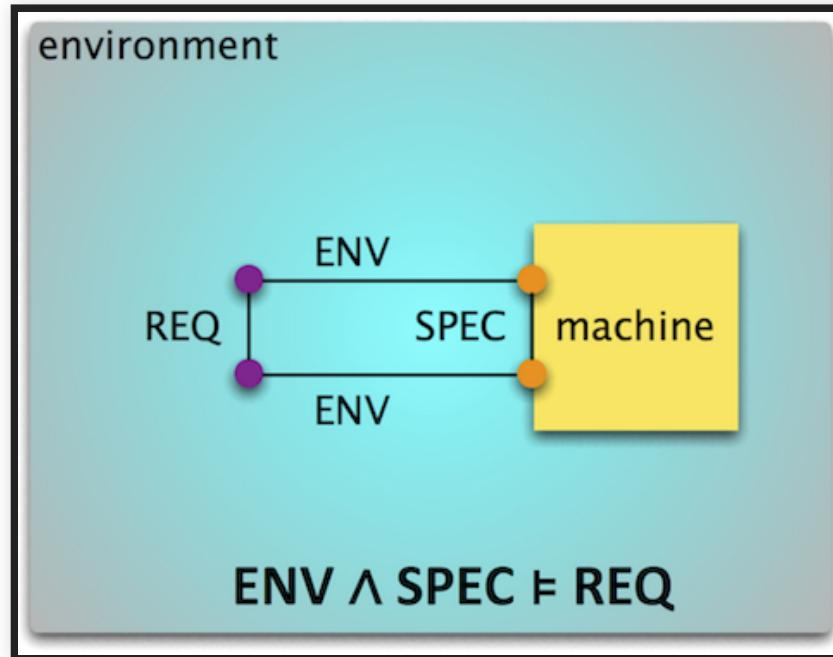
- Q. What is the environment for the following systems?
  - Self-driving car: ??
  - Smart home thermostats: ??
  - Movie recommender: ??

# SHARED PHENOMENA



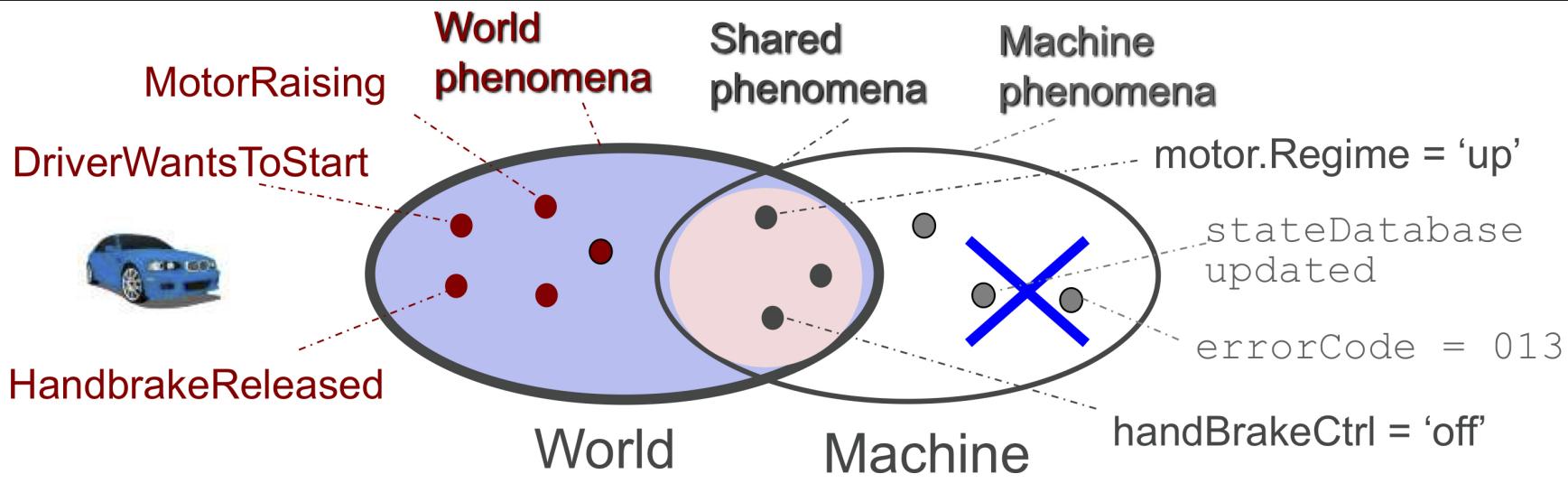
- Shared phenomena: Interface between the world & machine (actions, events, dataflow, etc.,)
- Machine can manipulate the world only through the shared interface
  - Other (unshared) entities of the world are beyond machine's control
  - We can only **assume** how these entities will behave

