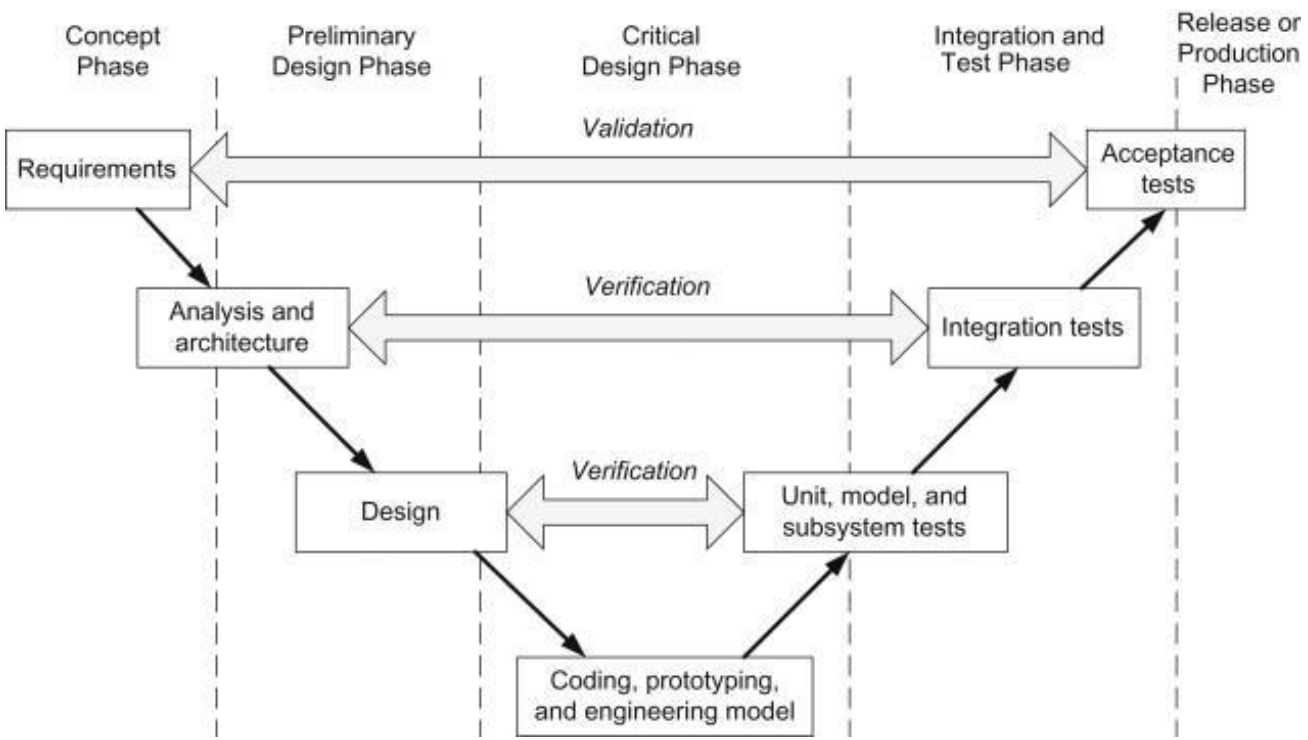
MANCHESTER 1824
The University of Manchester

Royal Academy of Engineering

# What Should I Verify?

Marie Farrell
The University of Manchester

IVOIRE Workshop 2024


Concept Phase
Preliminary Design Phase
Critical Design Phase
Integration and Test Phase
Release or Production Phase
Validation
Requirements
Acceptance tests
Verification
Analysis and architecture
Integration tests
Verification
Design
Unit, model, and subsystem tests
Coding, prototyping, and engineering model


WALLY, WE DON'T HAVE TIME TO GATHER THE PRODUCT REQUIREMENTS AHEAD OF TIME.
I WANT YOU TO START DESIGNING THE PRODUCT ANYWAY. OTHERWISE IT WILL LOOK LIKE WE AREN'T ACCOMPLISHING ANYTHING.
OF ALL MY PROJECTS, I LIKE THE DOOMED ONES BEST.
NEWS
S. Adams
www.unitedmedia.com

