

# Requirements

Michael McCool

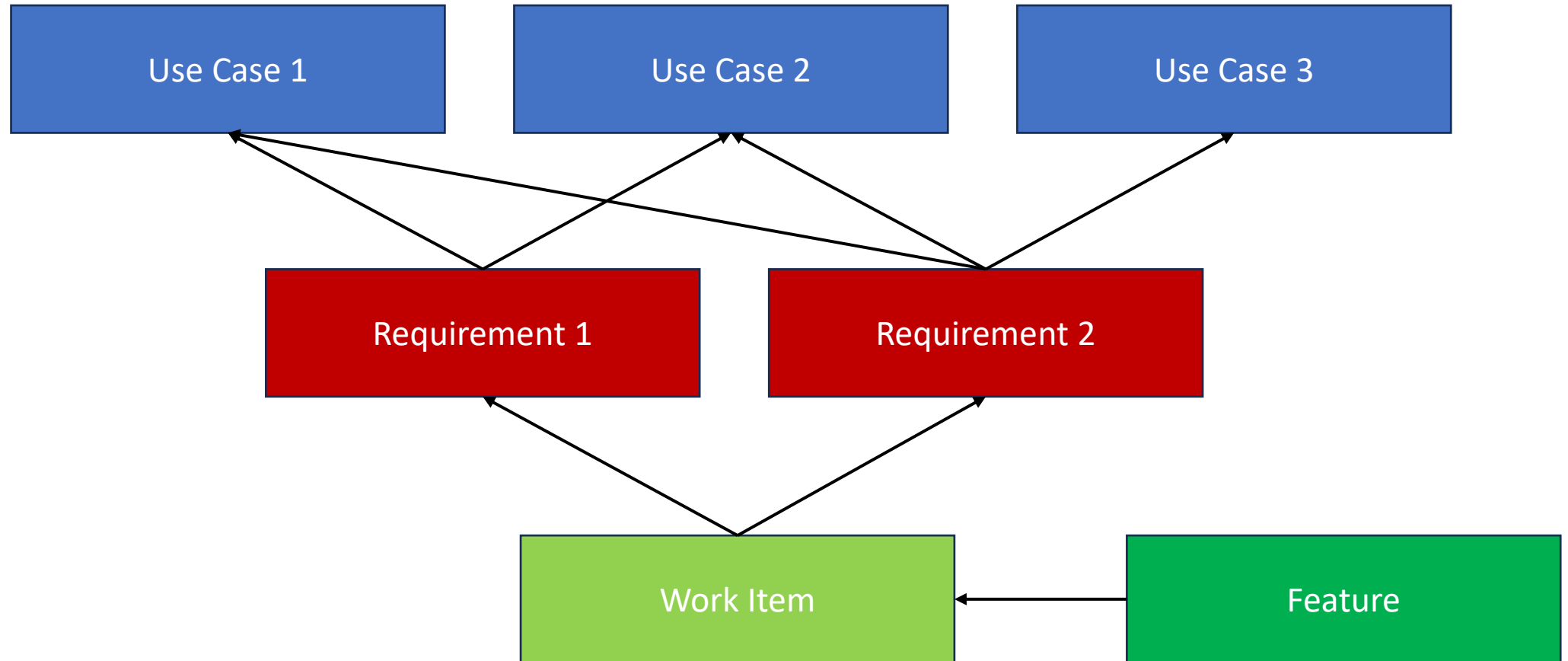
26 September 2024/24 October 2024 (update)

TPAC 2024

# Outline

- Purpose
  - Connecting use cases to features
- Format
  - User stories
  - Function vs technical
  - Examples
- Categories
- Security and Privacy
  - Relationship of requirements to risks
- Suggested Plan

# UC → Req → WI → Feature



# Requirements: Definition

From [IEEE SWEBOK](#):

- a **condition or capability** needed by a **user** to **solve a problem** or **achieve an objective**;
- a **condition or capability** that must be **met or possessed** by a **system** or **system component** to **satisfy a contract, standard, specification** or **other formally imposed document**
- **Summary:**
  - Condition or capability: What
  - User or contract/specification (from some stakeholder...): Who
  - Solve a problem or achieve an objective: Why

# Requirements: Format

- [User Story Template:](#)

As a **STAKEHOLDER**, I want **CAPABILITY** so that **PURPOSE**.

## **STAKEHOLDER:** Who

- Identifies primary user or beneficiary.
- Secondary stakeholders may also be identified in **PURPOSE**.

