Category Theory Framework for Variability Models with Non-Functional Requirements

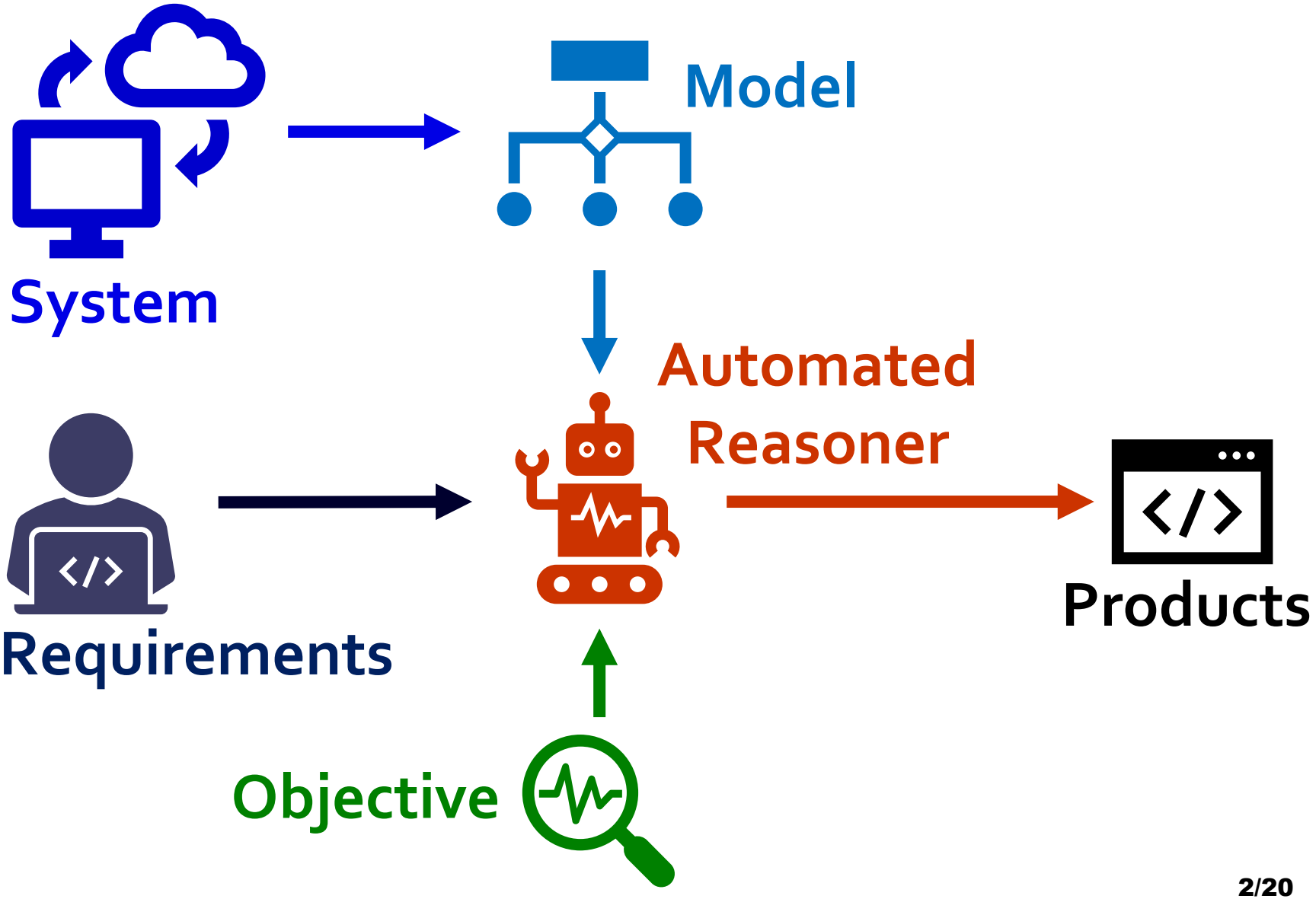
Daniel-Jesus Munoz, Mónica Pinto and Lidia Fuentes
ITIS Software, Universidad de Málaga, Andalucía Tech, Spain

Dilian Gurov
KTH Royal Institute of Technology, Stockholm, Sweden

CAiSE'21
Melbourne, Australia. July 2021

https://doi.org/10.5281/zenodo.5062145

Software Product Lines Analyses



The Situation

- **Client**: Could you analyse our highly configurable systems?
- **Software Product Line engineer**: Absolutely! Describe one:

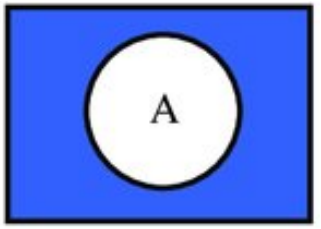
- These are the components and alternatives | Ok, a SAT…
- …,some arithmetical requirements | an SMT/CP could…
- …,products quality measurements | a hybrid model…
- …, system requirements | then cross-constraints…
- …, crossed quality requirements | WAIT!