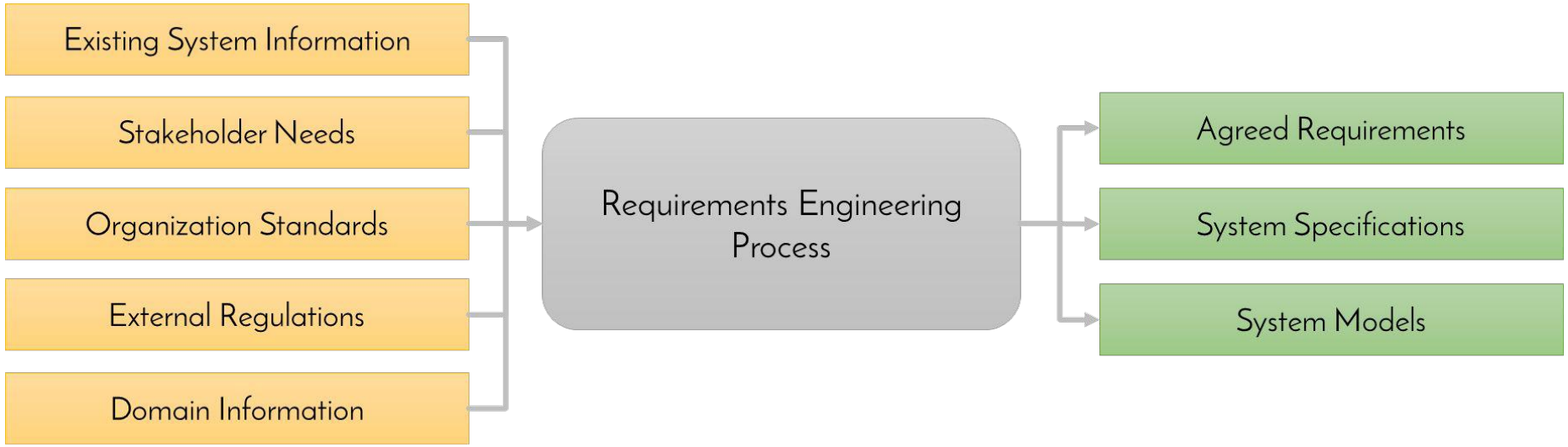
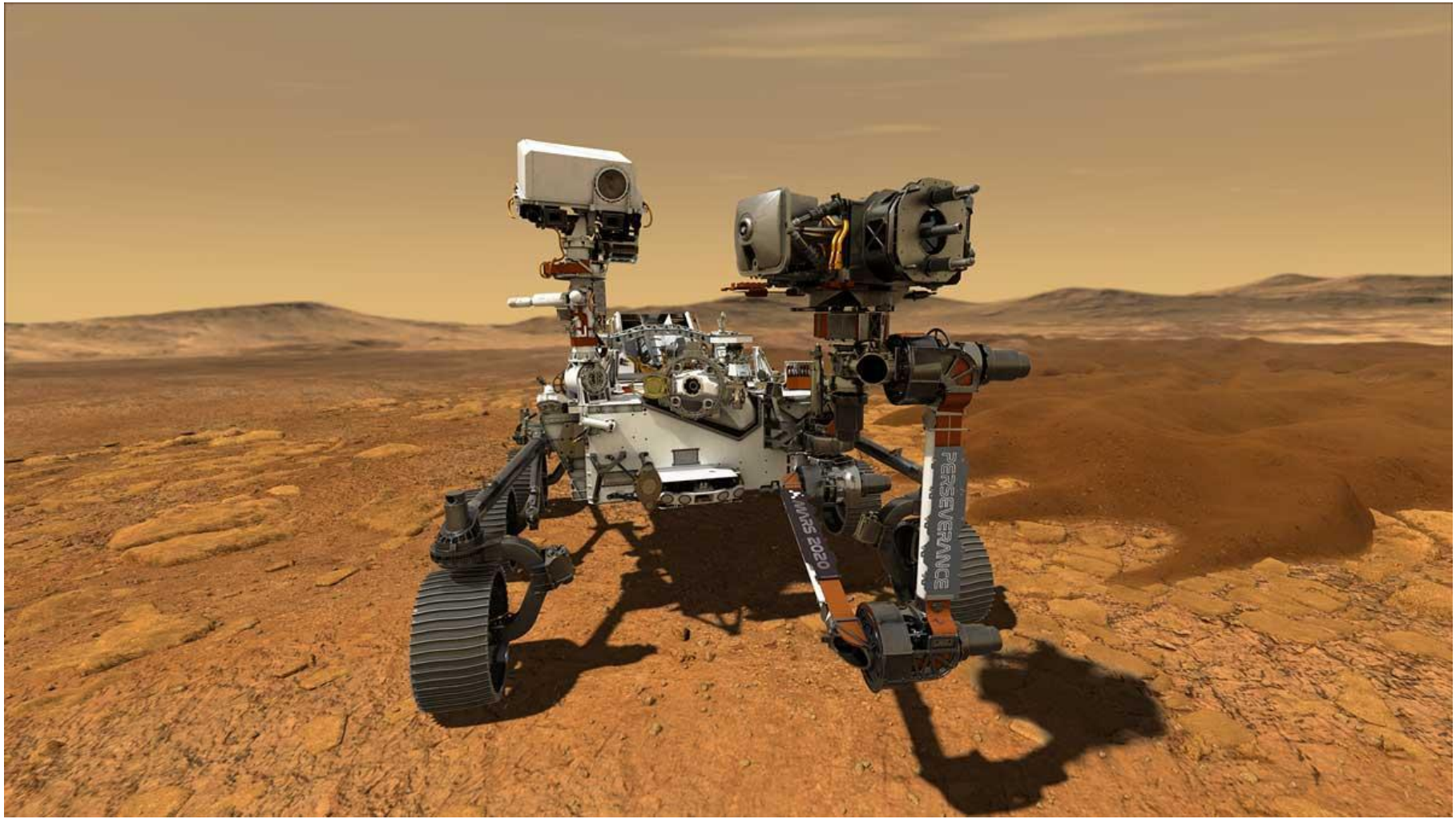
