When and How to Use Multi-Level Modelling

Authors: J. De Lara, E. Guerra & J. Sánchez Cuadrado

Presenter: Nick Fung

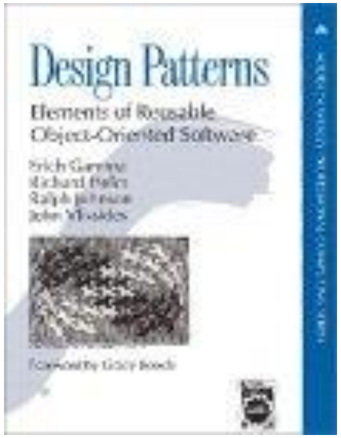
January 29, 2018

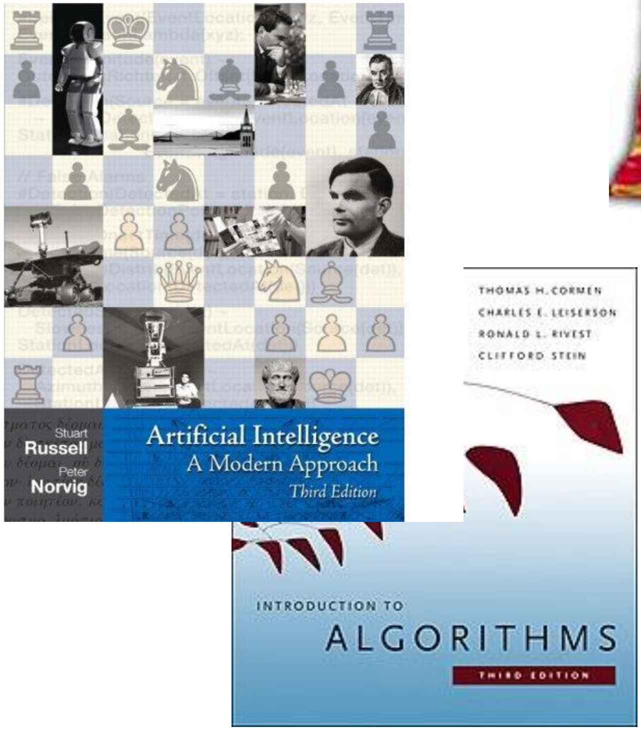
Running Example

- Task
  - Create a model-based computer system for managing the products sold in a bookstore

Running Example

- Task
  - Create a model-based computer system for managing the products sold in a bookstore





Standard Metamodelling Architecture

- **M0 (Terminal) Model**
  - Representation of a system

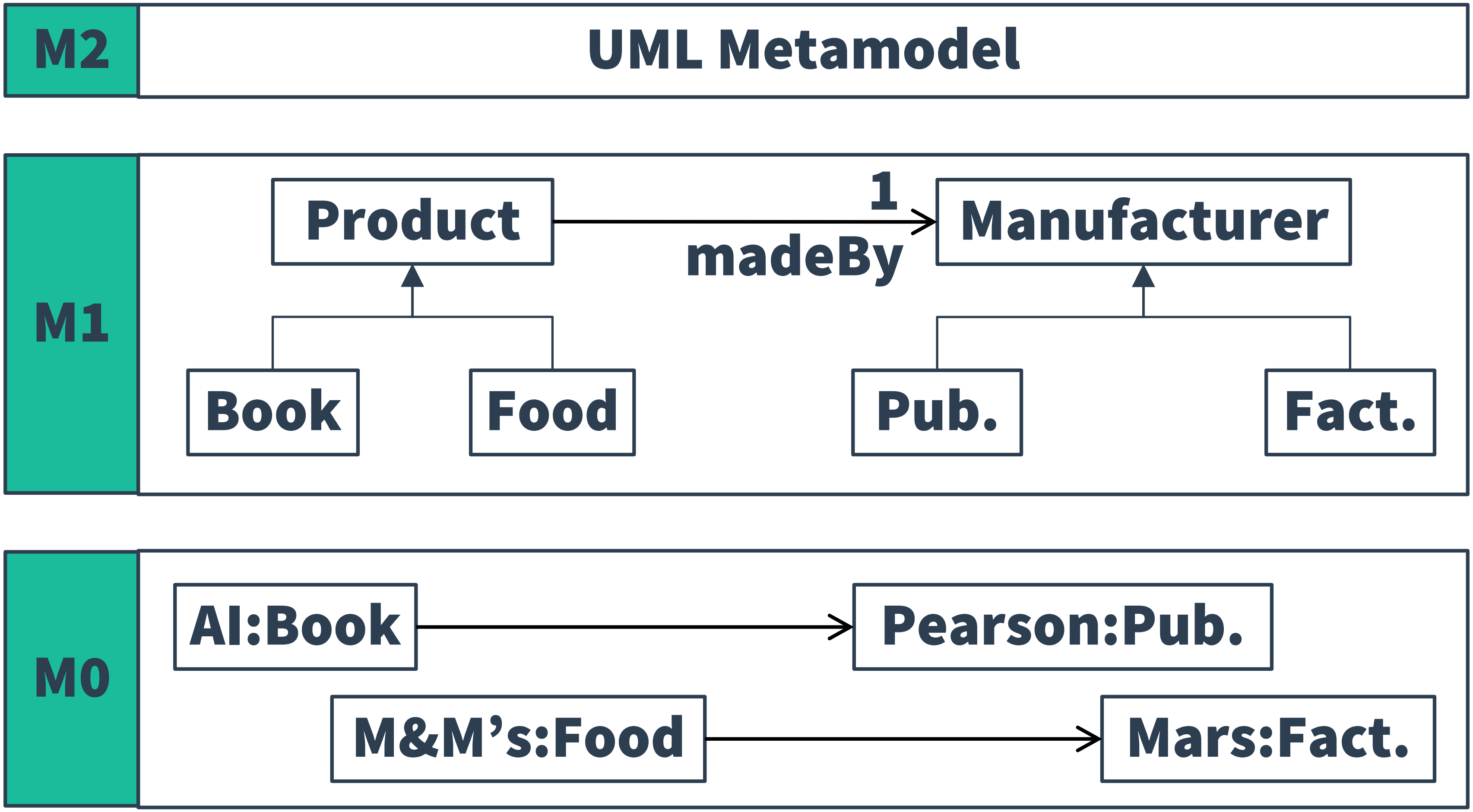
Standard Metamodelling Architecture

- **M0 (Terminal) Model**
  - Representation of a system
- **M1 Metamodel**
  - Abstract syntax of models
  - Prescription of what can be represented

Standard Metamodelling Architecture

- **M0 (Terminal) Model**
  - Representation of a system
- **M1 Metamodel**
  - Abstract syntax of models
  - Prescription of what can be represented
- **M2 Metametamodel**
  - Metamodelling facilities
  - Auto-descriptive

Standard Metamodelling Architecture

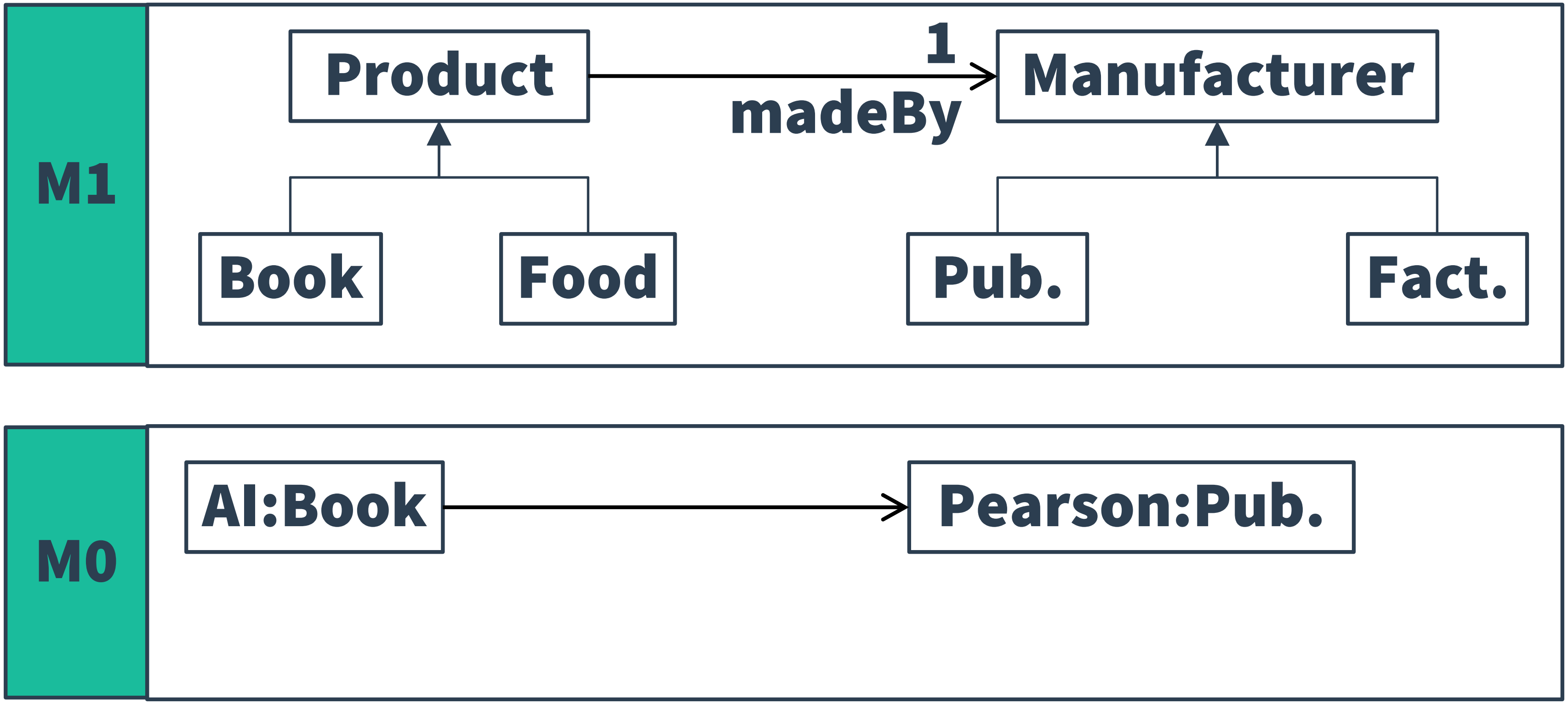


Limitations of Architecture (1)

