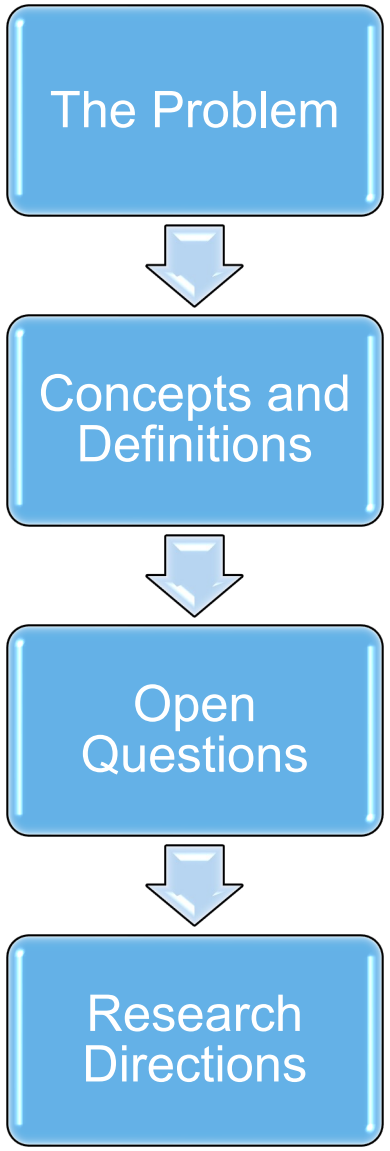
# The Relevance of System Context

## Evaluating Autonomy: What Does “Really Well” Mean?

IROS 2023 Workshop “It Works Really Well!”: Verification in Theory and Practice

**Mauricio Castillo-Effen, Ph.D.**

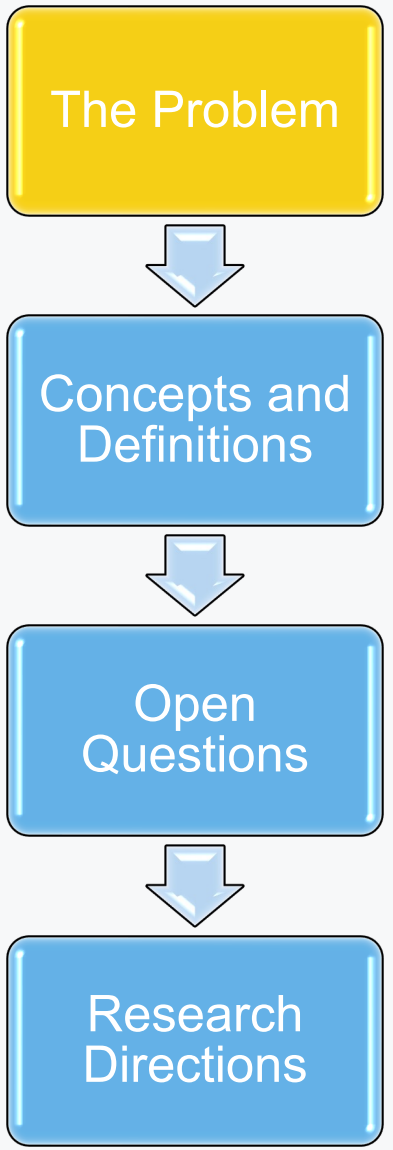
# Overview




**DISCLAIMER**
The views and opinions presented in this presentation are solely the author's, and they do not represent the official policy or position of the Lockheed Martin Corporation or the Lockheed Martin Advanced Technology Laboratories.