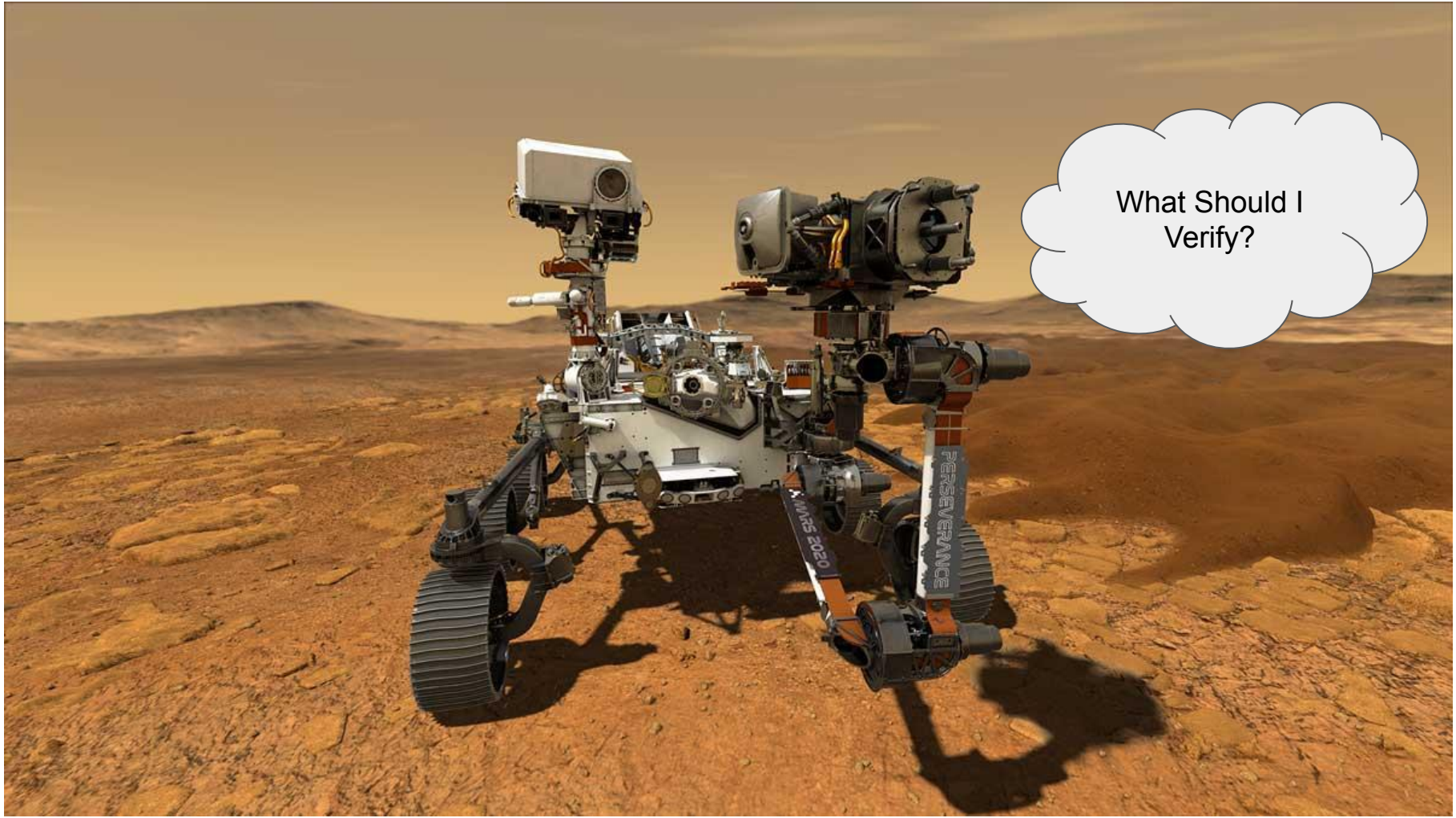