- **Type-Objects (Clabjects)**
  - New types (i.e. classes) cannot be instantiated dynamically

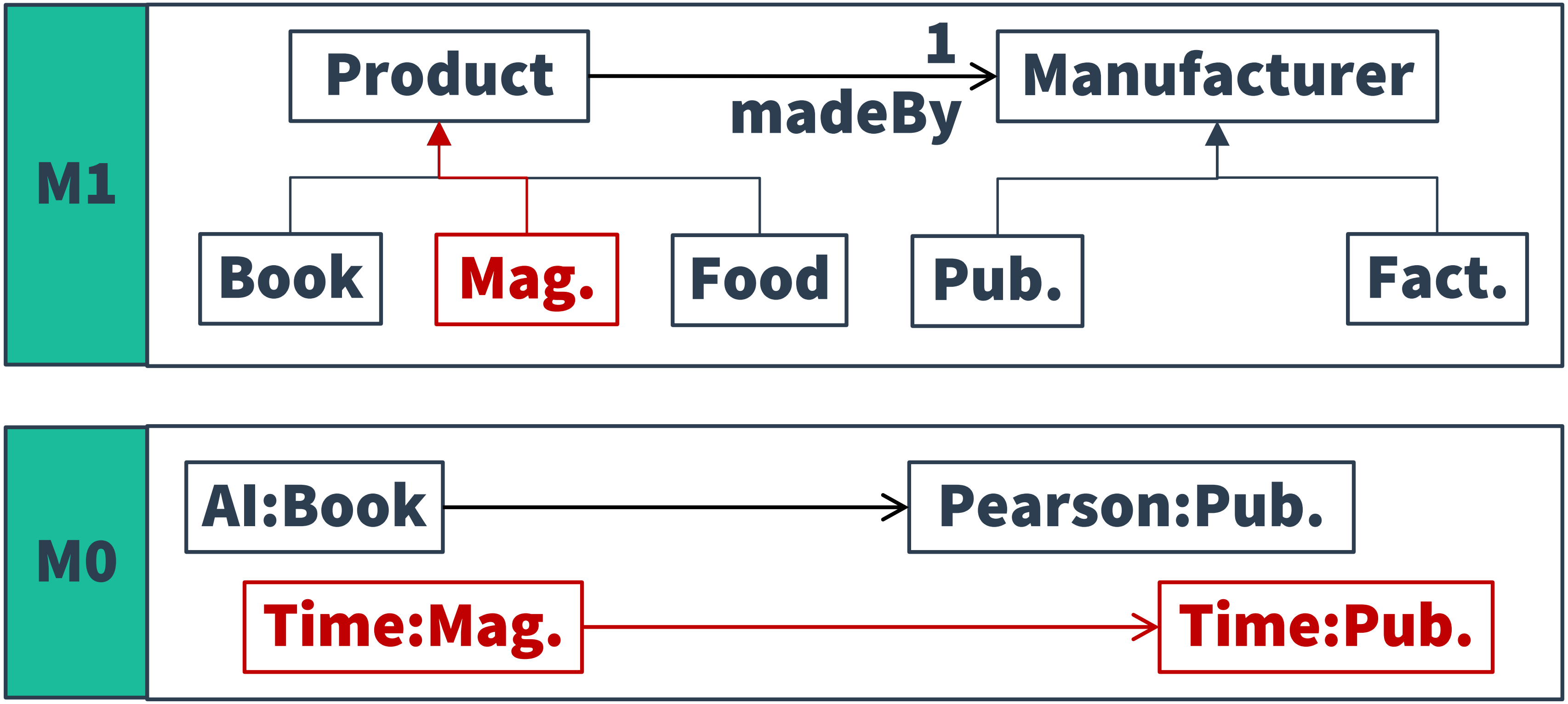
Limitations of Architecture (1)

- Type-Objects (Clabjects)
  - New types (i.e. classes) cannot be instantiated dynamically



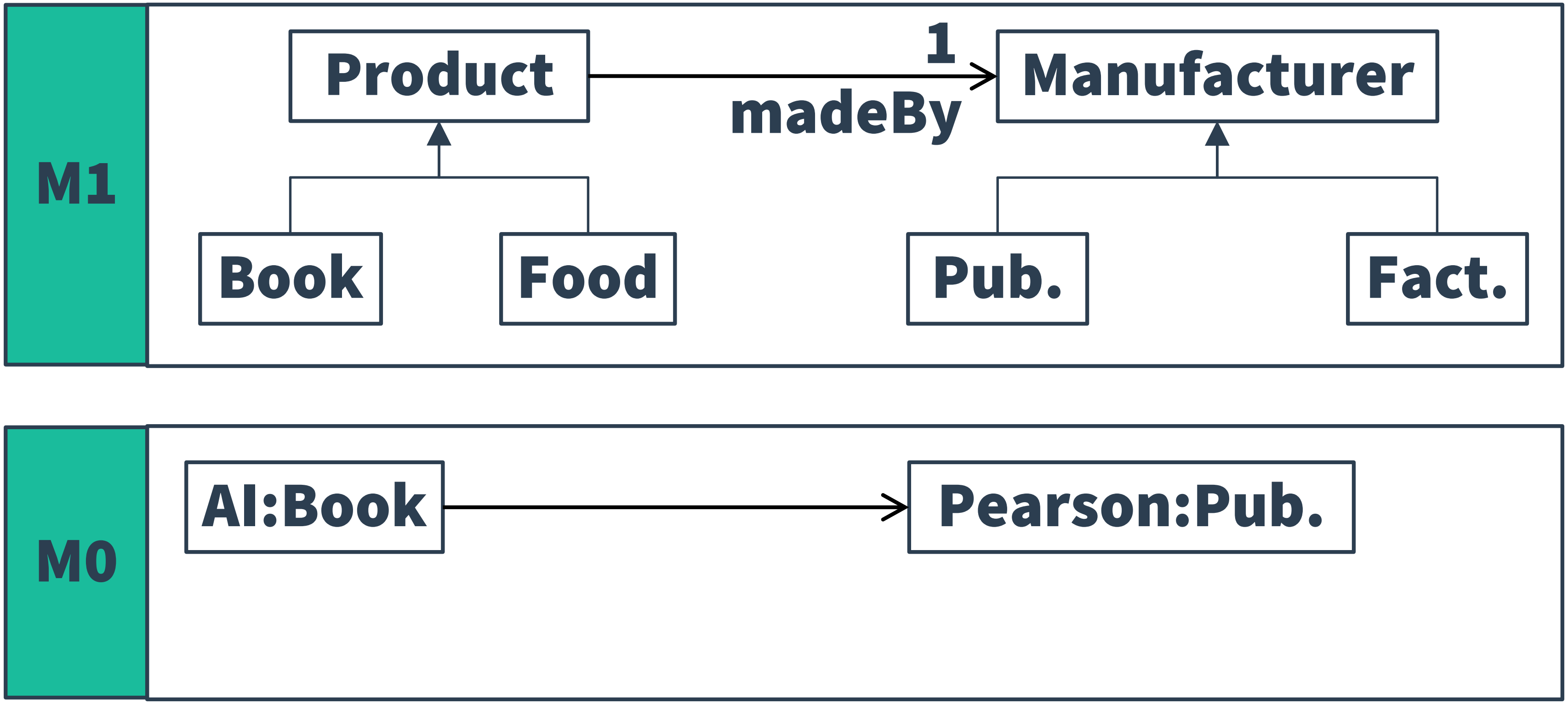
Limitations of Architecture (1)

- Type-Objects (Clabjects)
  - New types (i.e. classes) cannot be instantiated dynamically



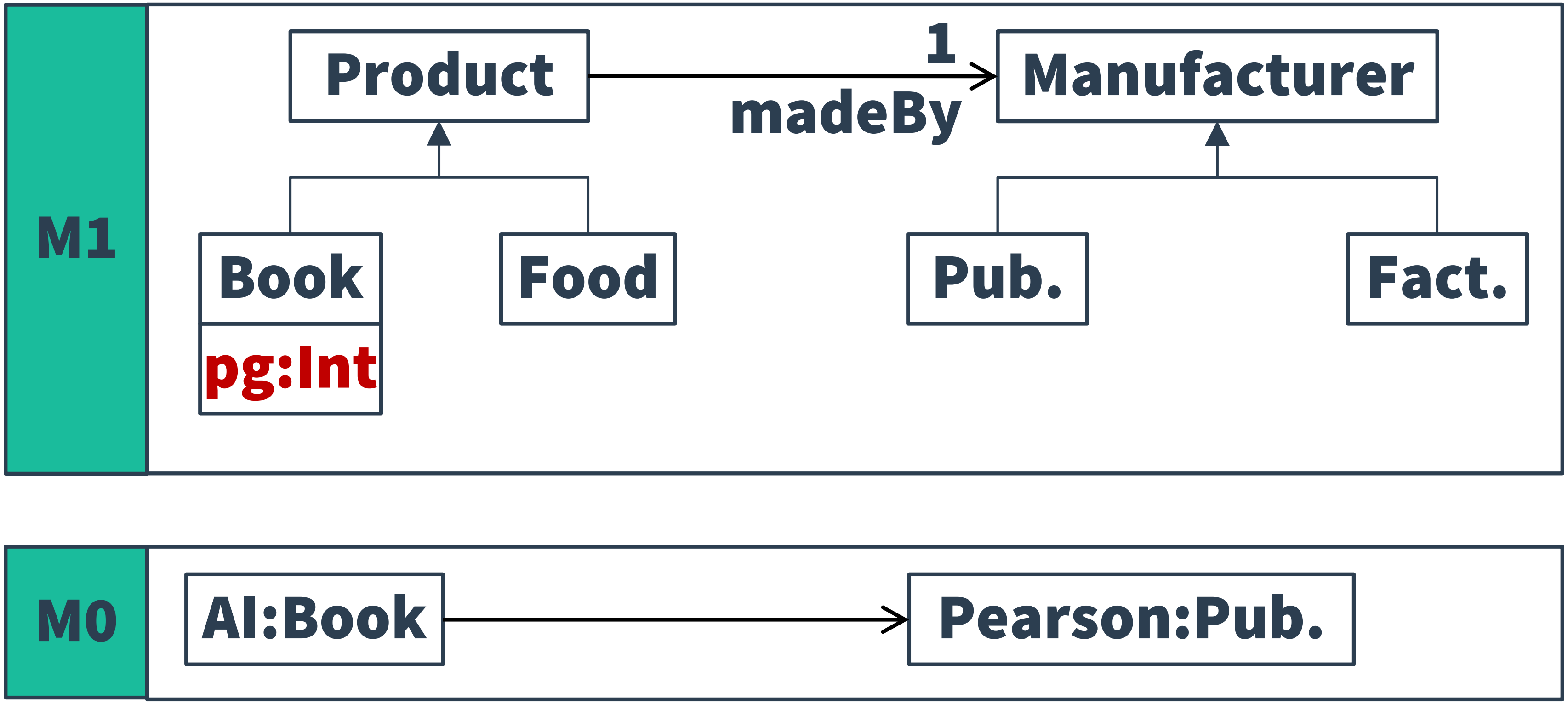
Limitations of Architecture (2)