## **CAPABILITY:** What

- May or may not map onto a single specific work item or feature (discuss).
- Should be specific enough that it is clear when it is satisfied.
- Should be satisfiable with finite effort.

## **PURPOSE:** Why

- Larger context of goal, other stakeholders (e.g. support another SDO)
- May not be finitely satisfiable.

# Requirements: Template

- **Who: Stakeholder**
  - User or Org (e.g. SDO)
- **What: Capability**
  - Technical requirement
  - May also be a Condition that needs to be satisfied (e.g. “minimize size”)
  - *Link to at least one work item/issue/PR/assertion*
- **Why: Purpose**
  - Functional requirement
  - Solve a Problem
  - Meet an Objective
  - *Link to at least one Use Case or Use Case Category (e.g. “ease of use”)*
- **Details (optional):**
  - Free-form extra details as needed

# Functional vs. Technical Requirements

## Functional: Why

- ***Purpose*** of needed functionality

## Technical: What

- ***Capability*** to support needed functionality
- Don't need to separate functional and technical requirements
  - User story format includes both!
  - Tends to form a natural hierarchy: many “capabilities” may support one “purpose”
- Technical requirements (capabilities) should be finitely satisfiable!
  - For example, “WoT systems should have good security” is a bad technical requirement, it's unclear when it is (or can ever be) fully satisfied.
  - It *may* be acceptable as a functional requirement, although it's still a bit vague

# Requirements: Examples

- As a *consumer of WoT TDs*,  
I want *the ability to poll the status of actions*  
so that *I can take corrective action or cancel unneeded actions*.
- As a *WoT System Owner*,  
I want *to be able to control who has access to individual entries in a WoT TD Directory and revoke their access at any time*  
so that *the security of my system can be maintained*.
- As a *producer of WoT TDs*,  
I want *to be able to publish a short form of TDs specifying only the variables to be filled into a TD Template*  
so that *network bandwidth can be minimized*.