Illustration: The overall requirements engineering process


PERSEVERANCE
MARS 2020


What Should I Verify?
PERSEVERANCE
MARS 2020

## Natural Language

*"After collecting sample, the rover shall place the sample into container."*

# Natural Language

*"After collecting sample, the rover shall place the sample into container."*

✓ Intuitive

- Ambiguous
- Not amenable to formal analysis

## Natural Language

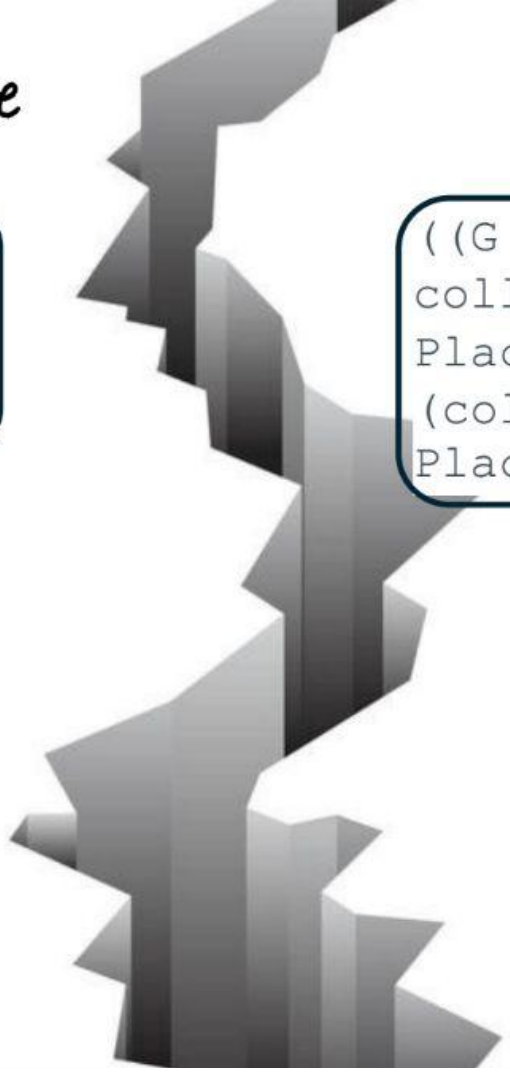
*"After collecting sample, the rover shall place the sample into container."*