- Dynamic Features
  - New features cannot be added to a type



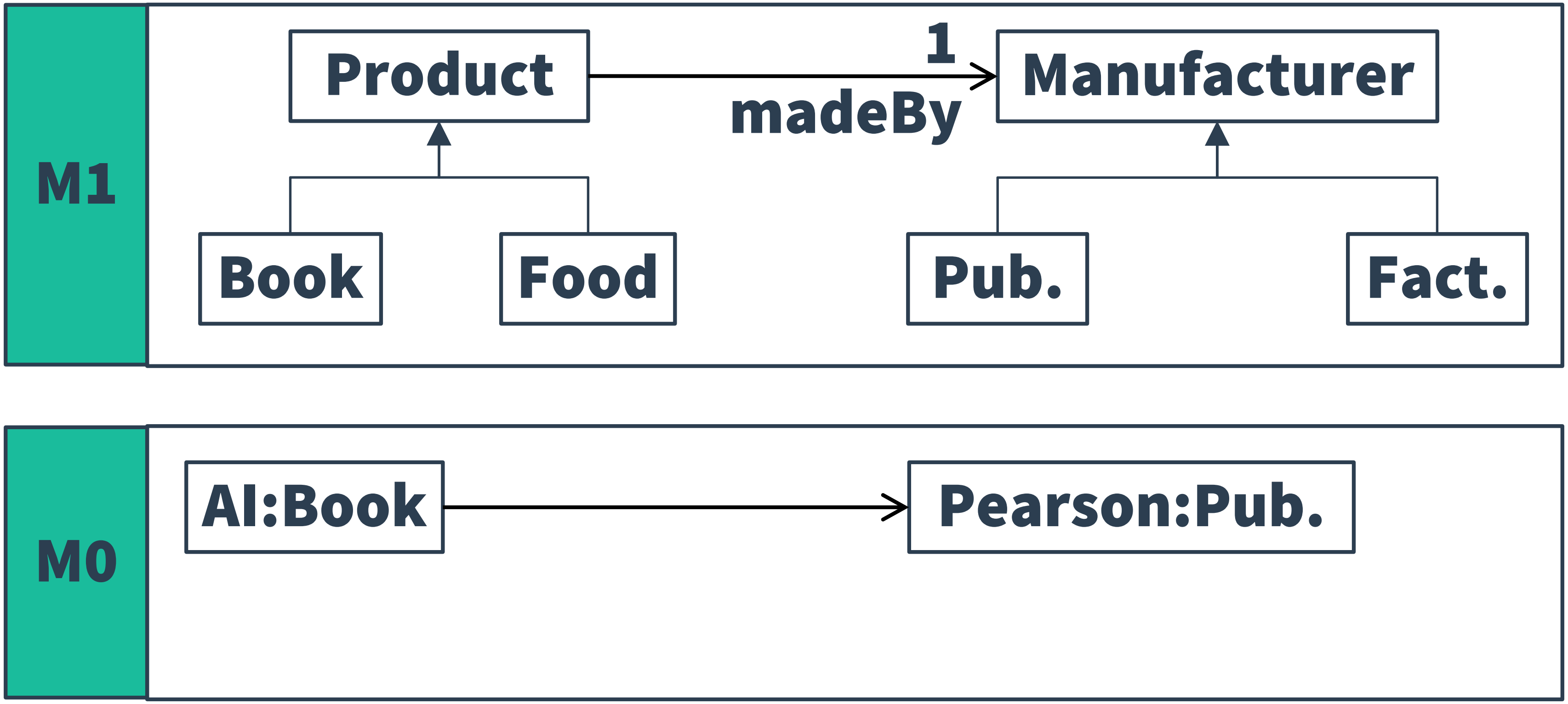
Limitations of Architecture (2)

- Dynamic Features
  - New features cannot be added to a type



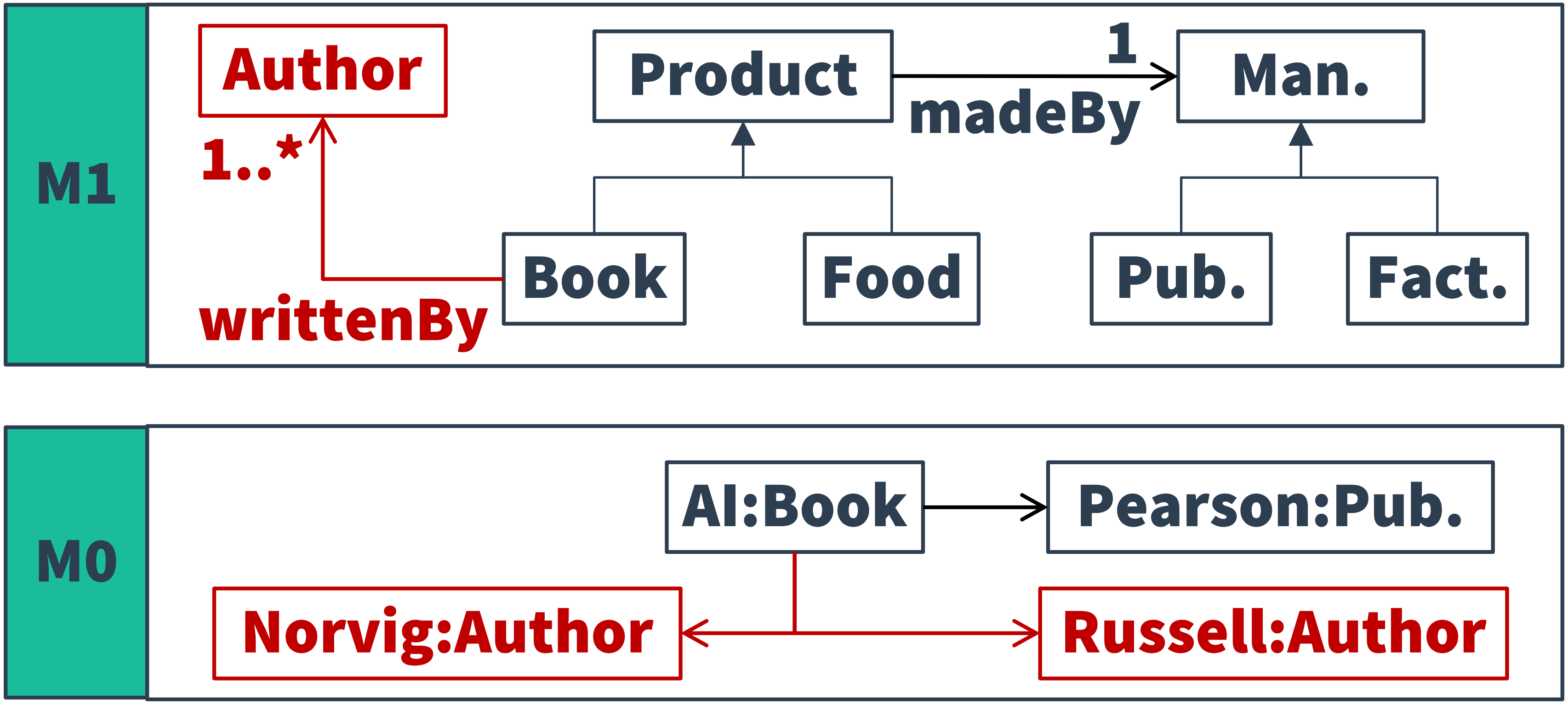
Limitations of Architecture (3)

- Auxiliary Domain Concepts
  - New entities relevant to an existing type cannot be added