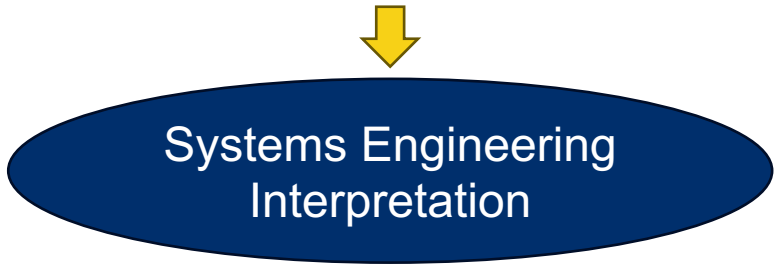
| Requirement | Context 1: Success | Context 2: Failure |
|---|---|---|
| *"The autonomous vehicle must follow all traffic signals"* | Nominal traffic conditions | Emergency situations<br>Construction zone<br>Faulty traffic light |
| *"The autonomous vehicle must halt at all stop signs"* | All stop signs are real | A stop sign shown on a placard |
| *"The cleaning robot must pick all small objects from the floor and throw them away"* | All small objects on the floor are trash | Somebody dropped their keys |
| *"AI must assign more resources to patients who spend more"* | Patients with grave illnesses in a mid to high income bracket | Low income patients |