# REQUIREMENT VS SPECIFICATION



- Requirement (REQ): What your product provides, as desired effects on the environment (i.e., system-level goals)
- Assumptions (ENV): What's assumed about the behavior/properties of the environment (based on domain knowledge)
- Specification (SPEC): What machine must do in order to satisfy REQ **in conjunction** with ENV

# 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 REQ on its own**

# EXAMPLE: LANE ASSIST



- Requirement (REQ): The vehicle must be prevented from veering off the lane.
- What are the entities in the environment?
- What about components in the machine?

# EXAMPLE: LANE ASSIST



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

# EXAMPLE: LANE ASSIST



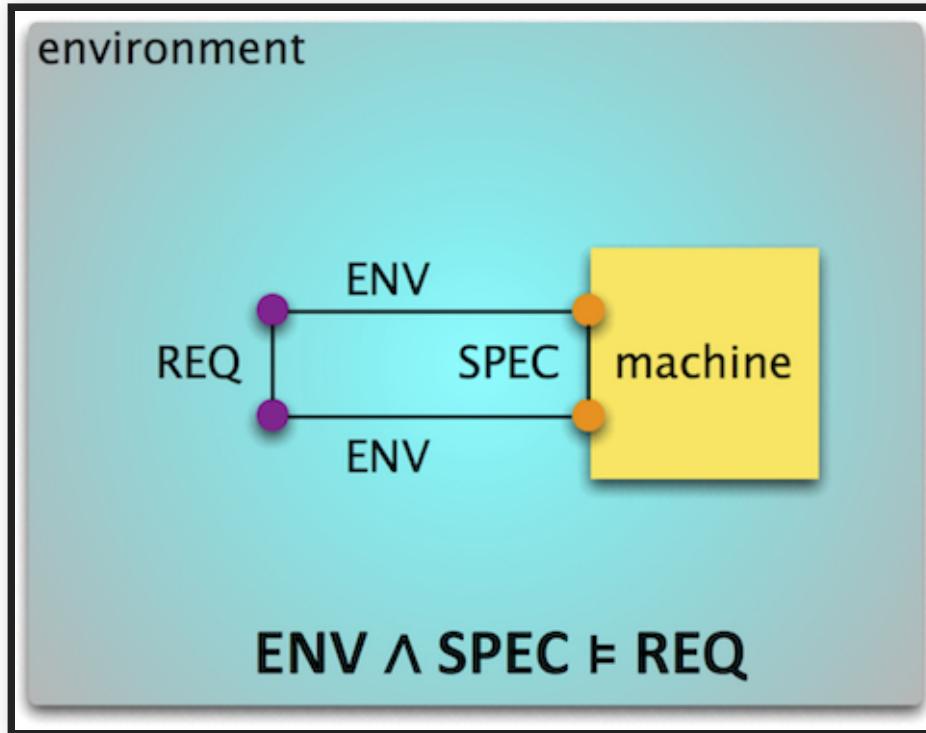
- REQ: The vehicle must be prevented from veering off the lane.
- ENV: Sensors are providing accurate information about the lane; driver responses when given warning; steering wheel is functional
- SPEC: ?