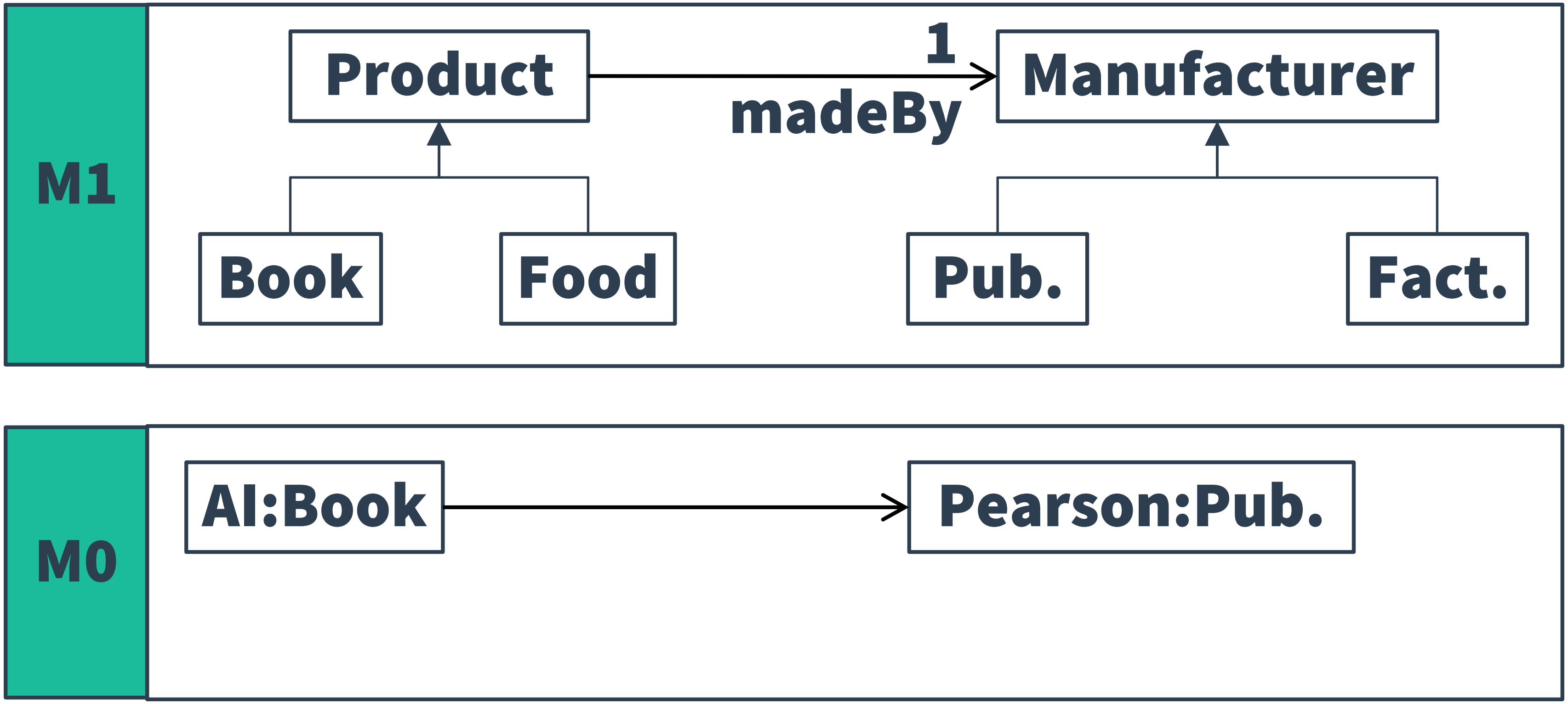
Limitations of Architecture (3)

- Auxiliary Domain Concepts
  - New entities relevant to an existing type cannot be added



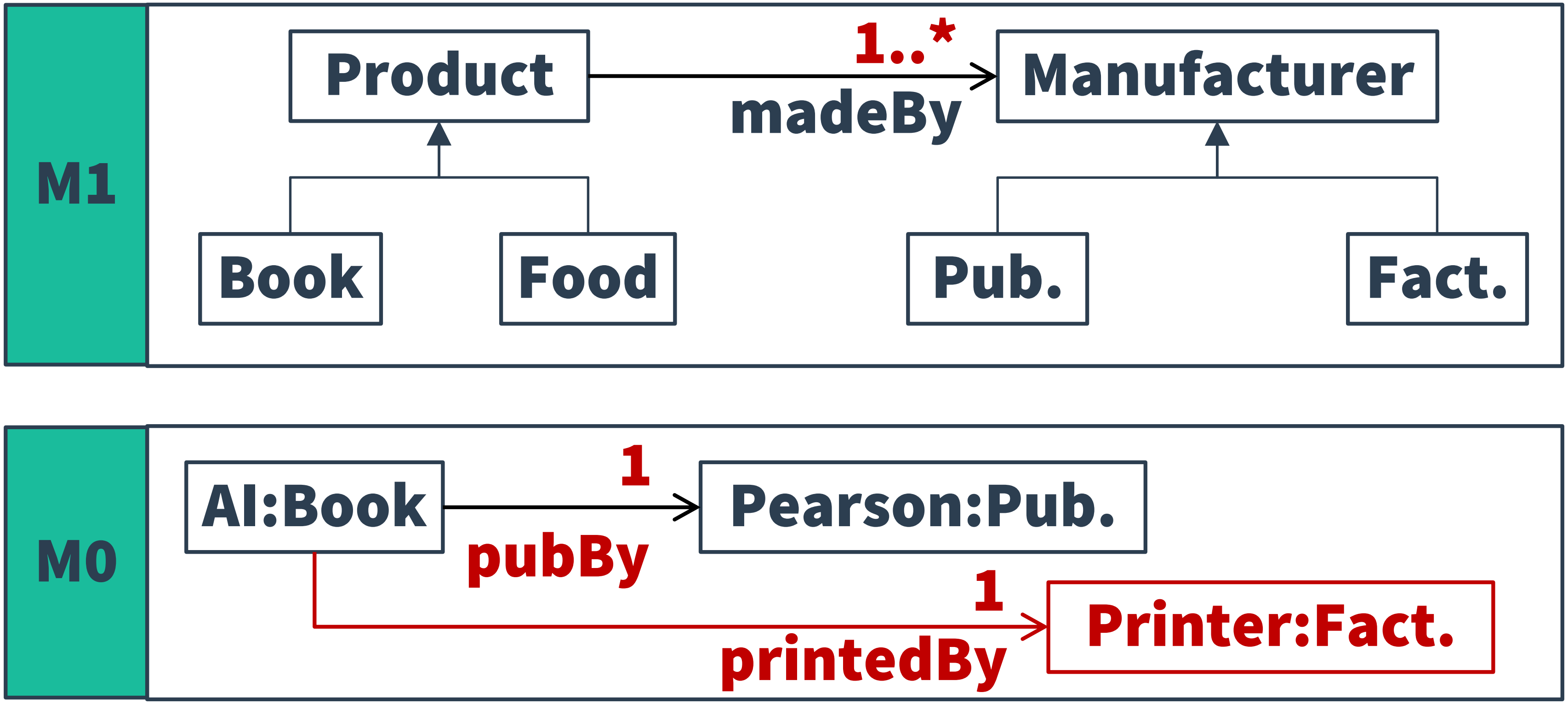
Limitations of Architecture (4)

- Relation Configuration
  - Reference types (i.e. relations) cannot be reconfigured



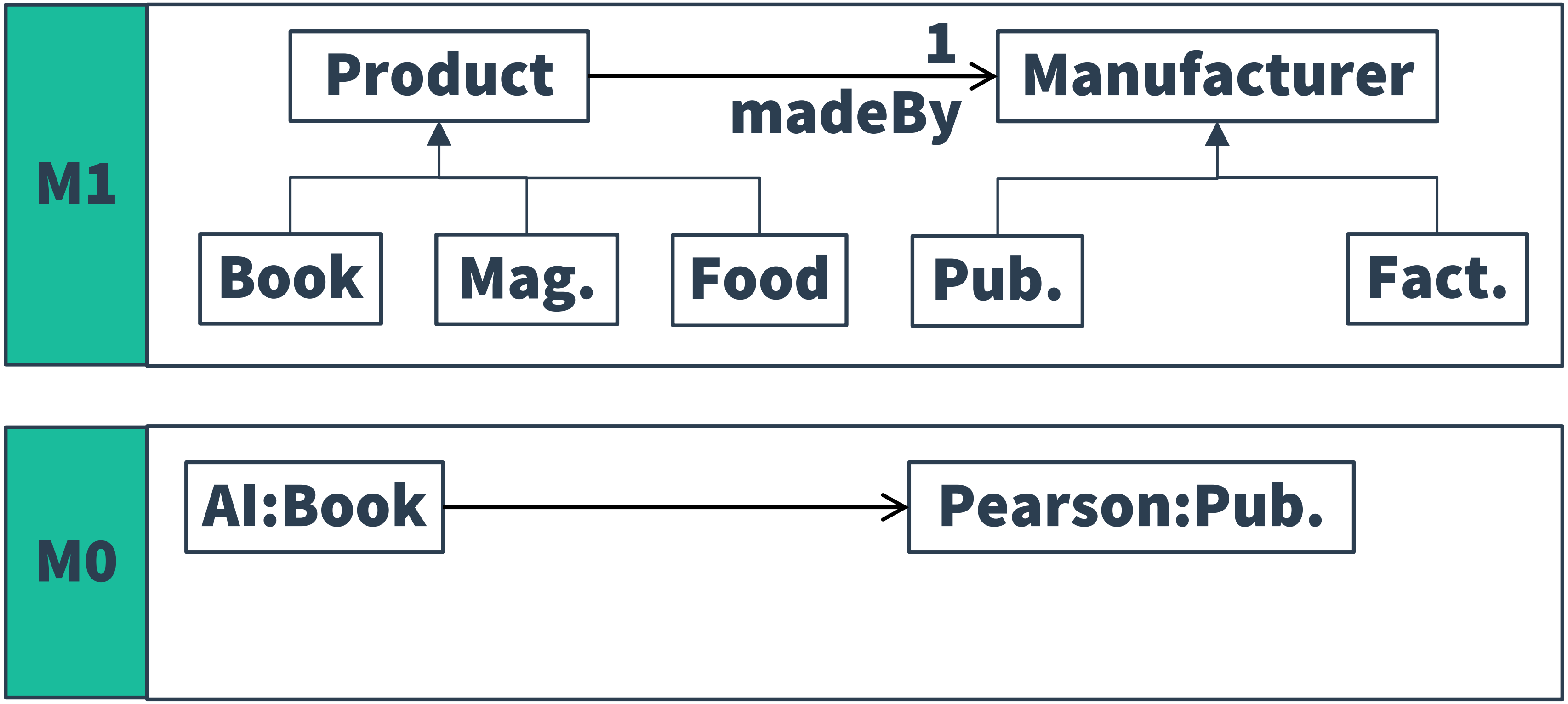
Limitations of Architecture (4)

- Relation Configuration
  - Reference types (i.e. relations) cannot be reconfigured