Context is necessary to define what "Really Well!!" means

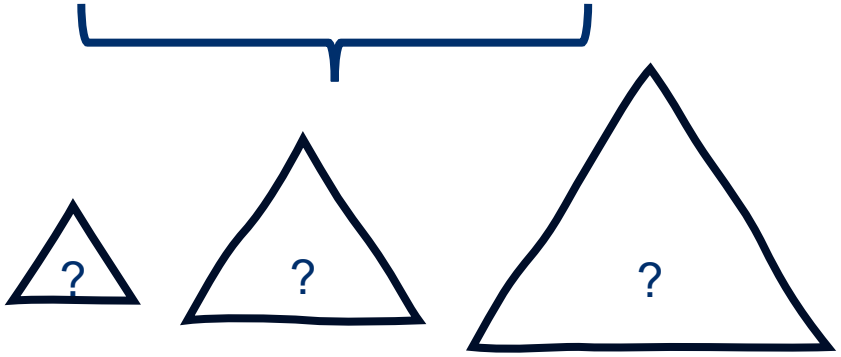


# What happens when context is undefined?

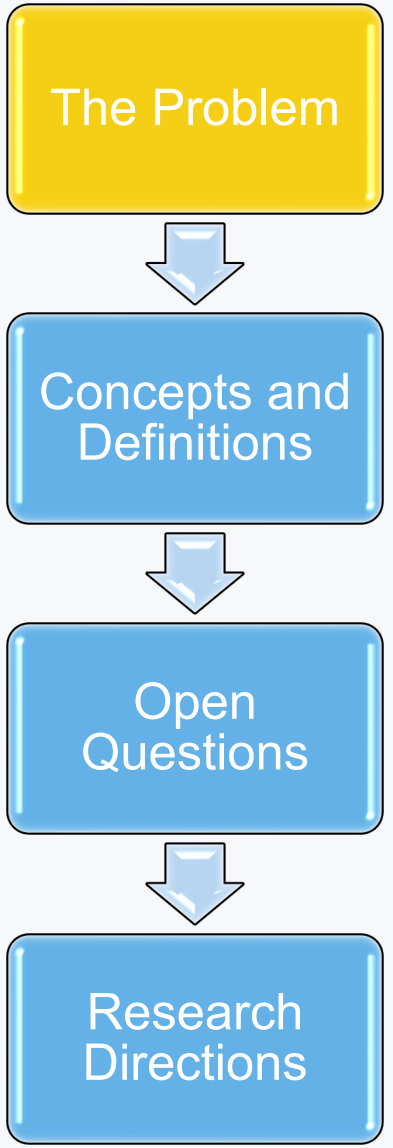
Requirement: *"System must do X"*




*System must do X in all foreseeable situations*



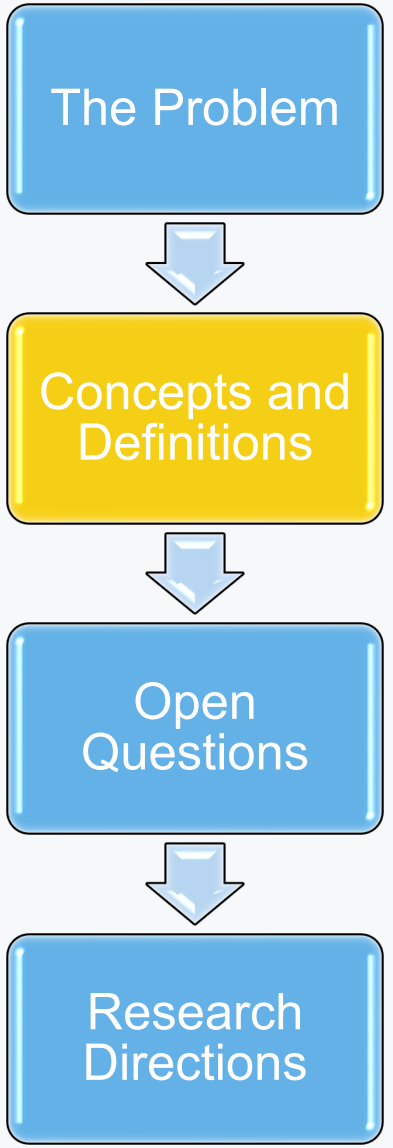

The term "all foreseeable situations" is ambiguously defined making the assurance burden equally unclear!



# Definitions of “Context”

- INCOSE: “The setting in which a system exists, including the stakeholders, system elements, environment, and the relationships among them”
- ISO/IEC/IEEE 15288: “The environment in which a system, product, or service is used or intended to be used”
- “Systems Architecting” (Eberhardt Rechtin): “The broad set of constraints, needs, and goals that shape a system’s requirements, structure, and function”

Existing definitions are too general—not useful for evaluation

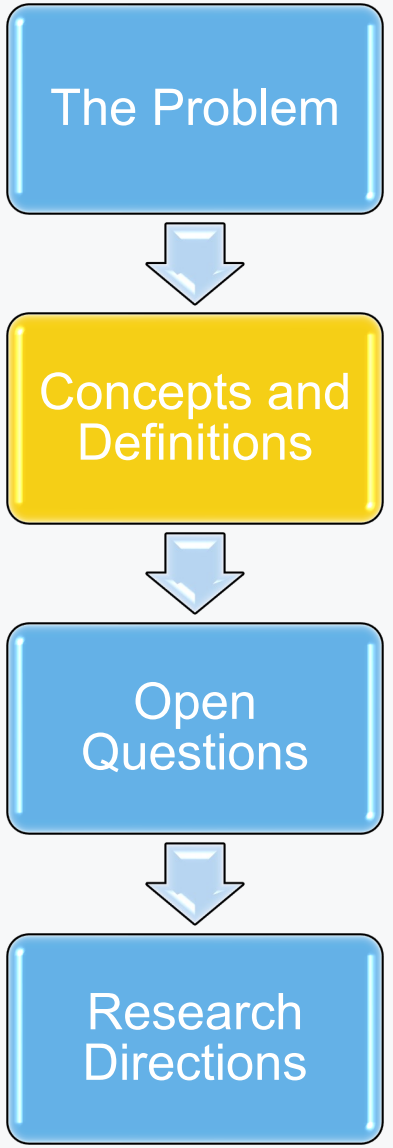


# Operational Design Domains (ODDs)

"Operating conditions under which a given driving automation system or feature thereof is specifically designed to function, including, but not limited to, environmental, geographical, and time-of-day restrictions, and/or the requisite presence or absence of certain traffic or roadway characteristics."