- **Dear Santa Claus…**

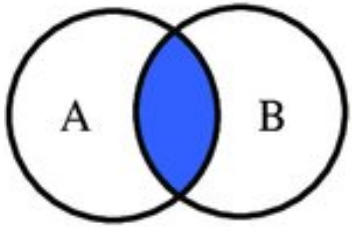
Our Scenario: Edge Computing

The SPLE Trend in Modelling

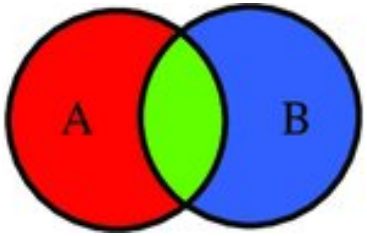
1. Extending pseudo-standards:



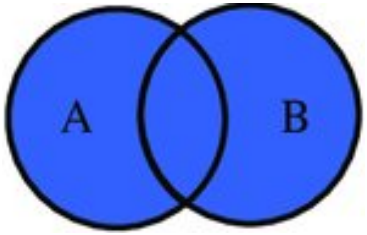
2. Creating new pseudo-standards:



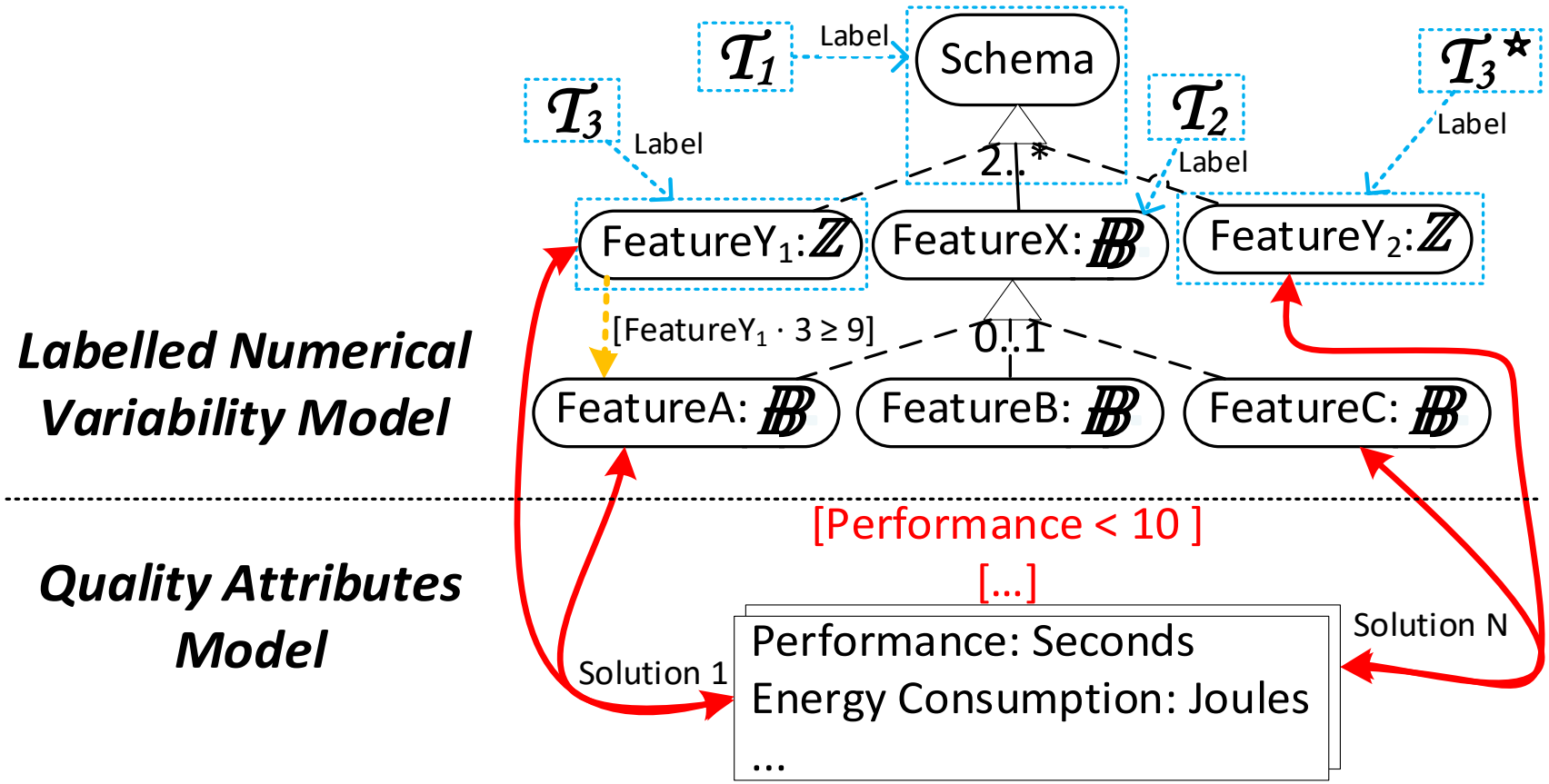
3. Transformations or hybrid models:

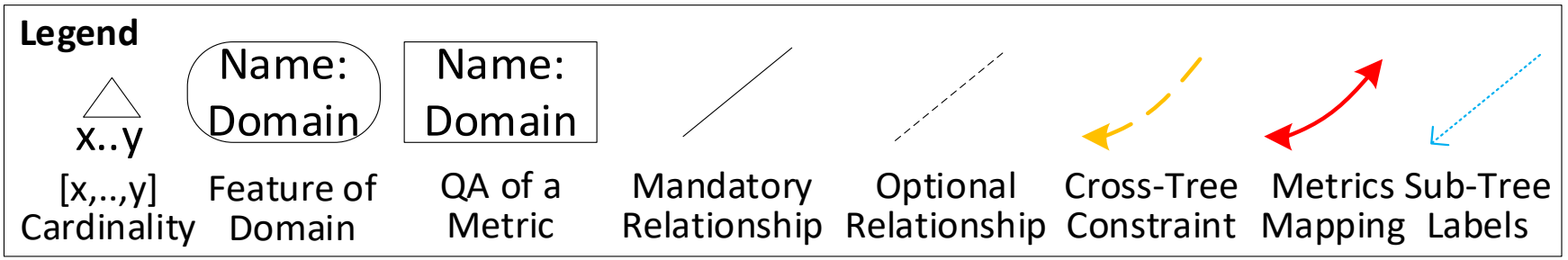


4. **Unification**:



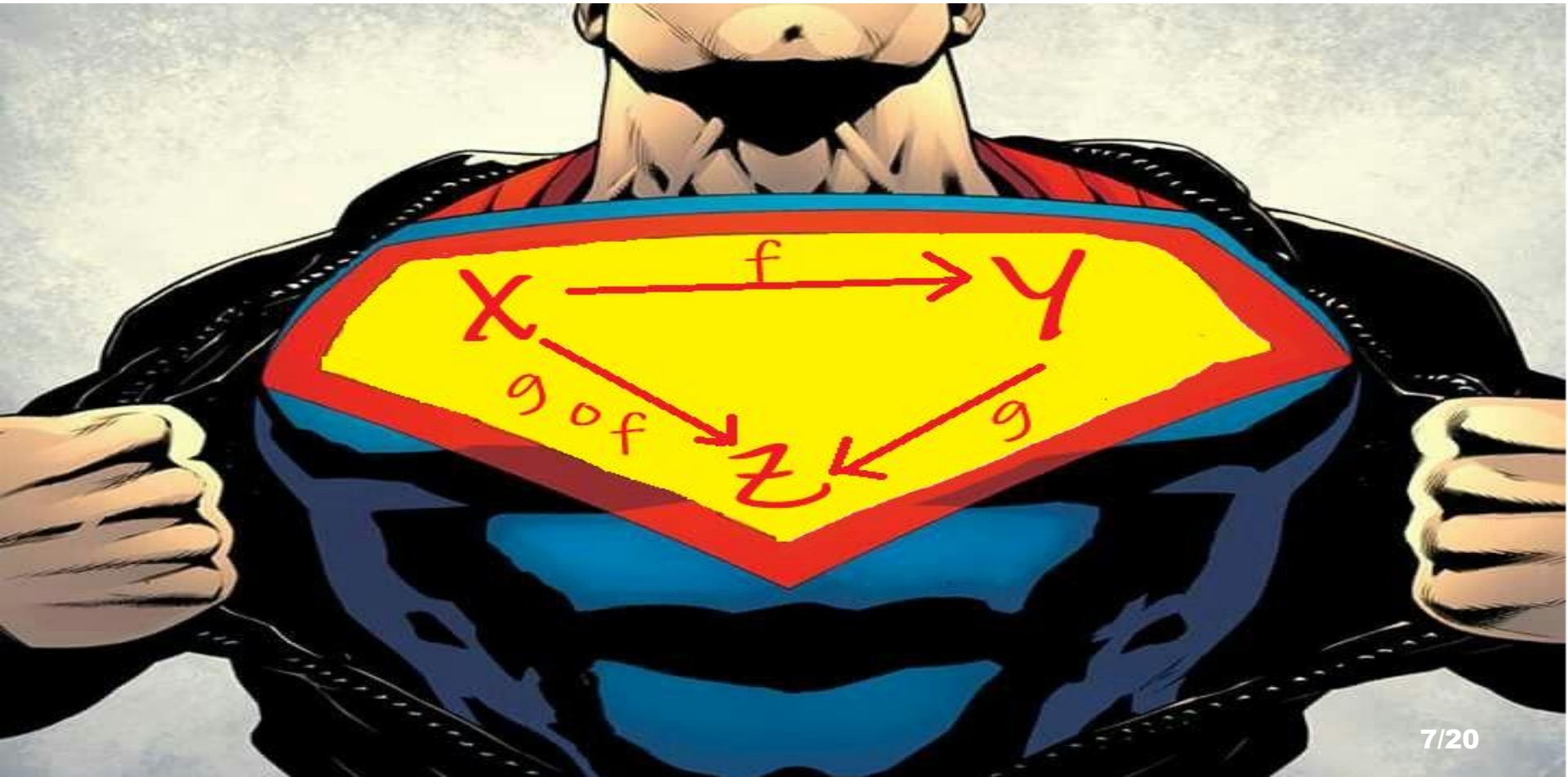
Variability and Quality Attributes Modelling