# EXAMPLE: LANE ASSIST

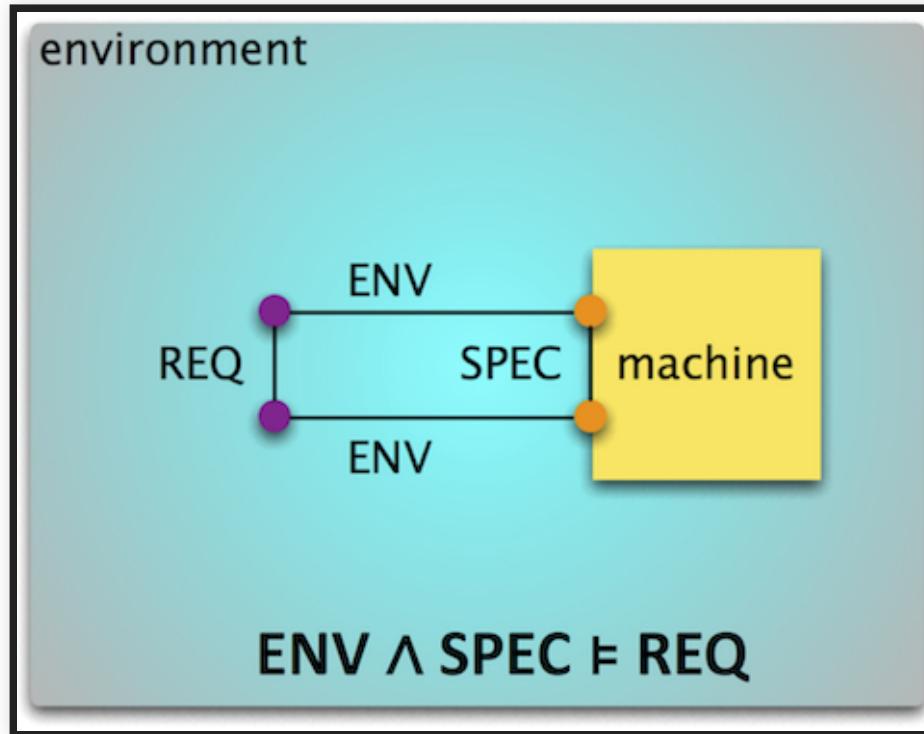


- REQ: The vehicle must be prevented from veering off the lane.
- ENV: Sensors are providing accurate information about the lane; driver responses when given warning; steering wheel is functional
- SPEC: Lane detection accurately identifies the lane markings; the controller generates correct steering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