✓ Intuitive

- Ambiguous
- Not amenable to formal analysis

## Formal Language

```
((G (((! collectSamples) & (X
collectSamples)) -> (X (F[0,5]
PlaceSamplesInContainers)))) &
(collectSamples -> (F[0,5]
PlaceSamplesInContainers)))
```



## Natural Language

*"After collecting sample, the rover shall place the sample into containers."*

✓ Intuitive

- Ambiguous
- Not amenable to formal analysis

## Formal Language

```
((G (((! collectSamples) & (X
collectSamples)) -> (X (F[0,5]
PlaceSamplesInContainers)))) &
(collectSamples -> (F[0,5]
PlaceSamplesInContainers)))
```

✓ Unambiguous
✓ Amenable to Formal Analysis

- Unintuitive

## Natural Language

"After collecting sample, the rover shall place the sample into container."

## Formal Language

```
((G ((((! collectSamples) & (X
collectSamples)) -> (X (F[0,5]
PlaceSamplesInContainers)))) &
(collectSamples -> (F[0,5]
PlaceSamplesInContainers)))
```

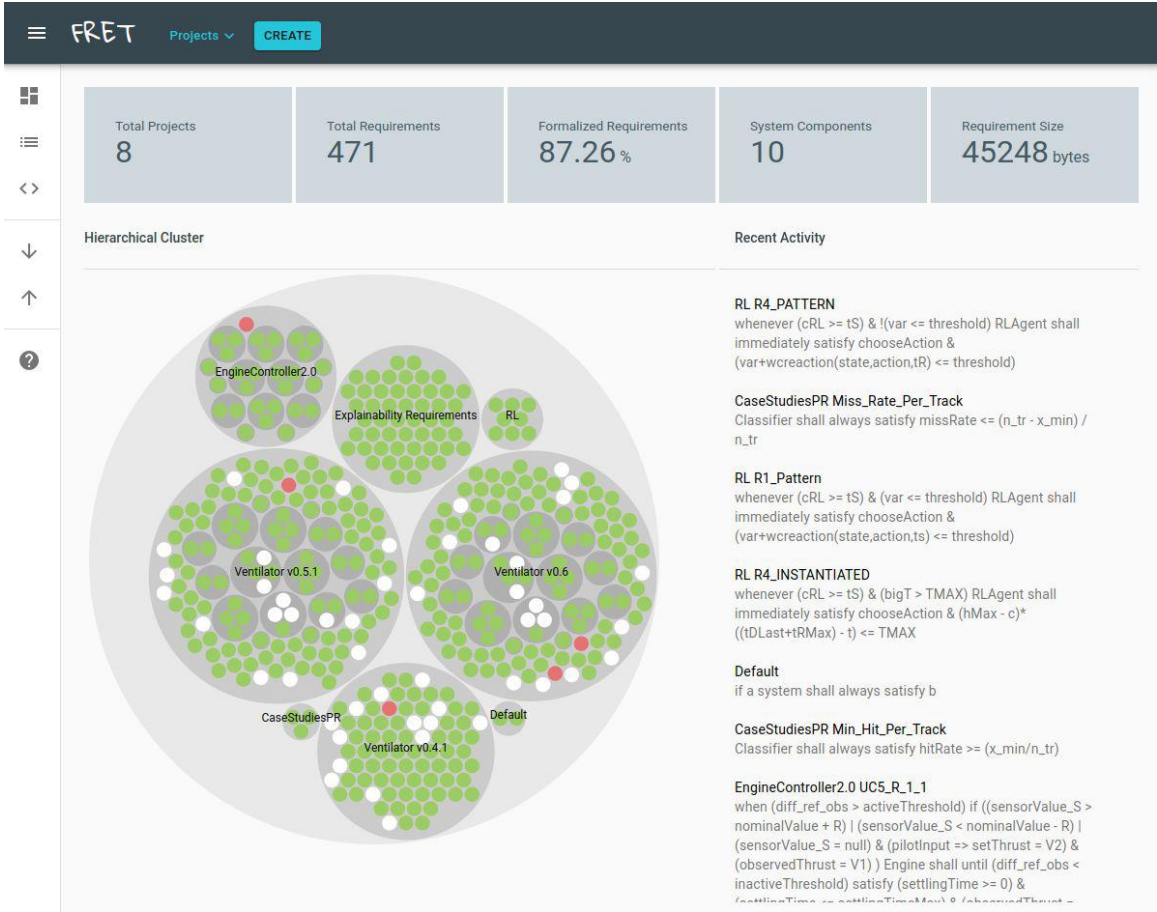

✓ Intuitive

- Ambiguous
- Not amenable to formal analysis

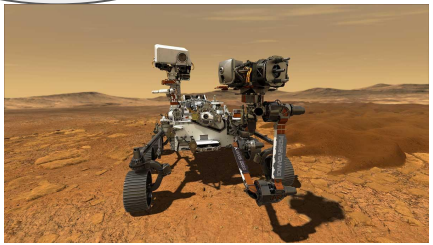
✓ Unambiguous
✓ Amenable to Formal Analysis

- Unintuitive

Let's Speak FRETish!



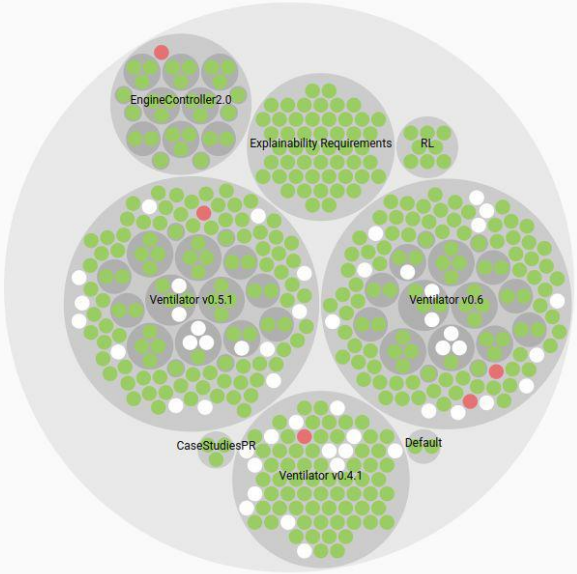

# Let's Speak FRETish!



FRET Projects CREATE

| Total Projects | Total Requirements | Formalized Requirements | System Components | Requirement Size |
|---|---|---|---|---|
| 8 | 471 | 87.26 % | 10 | 45248 bytes |

Hierarchical Cluster




Recent Activity

**RL R4_PATTERN**
whenever (cRL >= tS) & !(var <= threshold) RLAgent shall immediately satisfy chooseAction & (var+wcreaction(state,action,tR) <= threshold)

**CaseStudiesPR Miss_Rate_Per_Track**
Classifier shall always satisfy missRate <= (n_tr - x_min) / n_tr

**RL R1_Pattern**
whenever (cRL >= tS) & (var <= threshold) RLAgent shall immediately satisfy chooseAction & (var+wcreaction(state,action,ts) <= threshold)