Category Theory to the Rescue!

The unifier of apparently dissimilar models while abstracting individual specifics



Applied Category Theory: Crash Course

- Category: Represent related spaces
  - Objects: Structured classes ($\bullet X$)
  - Arrows: Structure preserving functions ($\bullet X \xrightarrow{a} \bullet Y$)
    - Identity: $\bullet X \xrightarrow{identity} \bullet X$
    - (Associative) Composition: $\bullet X \xrightarrow{a_1 \circ a_2} \bullet Z$
- Functor: Mapping among Categories ($\bullet C \xrightarrow{F} \bullet D$)
- Path: $\bullet X_0 \xrightarrow{a_1} \bullet X_1 \dots \bullet X_{n-1} \xrightarrow{a_n} \bullet X_n$
- Generalised Element: Select $\bullet U \xrightarrow{element} \bullet X$
- Instance: A set-valued functor for elements.

Category Theory Flexibility

- A single object can form a category
- A category can be formed by, and divided into, sub-categories.

Our Proposal: A Category Theory Framework

- Models are **categories**.
- Relationships, constraints and requirements are **arrows**.
- Features definitions (i.e., nodes) are **generalised elements**, and their values are **instances**.

Category: Numerical Variability Model

$\mathcal{NVM}$ Objects:

Tree$_1$: $\mathcal{T}_1$

Tree$_2$: $\mathcal{T}_2$

Tree$_3$: $\mathcal{T}_3$

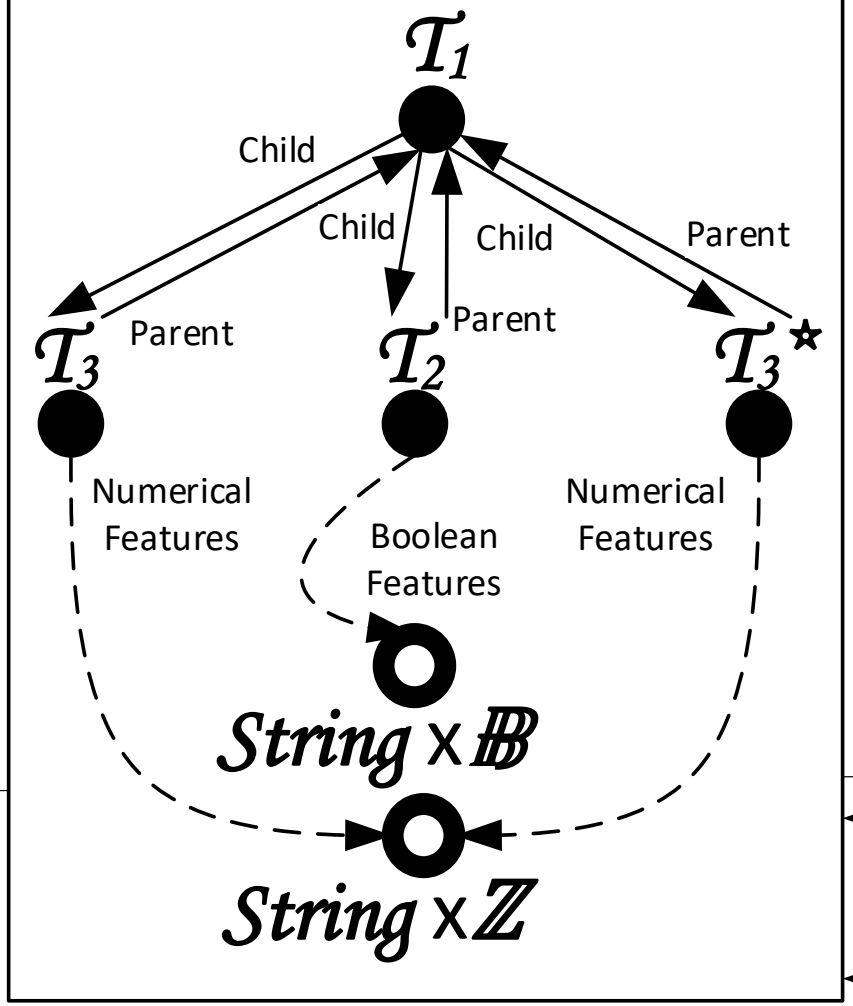
Tree$_3$ Clone: $\mathcal{T}_3^\star$

Integer Set: $\mathbb{Z}$

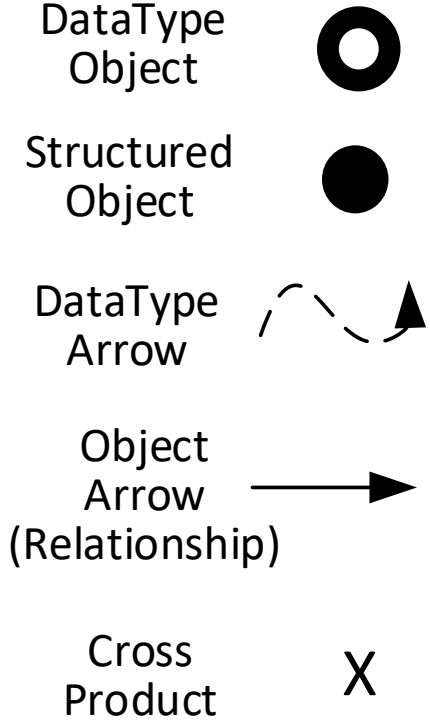
Character Set: $String$

Boolean Set: $\mathbb{B}$

$\mathcal{NVM}$ Category Example



Legend:



Category: Quality Attributes Model

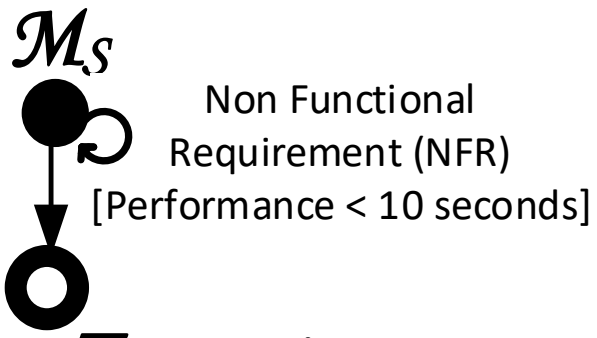
QAM Objects:

Metric Set: $\mathcal{M}_S$

Integer Set: $\mathbb{Z}$

Character Set: $String$

QAM Category Example



Legend:

DataType Object

Structured Object

DataType or NFR Arrow

Cross Product X

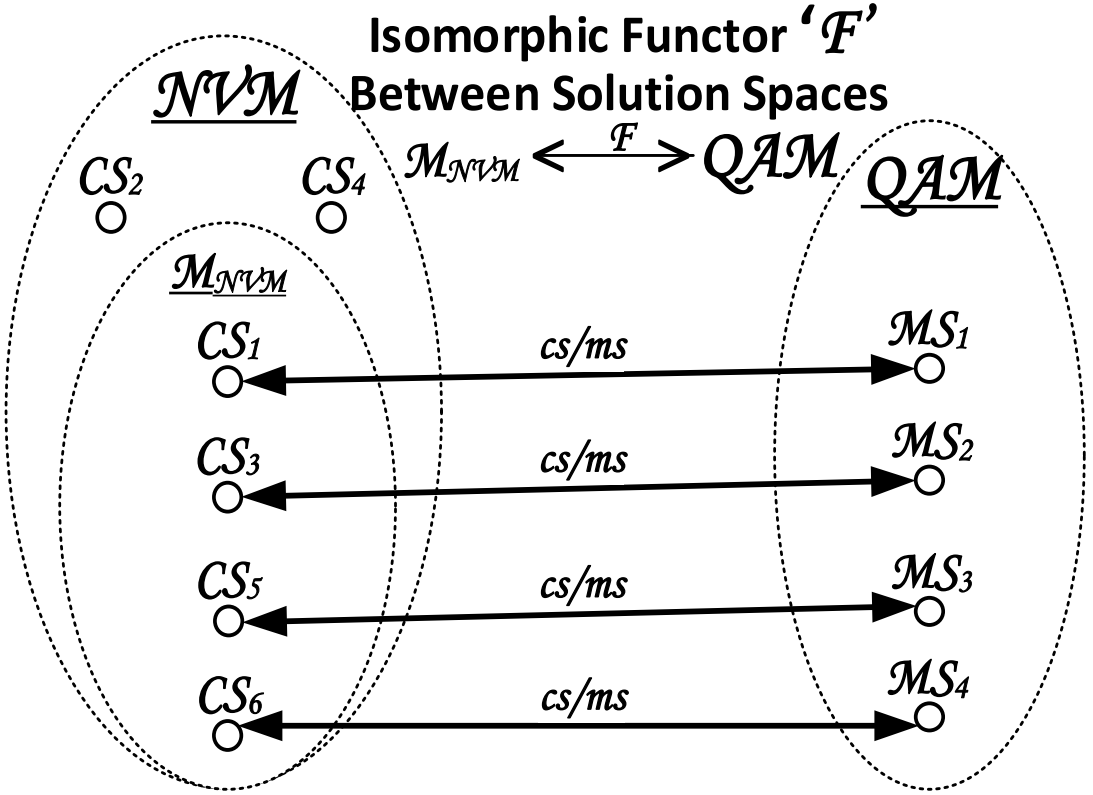
Category: Solution Space

Abstract Components:

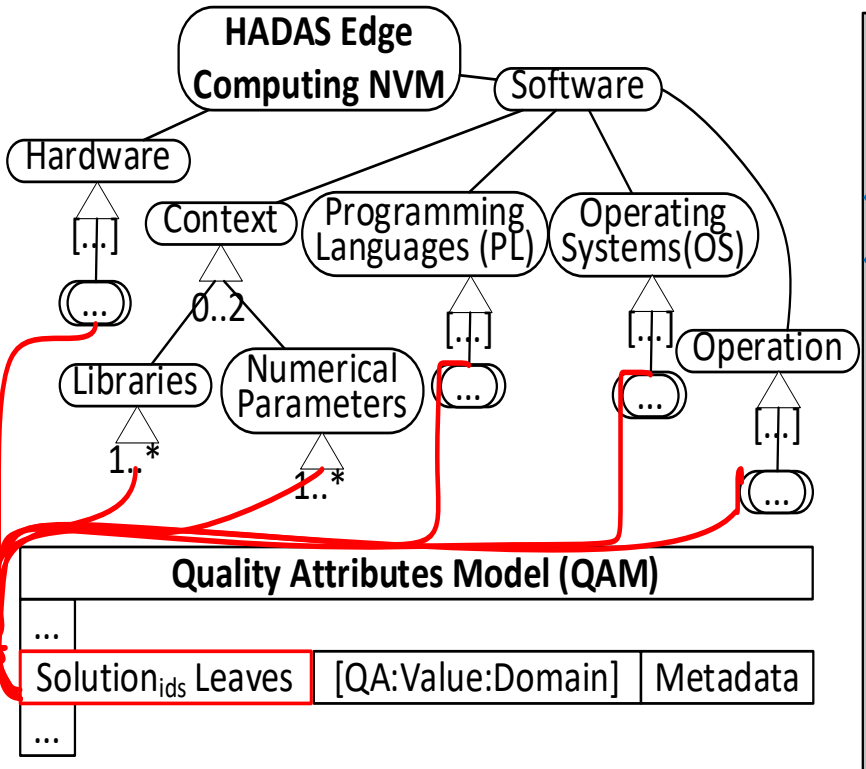
- **Numerical Variability Model Category:** $\mathcal{NVM}$
- **Measured NVM Sub-Category:** $\mathcal{M}_{\mathcal{NVM}}$
- **Quality Attributes Model Category:** $\mathcal{QAM}$
- **Complete Solution Object:** $\mathcal{CS}$
- **Measurements Set Object:** $\mathcal{MS}$