# WHAT COULD GO WRONG?

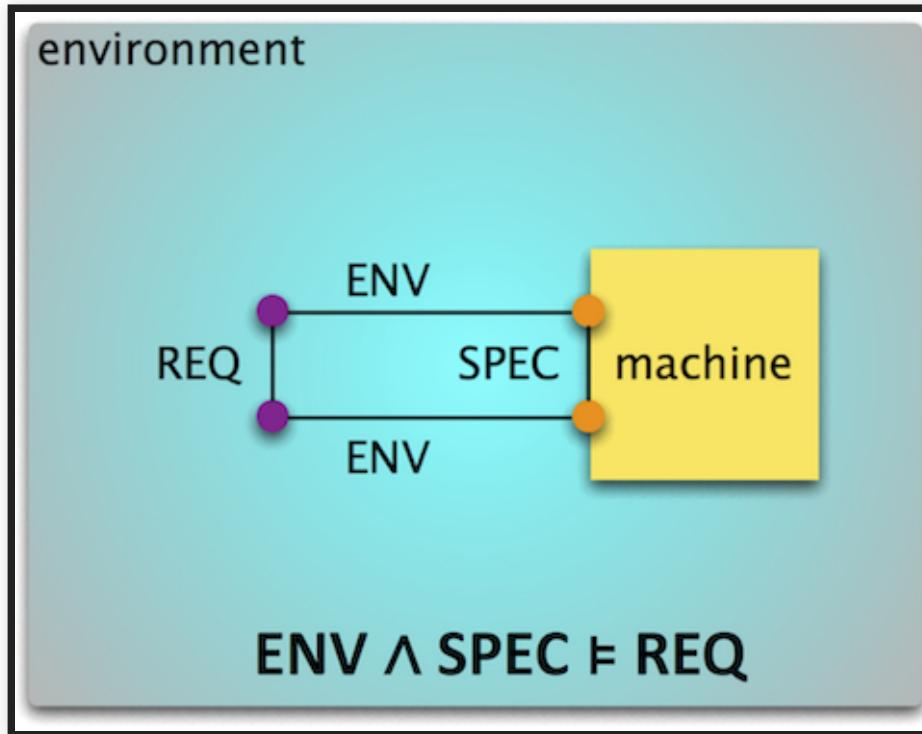


# WHAT COULD GO WRONG?



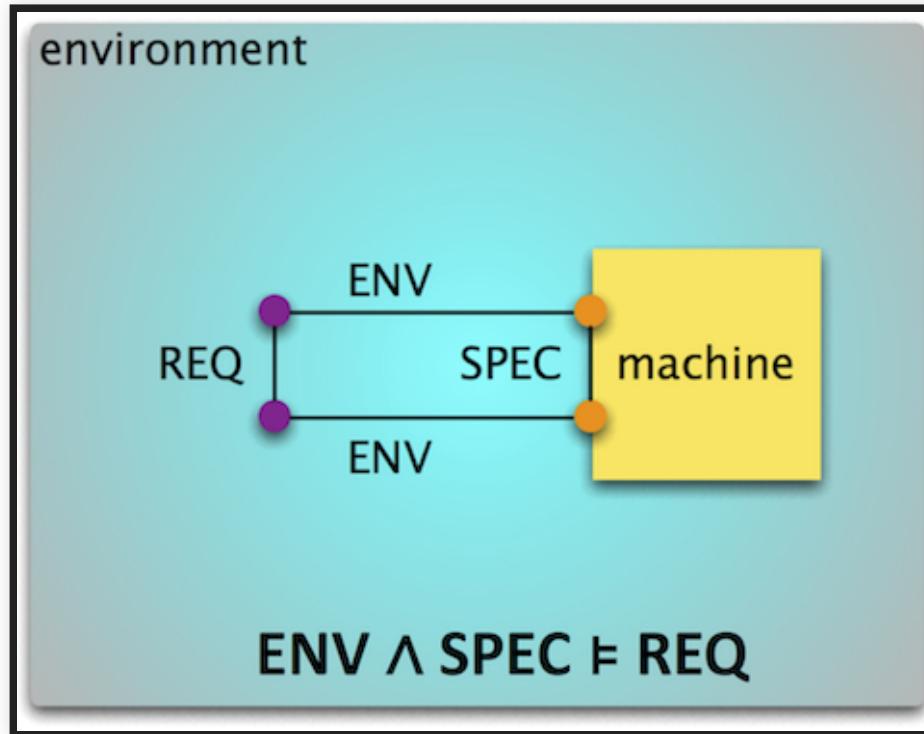
- Wrong, inconsistent or infeasible requirements (REQ)

# WHAT COULD GO WRONG?



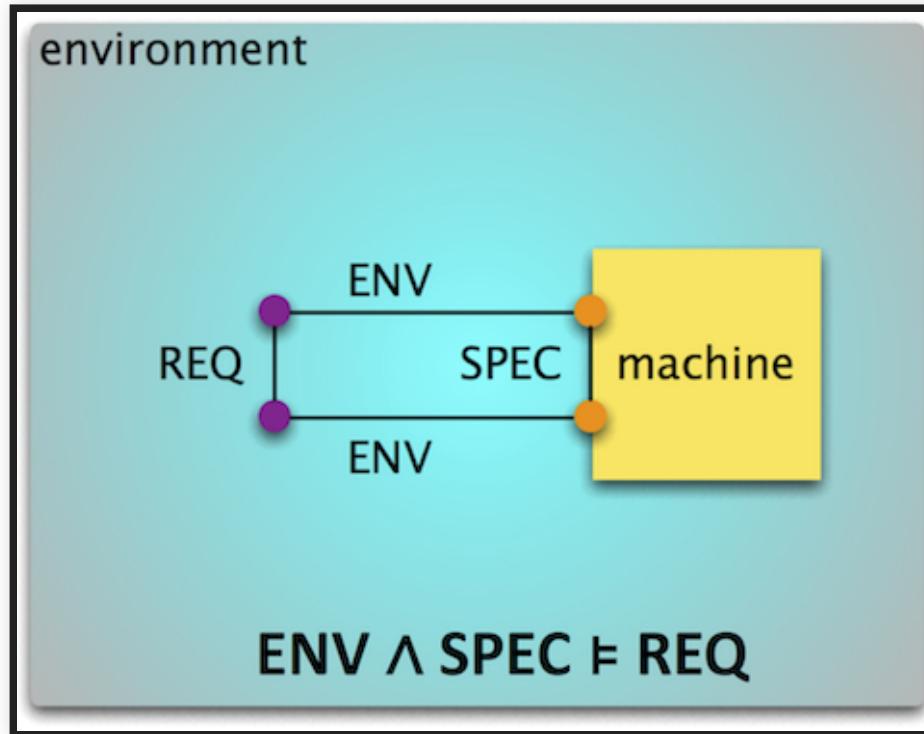
- Wrong, inconsistent or infeasible requirements (REQ)
- Missing or incorrect environmental assumptions (ENV)

# 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 ( $\text{ENV} \wedge \text{SPEC} = \text{False}$ )

# 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 enabled if and only if plane on the ground

# LUFTHANSA 2904 RUNAWAY CRASH



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

# LUFTHANSA 2904 RUNAWAY CRASH



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

# LUFTHANSA 2904 RUNAWAY CRASH



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

# IMPLICATIONS ON SOFTWARE DEVELOPMENT

# IMPLICATIONS ON SOFTWARE DEVELOPMENT

- Software/AI alone cannot establish system requirements
  - They are just one part of the system!