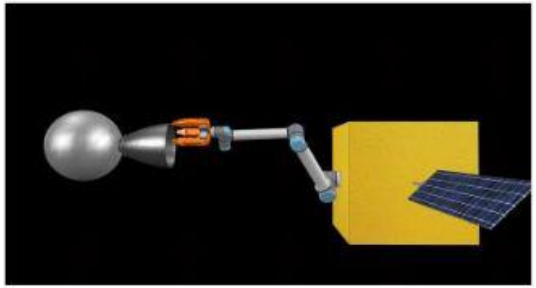
**RL R4_INSTANTIATED**
whenever (cRL >= tS) & (bigT > TMAX) RLAgent shall immediately satisfy chooseAction & (hMax - c)* ((tDLast+tRMax) - t) <= TMAX

**Default**
if a system shall always satisfy b

**CaseStudiesPR Min_Hit_Per_Track**
Classifier shall always satisfy hitRate >= (x_min/n_tr)

**EngineController2.0 UC5_R_1_1**
when (diff_ref_obs > activeThreshold) if ((sensorValue_S > nominalValue + R) | (sensorValue_S < nominalValue - R) | (sensorValue_S = null) & (pilotInput => setThrust = V2) & (observedThrust = V1) ) Engine shall until (diff_ref_obs < inactiveThreshold) satisfy (settlingTime >= 0) &

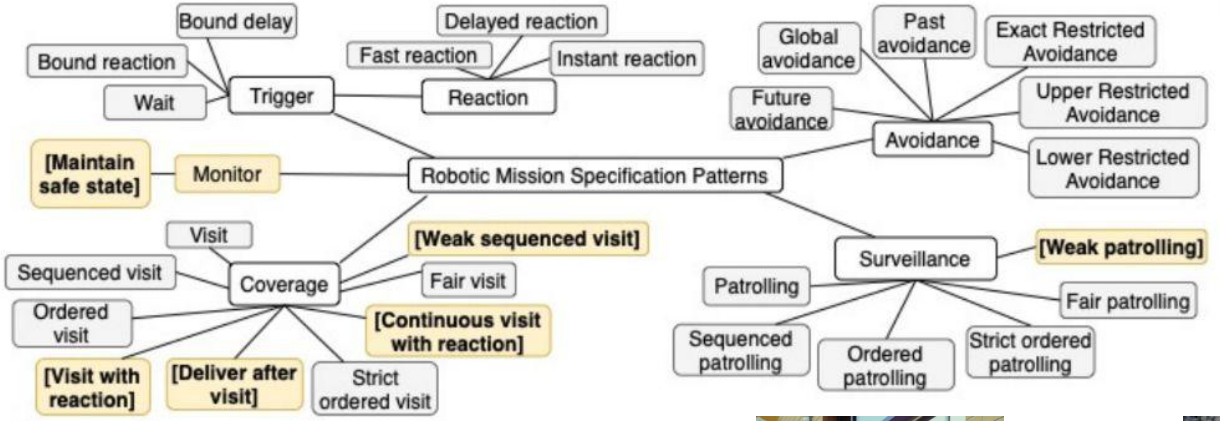
# Uses of FRET

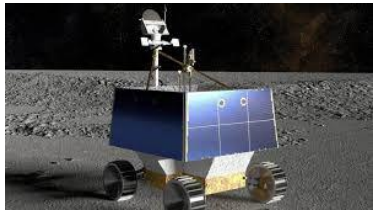



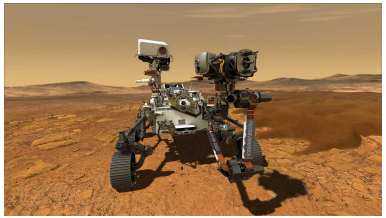

# Specification of Robotic Missions

- Mobile robots separate humans from hazards and inaccessible environments.
- Robotic mission requirements describe the high-level tasks that the robotic system must accomplish.