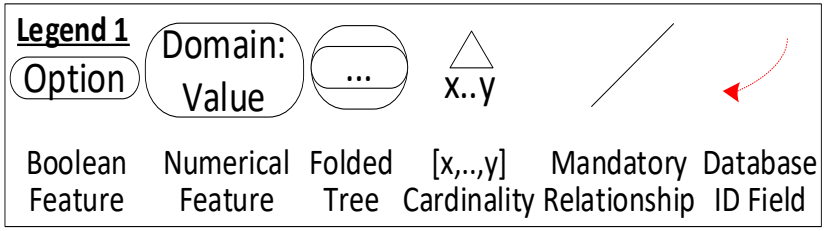
Legend

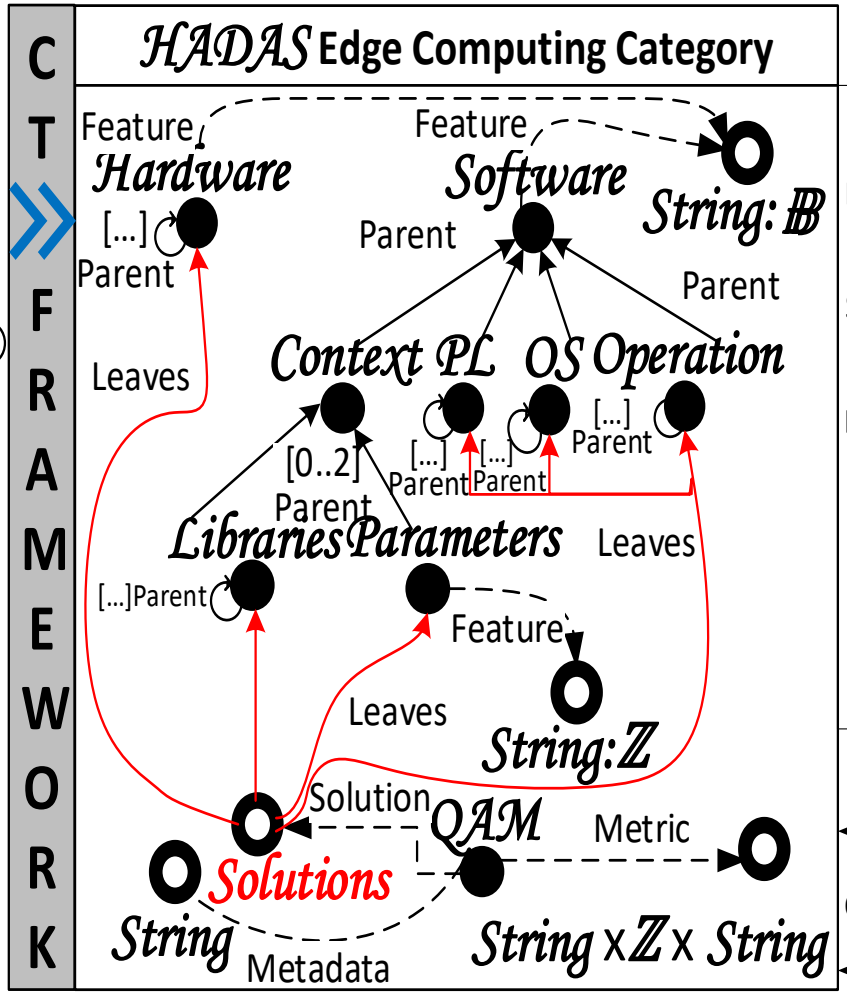
Schema Solution Space | Instances | Object Arrow (Relationship)