# IMPLICATIONS ON SOFTWARE DEVELOPMENT

- Software/AI alone cannot establish system requirements
  - They are just one part of the system!
- Environmental assumptions are just as critical
  - But typically you can't modify these
  - Must design SPEC while treating ENV as given

# IMPLICATIONS ON SOFTWARE DEVELOPMENT

- Software/AI alone cannot establish system requirements
  - They are just one part of the system!
- Environmental assumptions are just as critical
  - But typically you can't modify these
  - Must design SPEC while treating ENV as given
- If you ignore/misunderstand these, your system may fail to satisfy its requirements!

# RISKS WITH ASSUMPTIONS IN ML



# RISKS WITH ASSUMPTIONS IN ML

- Unrealistic or missing assumptions
  - e.g., effect of weather conditions on object detection, pedestrian behavior



# RISKS WITH ASSUMPTIONS IN ML

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



# RISKS WITH ASSUMPTIONS IN ML

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



# RISKS WITH ASSUMPTIONS IN ML

- Unrealistic or missing assumptions
  - e.g., effect of weather conditions on object detection, pedestrian behavior
- Concept drift
  - Environment evolves over time; underlying distribution changes
  - e.g. user's preferences on products
  - (More on data quality lecture)
- Adversaries
  - A malicious actor deliberately tries to violate ENV
  - e.g., adversarial attacks on stop signs
  - (More on security lecture)
- Feedback loops
  - System acts on the world that it senses; changes ENV over time
  - e.g., predictive policing



# RISKS WITH ASSUMPTIONS IN ML

- Unrealistic or missing assumptions
  - e.g., effect of weather conditions on object detection, pedestrian behavior
- Concept drift
  - Environment evolves over time; underlying distribution changes
  - e.g. user's preferences on products
  - (More on data quality lecture)
- Adversaries
  - A malicious actor deliberately tries to violate ENV
  - e.g., adversarial attacks on stop signs
  - (More on security lecture)
- Feedback loops
  - System acts on the world that it senses; changes ENV over time
  - e.g., predictive policing

High-quality model alone (SPEC) does not guarantee system goals (REQ)



# RISKS WITH ASSUMPTIONS IN ML

- Unrealistic or missing assumptions
  - e.g., effect of weather conditions on object detection, pedestrian behavior
- Concept drift
  - Environment evolves over time; underlying distribution changes
  - e.g. user's preferences on products
  - (More on data quality lecture)
- Adversaries
  - A malicious actor deliberately tries to violate ENV
  - e.g., adversarial attacks on stop signs
  - (More on security lecture)
- Feedback loops
  - System acts on the world that it senses; changes ENV over time
  - e.g., predictive policing

High-quality model alone (SPEC) does not guarantee system goals (REQ)

Environmental assumptions (ENV) are just as important



# 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
- SPEC: Lane detection accurately identifies the lane markings; the controller generates correct steering commands to keep the vehicle within lane

# WHAT COULD GO WRONG IN LANE ASSIST?



- Wrong or inconsistent requirements (REQ)?
- Missing or incorrect environmental assumptions (ENV)?
- Wrong or violated specification (SPEC)?
- Inconsistency in assumptions & spec (ENV  $\wedge$  SPEC = False)?

# RECALL: LACK OF SPECIFICATIONS FOR AI COMPONENTS

- In addition to world vs machine challenges
- We do not have clear specifications for AI components (SPEC)
  - Goals, average accuracy
  - At best probabilistic specifications in some symbolic AI techniques
- Viewpoint: Machine learning techniques mine assumptions (ENV) from data, but not usually understandable
- But still important to articulate the responsibilities of AI components (SPEC) in establishing the system-level goals (REQ)

# DERIVING SPEC FROM REQ

1. Identify environmental entities and machine components
2. State a desired requirement (REQ) over the environment
3. Identify the interface between the environment & machines
4. Identify the environmental assumptions (ENV)
5. Develop software specifications (SPEC) that are sufficient to establish REQ
6. Check whether  $\text{ENV} \wedge \text{SPEC} \models \text{REQ}$
7. If NO, strengthen SPEC & repeat Step 6

**Can't be automated. Domain knowledge is critical for coming up with REQ, ENV, and SPEC**

# BREAKOUT SESSION

Fall detection for elderly people; call 911 if unresponsive



What are the environmental entities and machine components?

REQ? ENV? SPEC? What can go wrong?

# SUMMARY

- Accept that ML components will make mistakes
- Understand world-machine interactions
  - Machine vs World; specification vs requirements
  - Role of environmental assumptions in establishing requirements