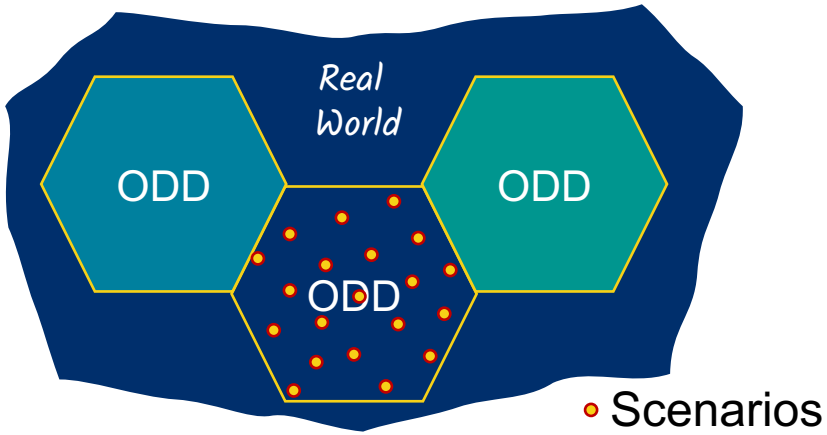
SAE International, J3016 (Apr, 2021): (R) Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles.

ODDs provide a more concrete approach to defining "context"

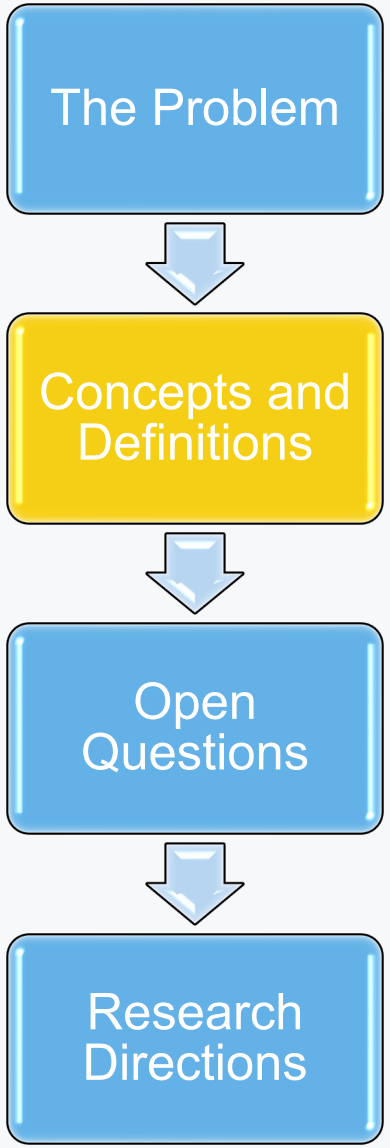


# Related Definitions

- Operational Domain (OD): “real world situations”
- ODD is a model of the OD
- ODD is defined by attributes and their variability
- Scenario:
  - “A description of a driving situation that includes the pertinent actors, environment, objectives and sequences of events” [SOURCE: BSI Flex 1890 v4.0:2022-03, 2.1.73]
  - A unique combination of ODD attributes