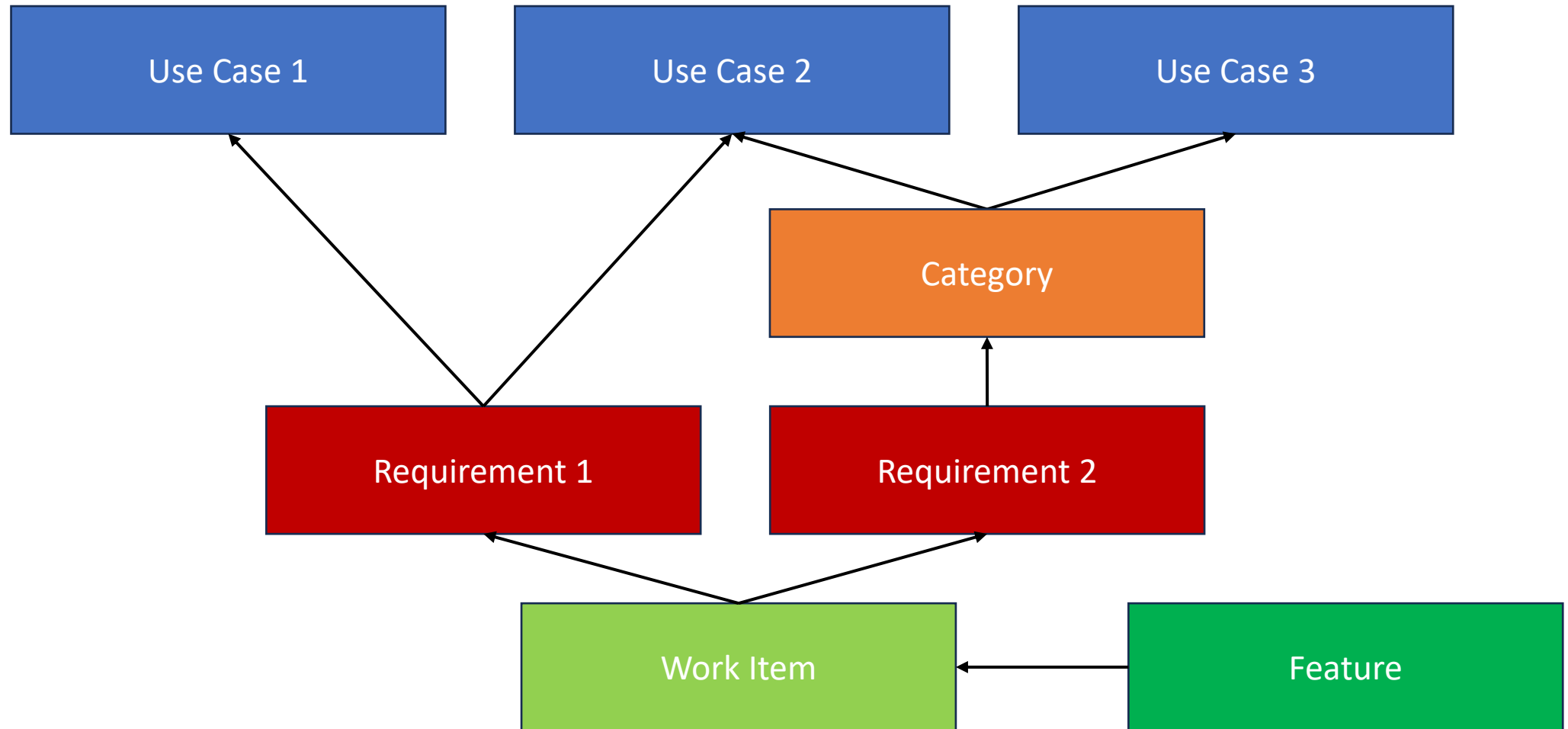


# Requirements: Examples

Based on <https://github.com/w3c/wot-thing-description/issues/2039>

- As a ***consumer of WoT TDs***,  
I want ***to know when or if writing a property returns a value***  
so that ***I can understand when I can use this value to confirm writes.***
- As a ***producer of WoT TDs***,  
I want ***to be able to specify simple security schemes inline***  
so that ***TDs are less verbose and easier to write in simple cases.***
- As a ***consumer of WoT TDs***,  
I want ***to identify TDs limited to a finite feature subset***  
so that ***I can ensure interoperability.***
- As a ***producer of WoT TDs***,  
I want ***to signal when TDs have been limited to a finite feature subset***  
so that ***I can ensure interoperability.***

# UC → Cat → Req → WI → Feature



# Categories

- Intermediate, *optional* step but allows for generalization
  - Avoids having to constantly update “requirement” to “use case” mapping
  - ***JUST A CONVENIENCE*** when many use cases share common requirements

Some possible categories:

- Private (handles personal or confidential information)
- Flexible Protocol Usage (use multiple protocols)
- Cloud Integration (shares data with remote servers)
- Local Access (needs to operate without a global connection)
- Mobile (location is subject to change)
- Resiliency (needs to be robust to failures and attacks of various kinds)

# Workflow Considerations

- It is up to each TF to decide how to organize work items
  - MD files → issues → PRs → assertions
  - End state should be an assertion, however (e.g. a feature in a specification) for "capabilities".
- For Use cases
  - If one does not exist, it should be created
  - Keep it abstract
  - Don't need a ton of implementation details in use case, it should just state the purpose
  - A category can be suggested in the user story
    - Give a definition or link to at least one use case in the "details" section

# Security and Privacy Requirements

## Special case:

- Security/Privacy features are generally to mitigate “risks”
  - S&P sections generally each have a defined risk
  - ... then list mitigations for each risk, some of which may be normative
  - In general, mitigations map to capabilities and avoiding risks are purposes
- Risks are documented in “Security and Privacy Guidelines” document
  - Stakeholders need to be made consistent with other documents
- ***Need to identify which use cases have which risks***

# Suggested Plan

- Expand “Requirements” Section in Use Cases and Requirements document to define requirements and connect them to use cases.
  - Want to avoid editing use cases themselves for authorship and consolidation reasons
  - Consolidate: move requirements out of other documents, e.g. Architecture
- Keep it simple:
  - A named requirement and a user story defining each one is enough.
    - Optionally can have additional description paragraph
  - Can link to another document for more detailed definitions, e.g. security risks.
    - Ideally, linked details should be in a “published” document, not a random MD file somewhere...
  - Links to use cases motivating each requirement
  - Do not have to link each requirement to ALL use cases motivating it
    - Use categories *only* if requirement is motivated by large set of use cases
  - **Discuss:** do we want to allow “hierarchies” or “sets” of requirements?
    - The PURPOSE does this, partially
    - But it may still be helpful to cluster or group requirements somehow
    - Can do it when and if we need to...

# Discussion