# Specification Patterns



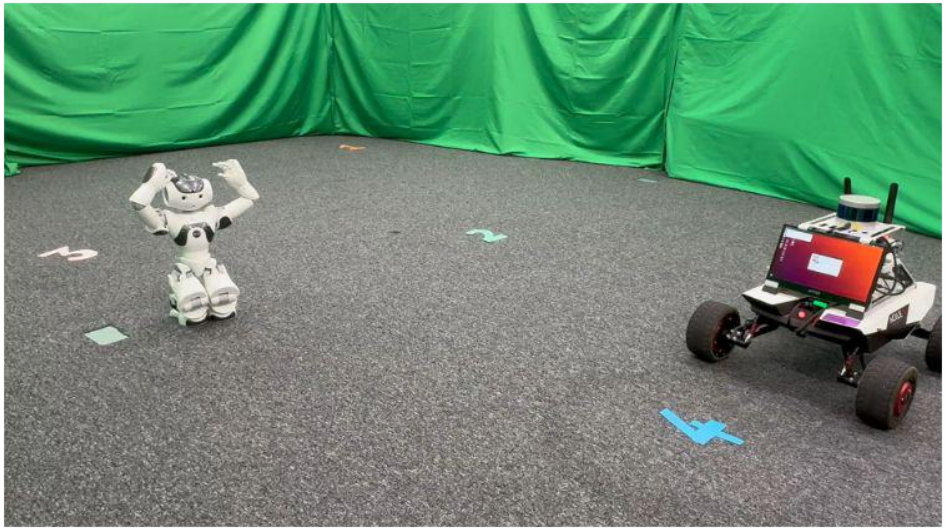





# Pattern: Visit With Reaction

**Robot shall eventually satisfy wp4 & flashlights**

# Pattern: Avoidance



**Robot shall never satisfy astronautPosition**

# Ongoing and Future Work

- Refactoring framework for maintaining requirements (MU-FRET)
- Probabilistic extension to FRETish
- Requirement patterns for Human-Robot-Interaction
- Drone use cases
- NASA’s Viper Mission
- Applicability to multi-robot systems

# References


M. Farrell, M. Luckcuck, R. Monahan, C. Reynolds, O. Sheridan. “**FRETting and Formal Modelling: A Mechanical Lung Ventilator**”. In ABZ. 2024.

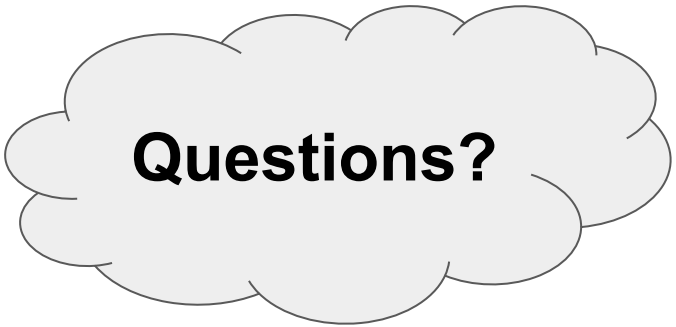
G. Vazquez, A. Mavridou, M. Farrell, T. Pressburger, R. Calinescu. “**Robotics: A New Mission for FRET Requirements**”. In NFM. 2024.

M. Luckcuck, M. Farrell, O. Sheridan, R. Monahan. “**FRETting about requirements: formalised requirements for an aircraft engine controller**”. In REFSQ. 2022.

M. Farrell, M. Luckcuck, O. Sheridan, R. Monahan. “**Towards Refactoring FRETish Requirements**”. In NFM. 2022.

M. Farrell, N. Mavrakis, A. Ferrando, et al. “**Formal Modelling and Runtime Verification of Autonomous Grasping for Active Debris Removal**”. In Frontiers in Robotics and AI. 2022.

H. Bourbouh, M. Farrell, A. Mavridou, et al. “**Integrating Formal Verification and Assurance: An Inspection Rover Case Study**”. In NFM. 2021.


Questions?

# FM|AS International Workshop on Formal Methods for Autonomous Systems

- **Submission:** 9th August 2024 (AoE)
- **Notification:** 26th September 2024
- **Final Version due:** 10th of October 2024
- **Workshop:** 11th and 12th of November 2024



@iFM 2024, Manchester UK
Nov 11th & 12th, 2024

- **Special Issue with SCP:** *Advances in Formal Methods for Autonomous Systems*