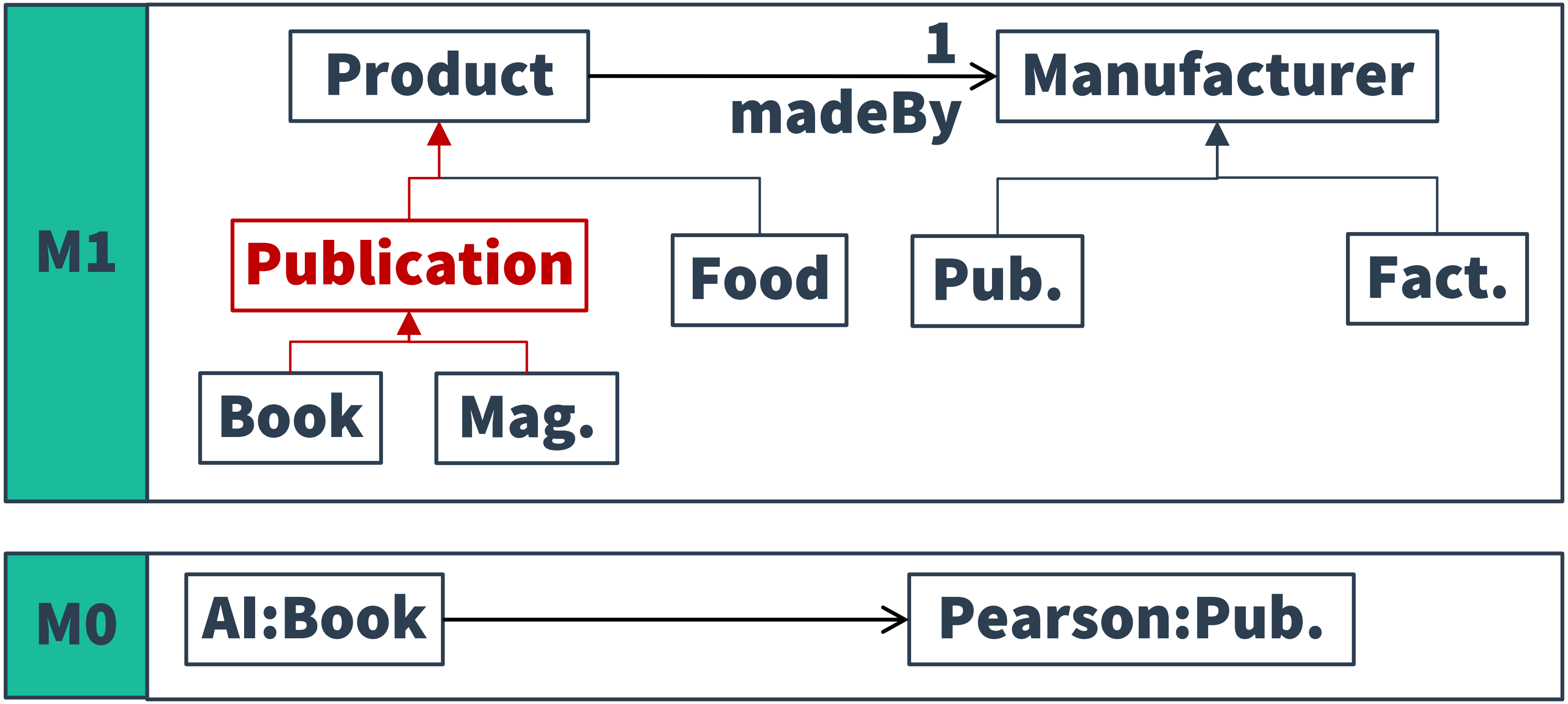
Limitations of Architecture (5)

- Element Classification
  - New classifications for (new) classes cannot be created



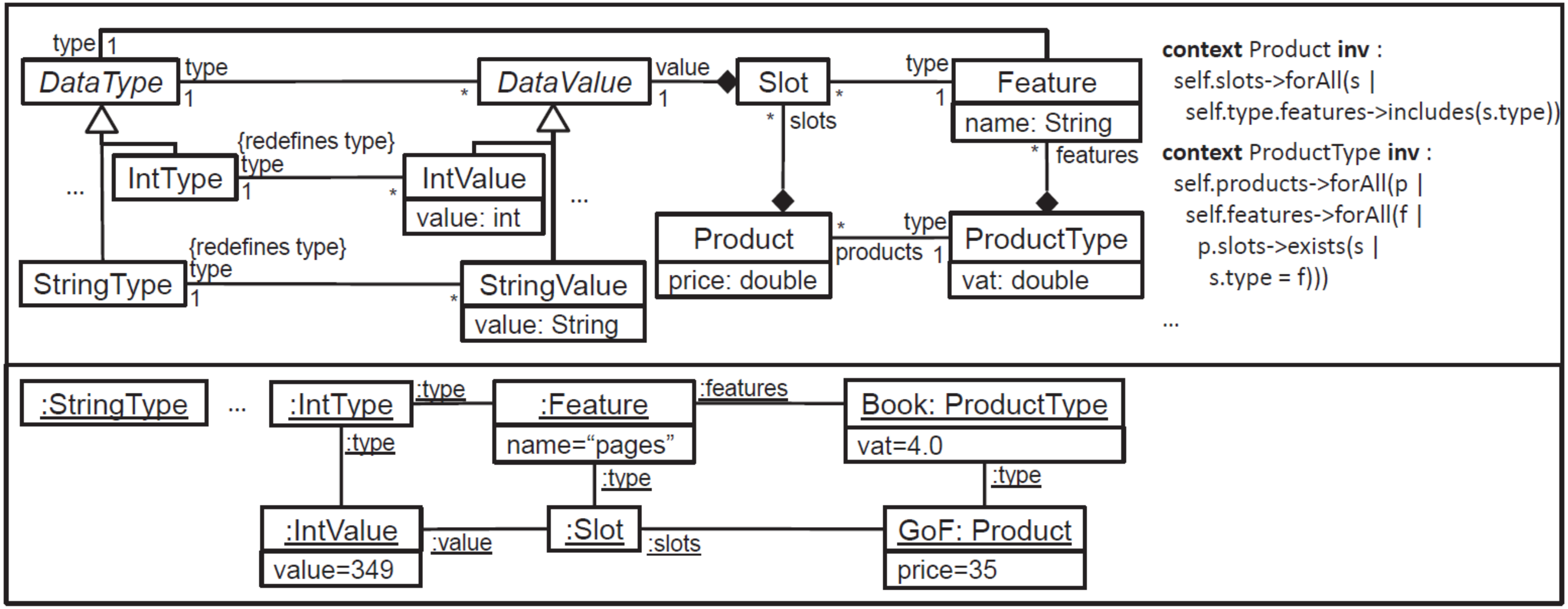
Limitations of Architecture (5)

- Element Classification
  - New classifications for (new) classes cannot be created



Two-Level Solutions (1)

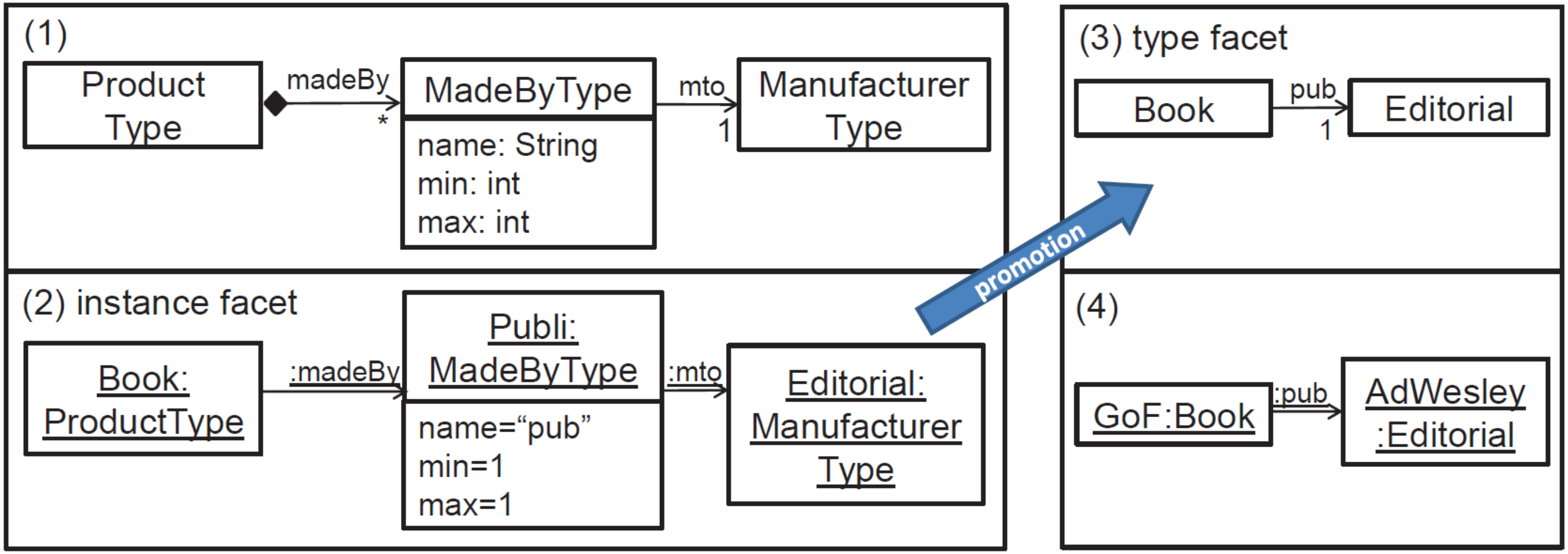
- **Explicit Modelling**
  - Model dynamic types, relations, etc. at the M0 level
  - Flexible, but yields complicated models



```
context Product inv :
  self.slots->forAll(s |
    self.type.features->includes(s.type))
context ProductType inv :
  self.products->forAll(p |
    self.features->forAll(f |
      p.slots->exists(s |
        s.type = f)))
...
```