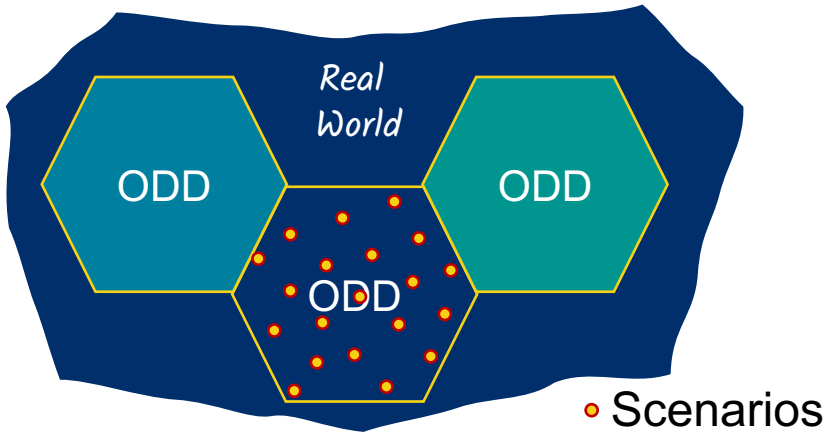


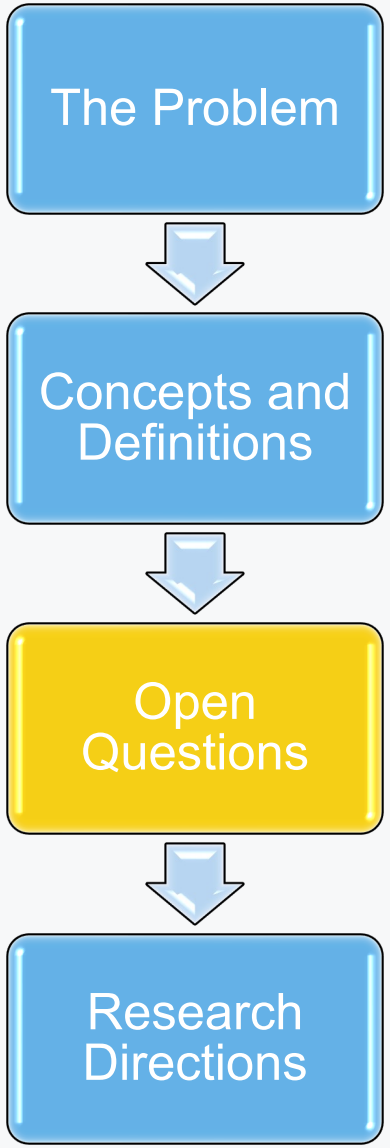

ODDs can be used for scenario-based validation



# 7 Intriguing Questions

1. How do determine how well an ODD covers the OD?
2. How to represent ODDs formally?
   - Examples: OpenScenario, Open-Drive, Scenic, etc.
3. How to define ODD boundaries? How many ODDs?
4. Scenario similarity (“distance”)
5. How many scenarios are sufficient to cover an ODD?
6. What do we do with Δ = OD – ODD?
7. What is the relationship between ODDs and requirements?

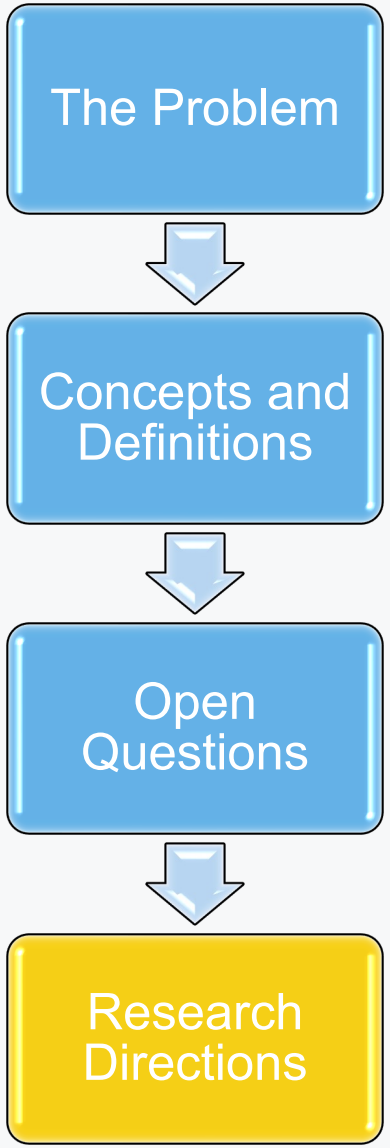


# Research Directions: ODD Description Language (ODD DL)

- Leading scenario description languages are restrictive
  - Need: multi-domain, include adversaries, behavior/intent, etc.
- Use description language to automatically generate “knobs” and “probes” for external simulators (scenario-based sim-based testing)
- Use ODD DL description language to generate scenarios automatically
- Experiments in generating ODD DL from natural language via LLMs
- Experiments in interpretation of ODD DL as probabilistic programming languages
- Complementary with work in requirement formalization and test generation from requirements