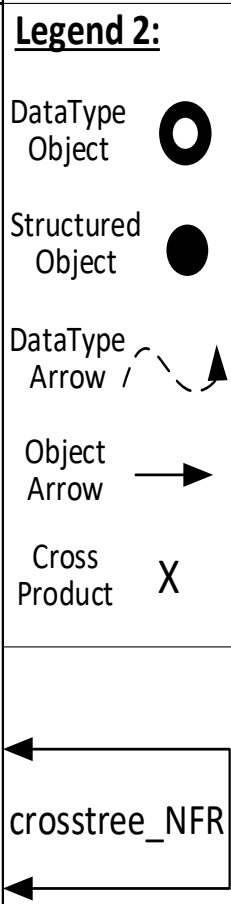


Validation: HADAS Edge Computing Category

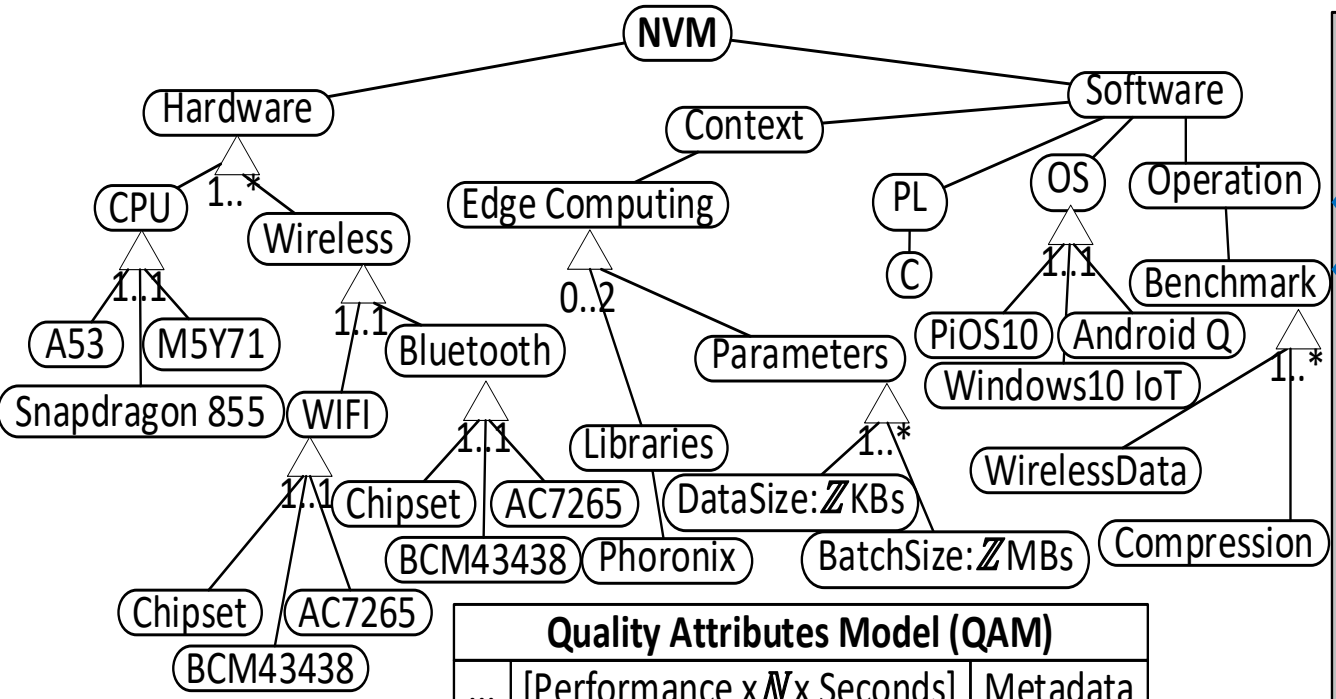




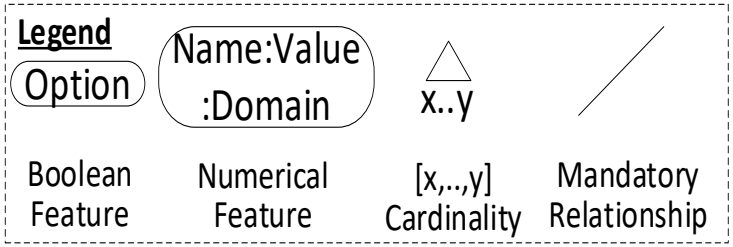




Evaluation: Edge Computing Case Study



Quality Attributes Model (QAM)		
...	[Performance x ℕ x Seconds]	Metadata
...	[Energy rate x ℕ x mWatts]	Metadata



CTFRAMEWORK

```
    Bool    = "Boolean.pars
    Int     = "Integer.parse
    String = "input[0]"
java_functions
    Solutions : String,list
}
schema HADAS = literal : Ty
    entities
        Hardware
        Software
        OS
        PL
        Operations
        Context
        Libraries
        Parameters
        QAM
```

CQL IDE

Different Automated Reasoning

From a constraint programming solver for variability plus MariaDB for quality attributes to…

…an automated theorem prover with Knuth-Bendix completion for logic and equations, and hashing, balanced trees and chasing for data-type and cross-object arrows.

CQL IDE Scalability with our Models

- We generated 162 products with their respective 324 measurements with a reasoning time of 0.1 seconds.
- Sub-categories also takes 0.1 seconds
- 3 times cross-product takes 0.2 seconds.

➢ Hence, CQL IDE scales linearly, and the minimum runtime is 0.1 seconds.

Results and Optimisation Analysis

- Compressing data increases the energy-rate.
- More powerful CPUs barely affects energy consumption besides if compressing.
- While communication peripherals affect similarly, WiFi is slightly more energy efficient for large data sizes, and vice-versa.

ACKNOWLEDGMENTS

- Eu-ropean Union's H2020 research and innovation programme under grant agreement DAEMON 101017109.
- Projects co-financed by FEDER funds LEIA UMA18-FEDERJA-15, MEDEA RTI2018-099213-B-I00 and Rhea P18-FR-1081.
- PRE2019-087496 grant from the Ministerio de Ciencia e Innovación

Conclusions and Future Work

1. Extended Variability models with quality requirements can be represented as categorical objects and arrows.
2. Our category theory framework is a viable approach, and CQL IDE is a scalable eco-system that can support efficient SPLE analyses.

1. Support more types of extensions and analyses.
2. Integrate quality models and larger SPLEs.

Thank You!


PhD. Student Daniel-Jesus Munoz
Email: danimg@lcc.uma.es
Universidad de Málaga
Spain