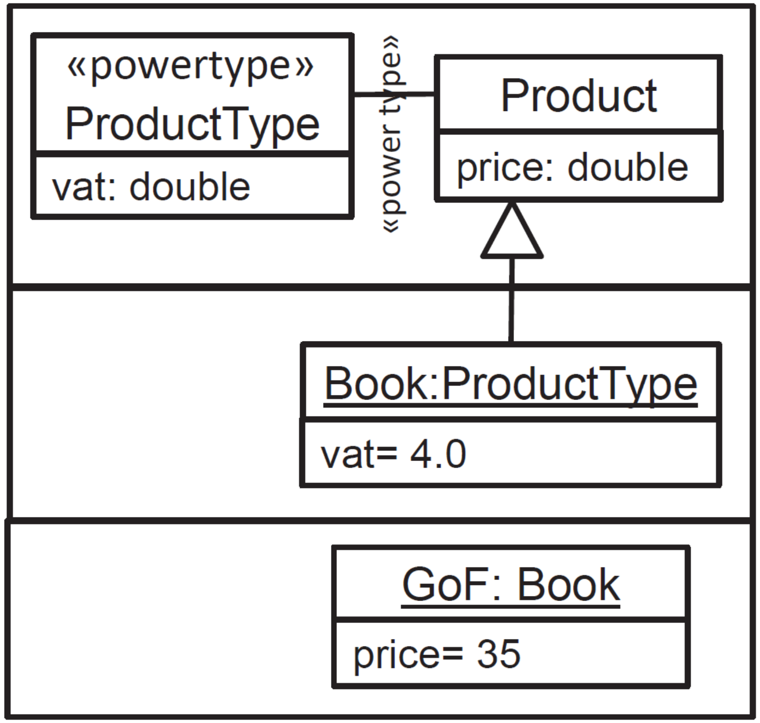
Two-Level Solutions (2)

- **Promotion**
  - Model types at M0 and transform into a meta-model
  - Flexible, but may require complex transformation



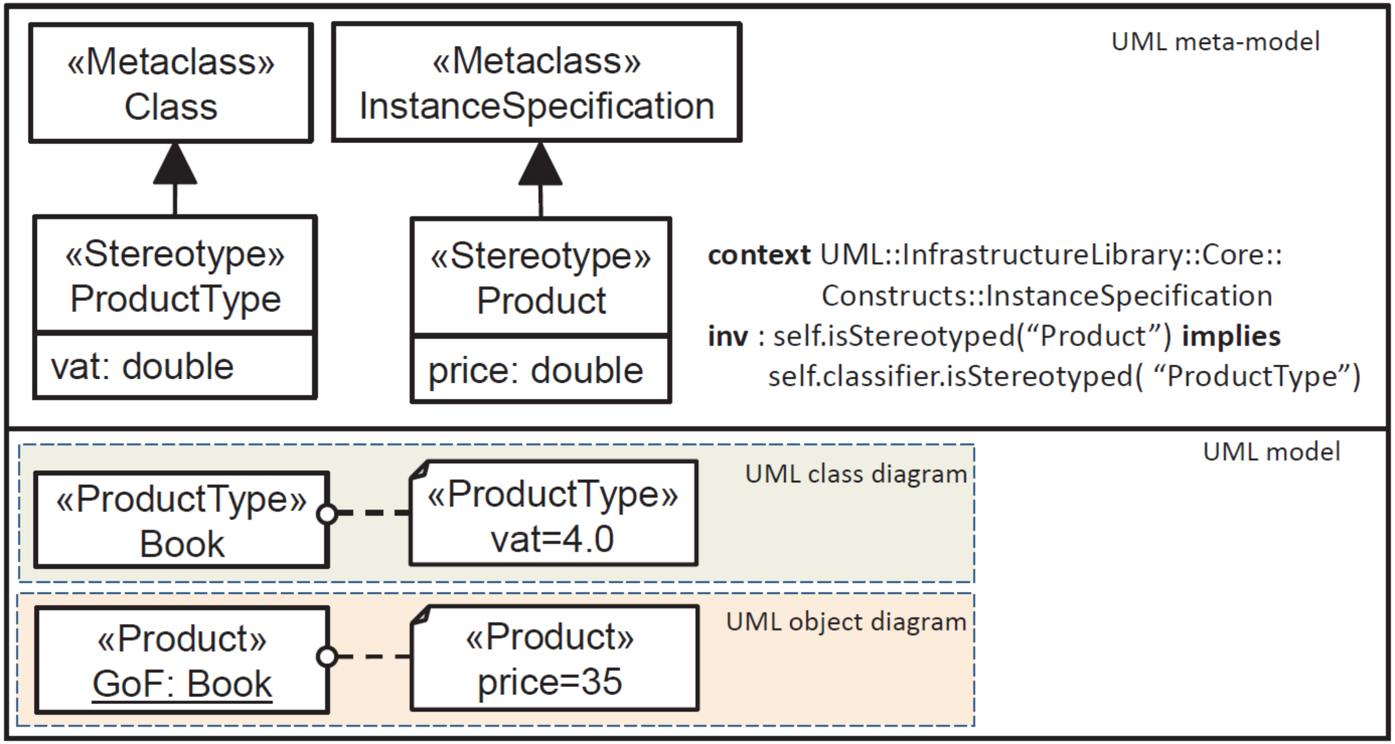
Two-Level Solutions (3)

- Powertypes
  - Types whose instances are subtypes of another type
  - Limited to modelling features in the next two levels



Two-Level Solutions (4)

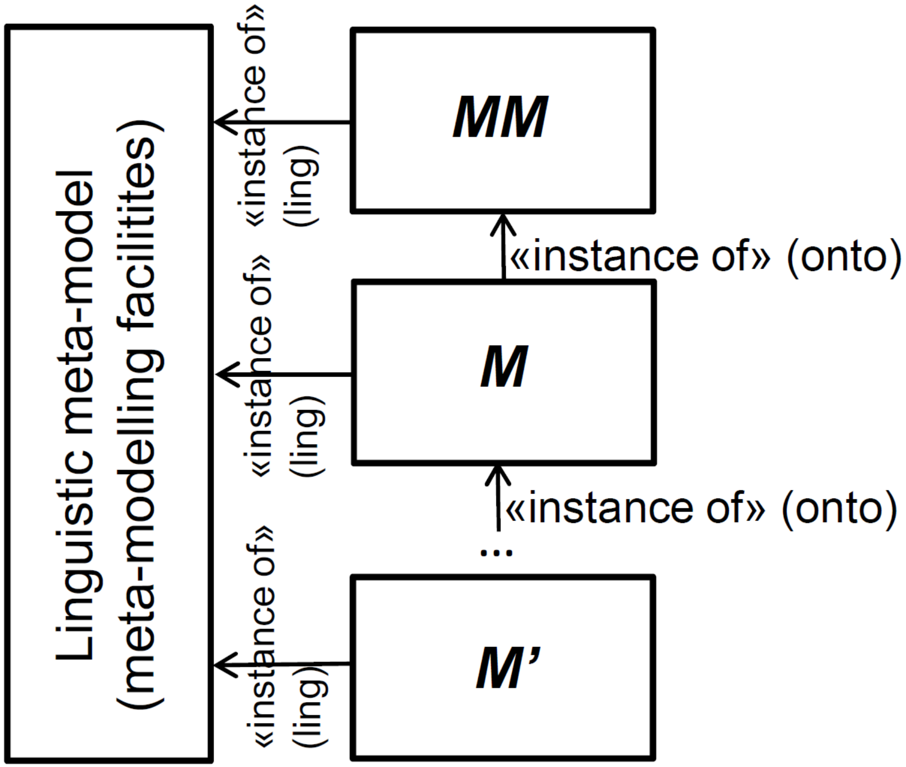
- Stereotypes
  - Extensions of the metametamodel
  - Limited to modelling domain concepts at two levels



```
context UML::InfrastructureLibrary::Core::
        Constructs::InstanceSpecification
inv : self.isStereotyped(“Product”) implies
      self.classifier.isStereotyped( “ProductType”)
```

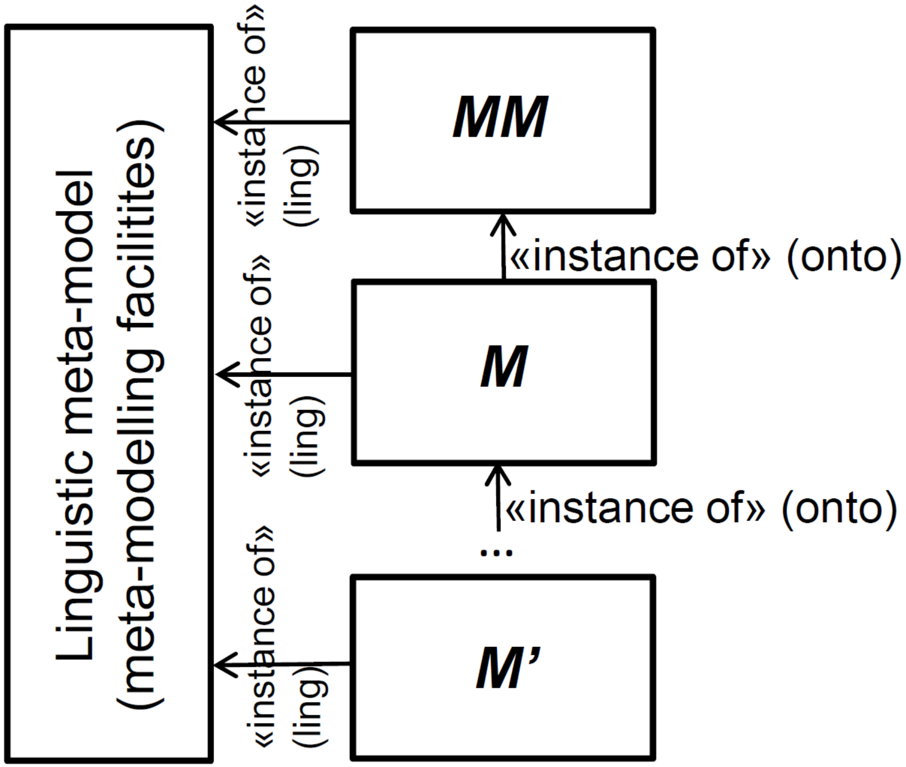
Multi-Level Modelling

- Potency
  - Model elements are clabjects
  - All elements are instantiable (for given number of times)