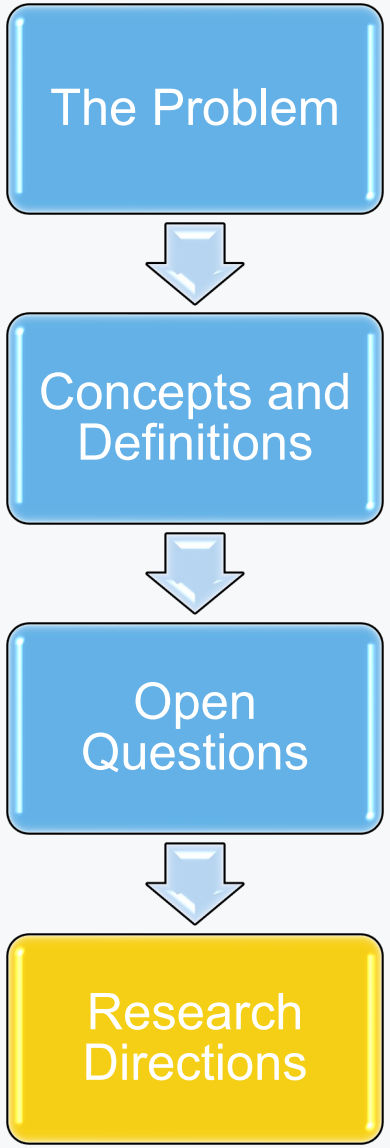
ODD DLs have opened rich and impactful lines of research



# Research Directions: Assurance

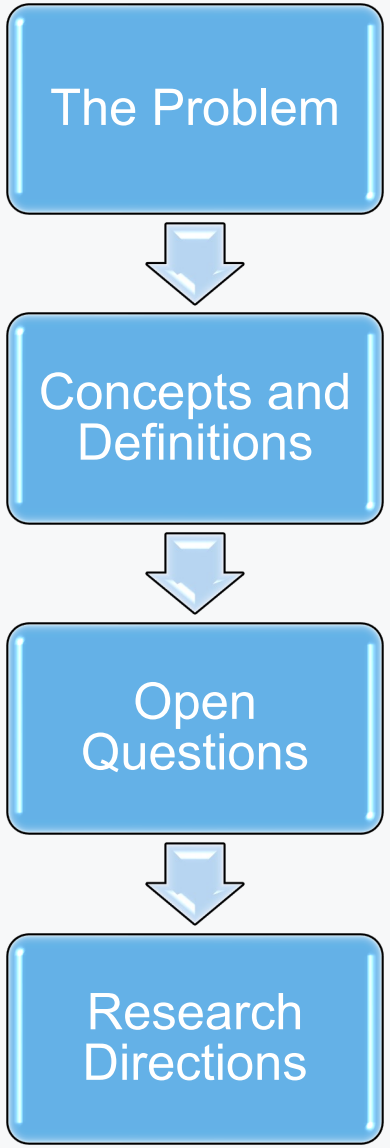
- Assurance case patterns for scenario-based validation
  - Real world coverage: $\bigcup_i ODD_i \sim \text{Real World}$
  - OD Coverage ($ODD_i$ description is representative)
  - Scenario Coverage (Scenario samples cover $ODD_i$)
  - Fidelity (Validation environment produces valid executions)
- Agile assurance
  - It is easier to achieve justified confidence if we restrict ODD's variability or the scope of the real world (# ODDs)
  - Automation for assurance case regeneration when context changes
- Semantic reasoning
  - Can we reuse evidence or assurance case fragments based on context similarity?

Reasoning about context opens the possibility for meaningful integration of assurance and agility!



# Conclusions

- There is important work in relation to context that will help with the deployment of autonomous systems with justified confidence
- We need to leverage the groundwork laid by the autonomous driving community, e.g.: ODDs, scenario-based validation, etc.
- Reasoning about context could serve as the basis for the introduction of agility in assurance
- Context size ∝ Assurance burden




"Really Well" only makes sense if we specify context formally and rigorously

LOCKHEED MARTIN