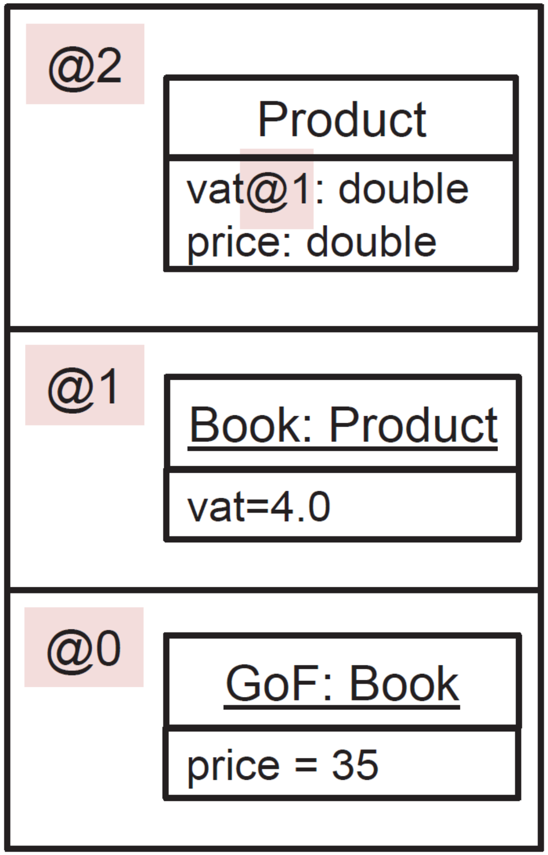
Multi-Level Modelling

- **Potency**
  - Model elements are clabjects
  - All elements are instantiable (for given number of times)
- **Orthogonal Classification Architecture**
  - Metamodelling facilities available at all meta-levels
  - Two kinds of types: ontological & linguistic



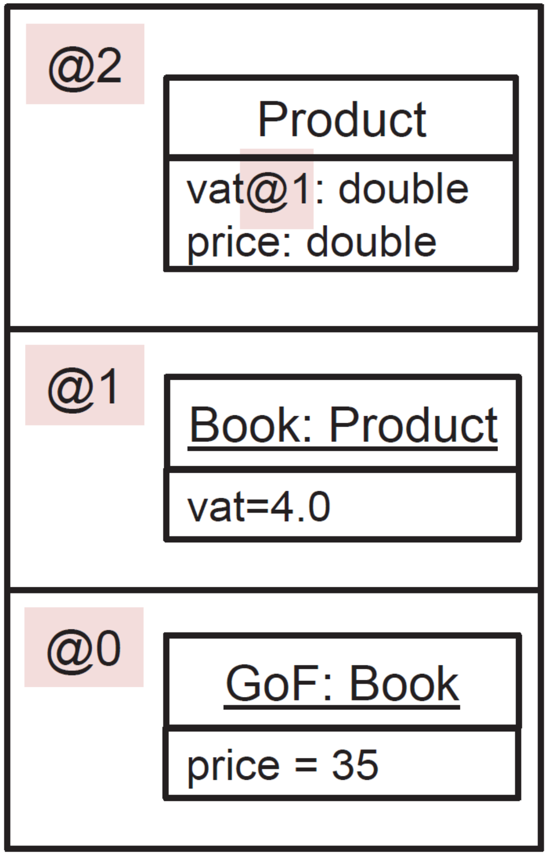
Multi-Level Modelling

- Type-Object

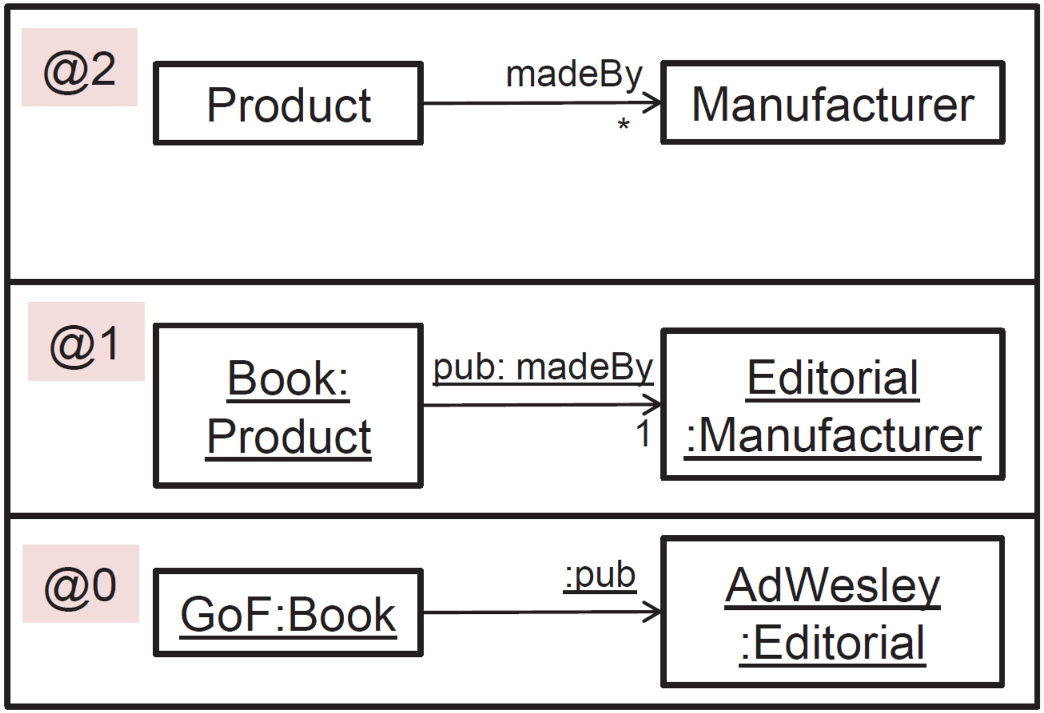


Multi-Level Modelling

- Type-Object



- Relation Configurator

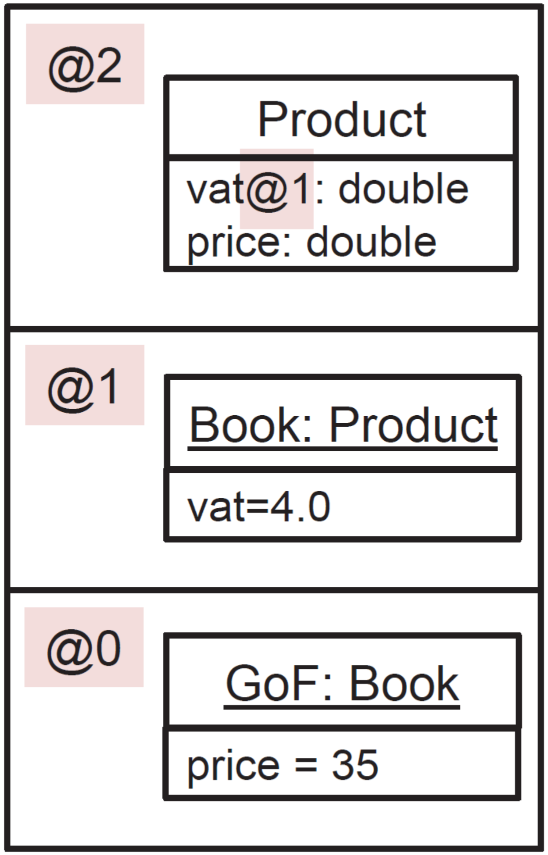


Field Study

- Results
  - Over 400 metamodels surveyed
  - 84 contain at least one type-object
  - 459 occurrences of patterns in total

Field Study

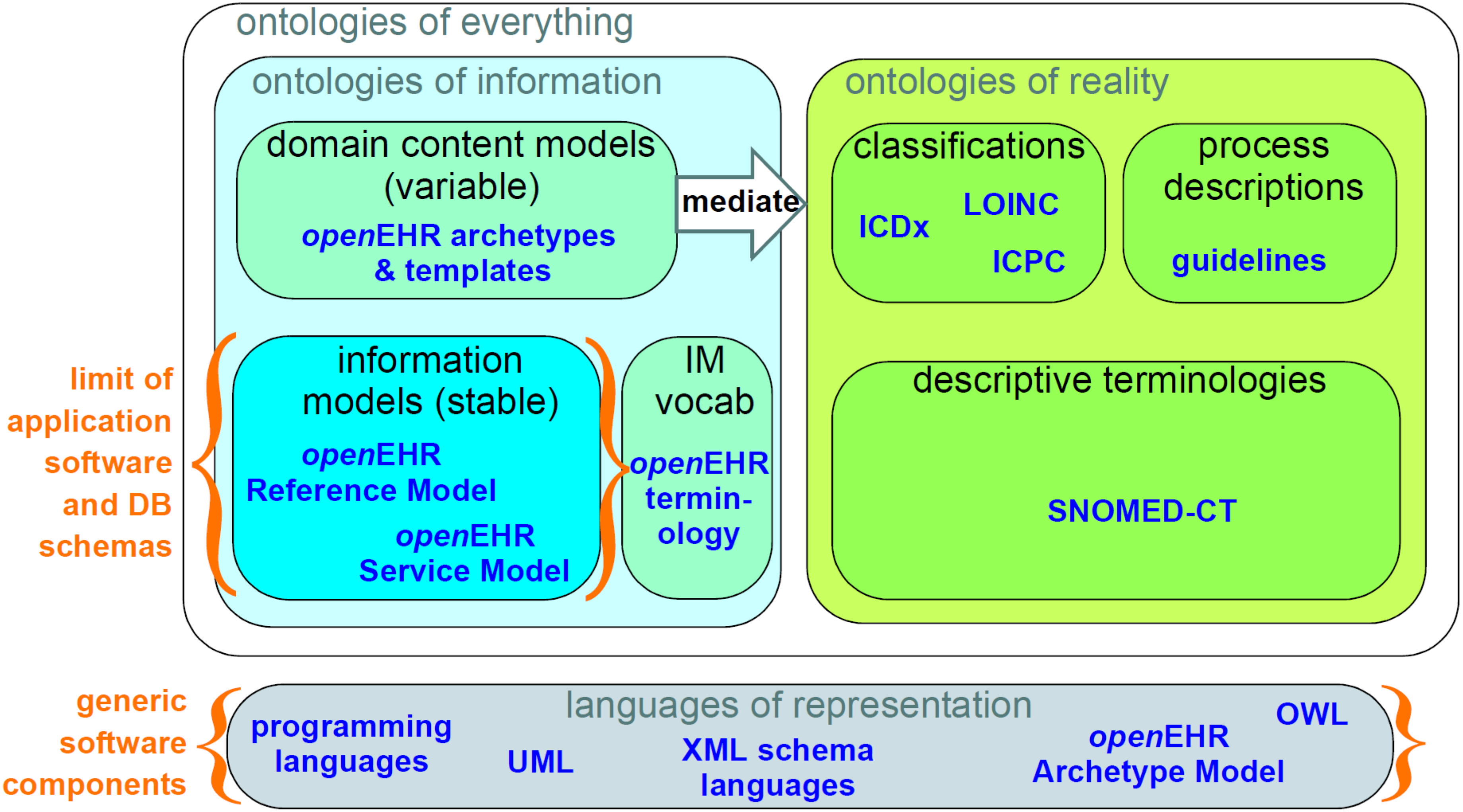
- **Results**
  - Over 400 metamodels surveyed
  - 84 contain at least one type-object
  - 459 occurrences of patterns in total
- **Discussion Points**
  - Which approach is more “natural”?
  - Is multi-level modelling a workaround?



Electronic Health Record (EHR)

- **Requirements**
  - Capture holistic view of patient
  - Applicable/extensible to all clinical domains
  - Applicable for all clinical environments

openEHR Architecture



limit of application software and DB schemas

ontologies of everything

ontologies of information

domain content models (variable)

*open*EHR archetypes & templates

mediate

information models (stable)

*open*EHR Reference Model

*open*EHR Service Model

IM vocab

*open*EHR termin-ology

ontologies of reality

classifications

ICDx LOINC ICPC

process descriptions

guidelines

descriptive terminologies

SNOMED-CT

generic software components

languages of representation

programming languages

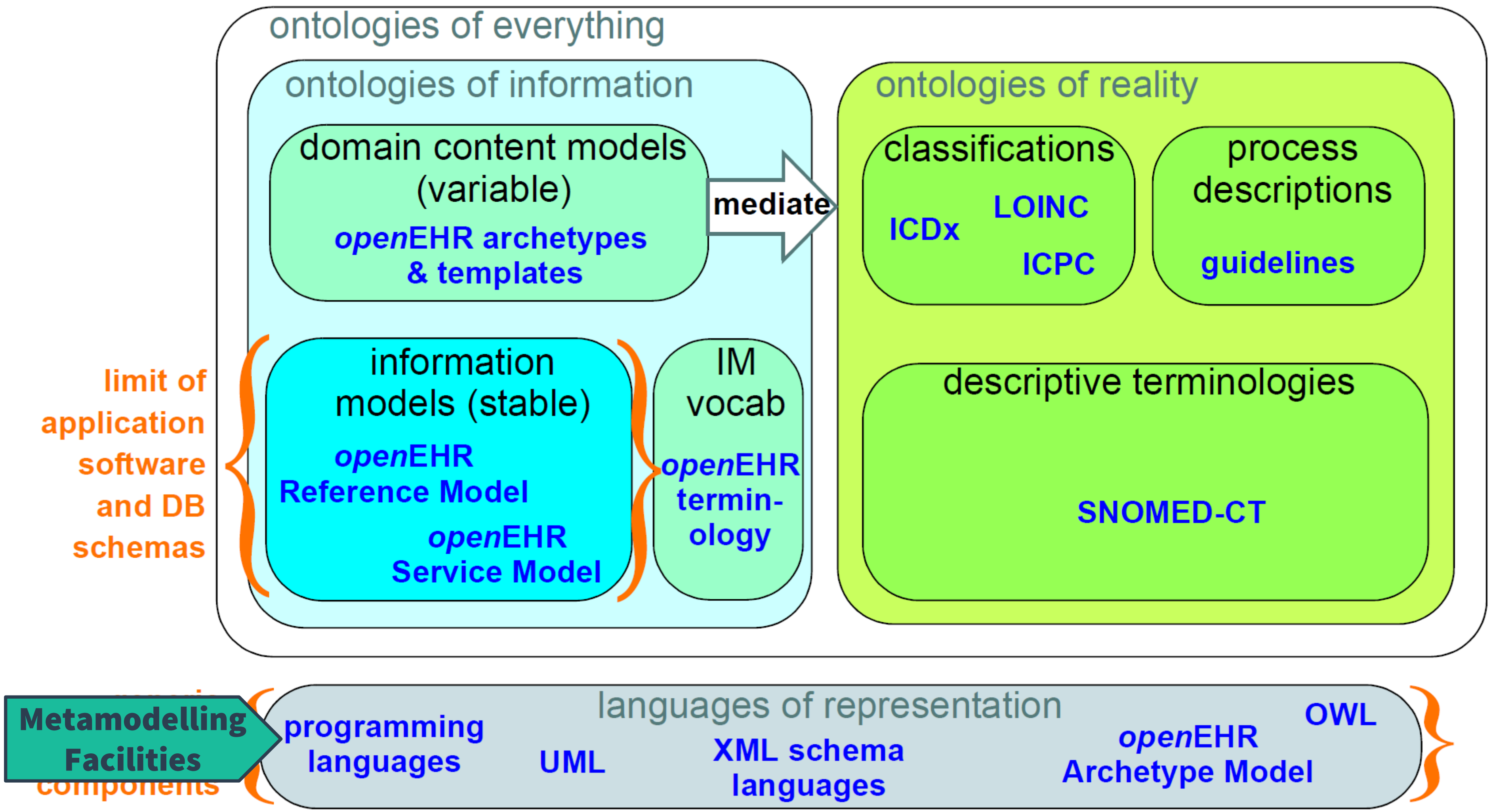
UML

XML schema languages

*open*EHR Archetype Model

OWL

openEHR Architecture



ontologies of everything

ontologies of information

domain content models (variable)

*open*EHR archetypes & templates

mediate

ontologies of reality

classifications

ICDx LOINC ICPC

process descriptions

guidelines

limit of application software and DB schemas

information models (stable)

*open*EHR Reference Model

*open*EHR Service Model

IM vocab

*open*EHR termin-ology

descriptive terminologies

SNOMED-CT

Metamodelling Facilities

components

languages of representation

programming languages

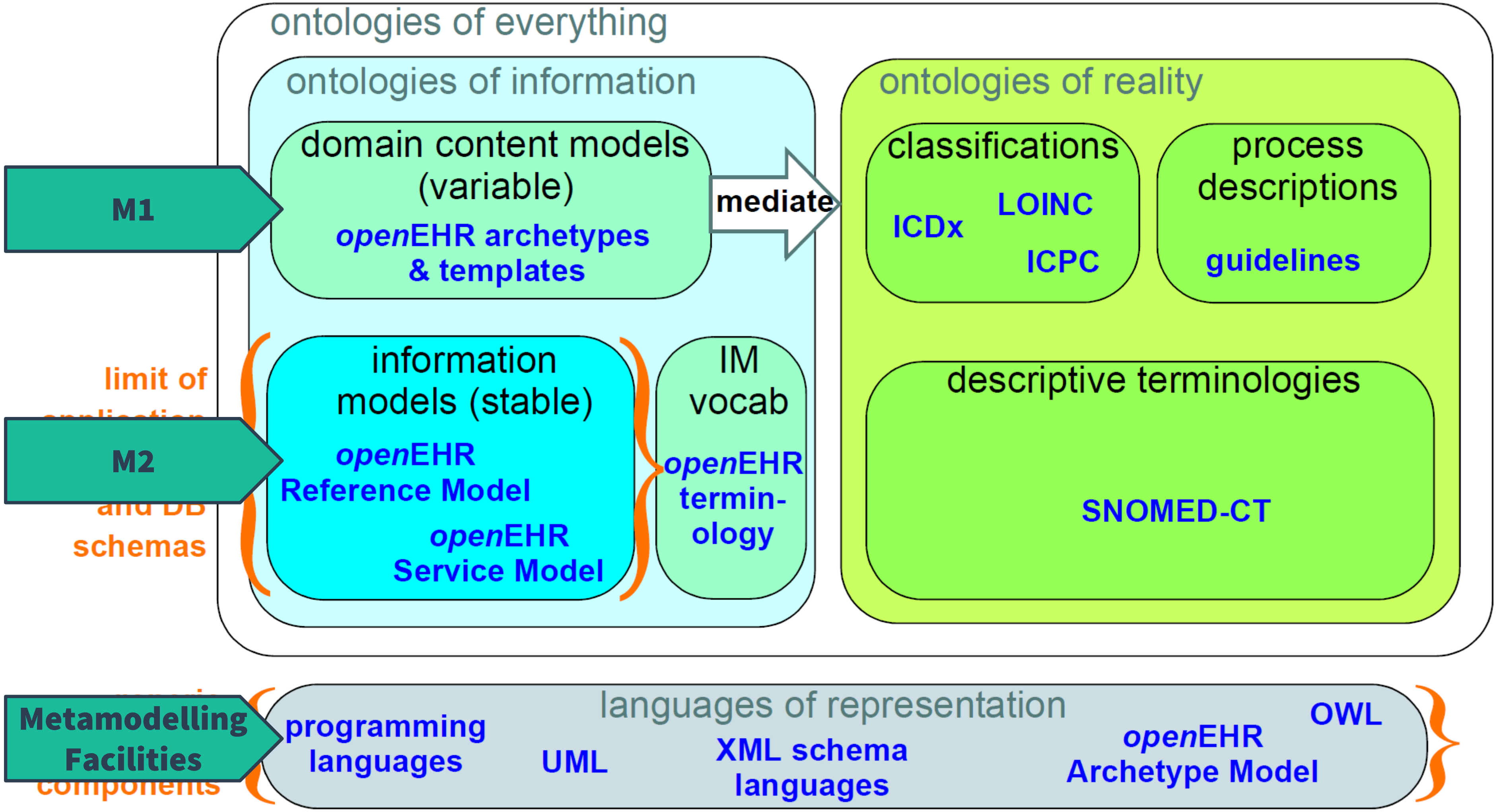
UML

XML schema languages

*open*EHR Archetype Model

OWL

openEHR Architecture



ontologies of everything

ontologies of information

domain content models (variable)

*open*EHR archetypes & templates

mediate

ontologies of reality

classifications

ICDx

LOINC

ICPC

process descriptions

guidelines

information models (stable)

*open*EHR Reference Model

*open*EHR Service Model

IM vocab

*open*EHR terminology

descriptive terminologies

SNOMED-CT

M1

M2

limit of application and DB schemas

Metamodelling Facilities

components

languages of representation

programming languages

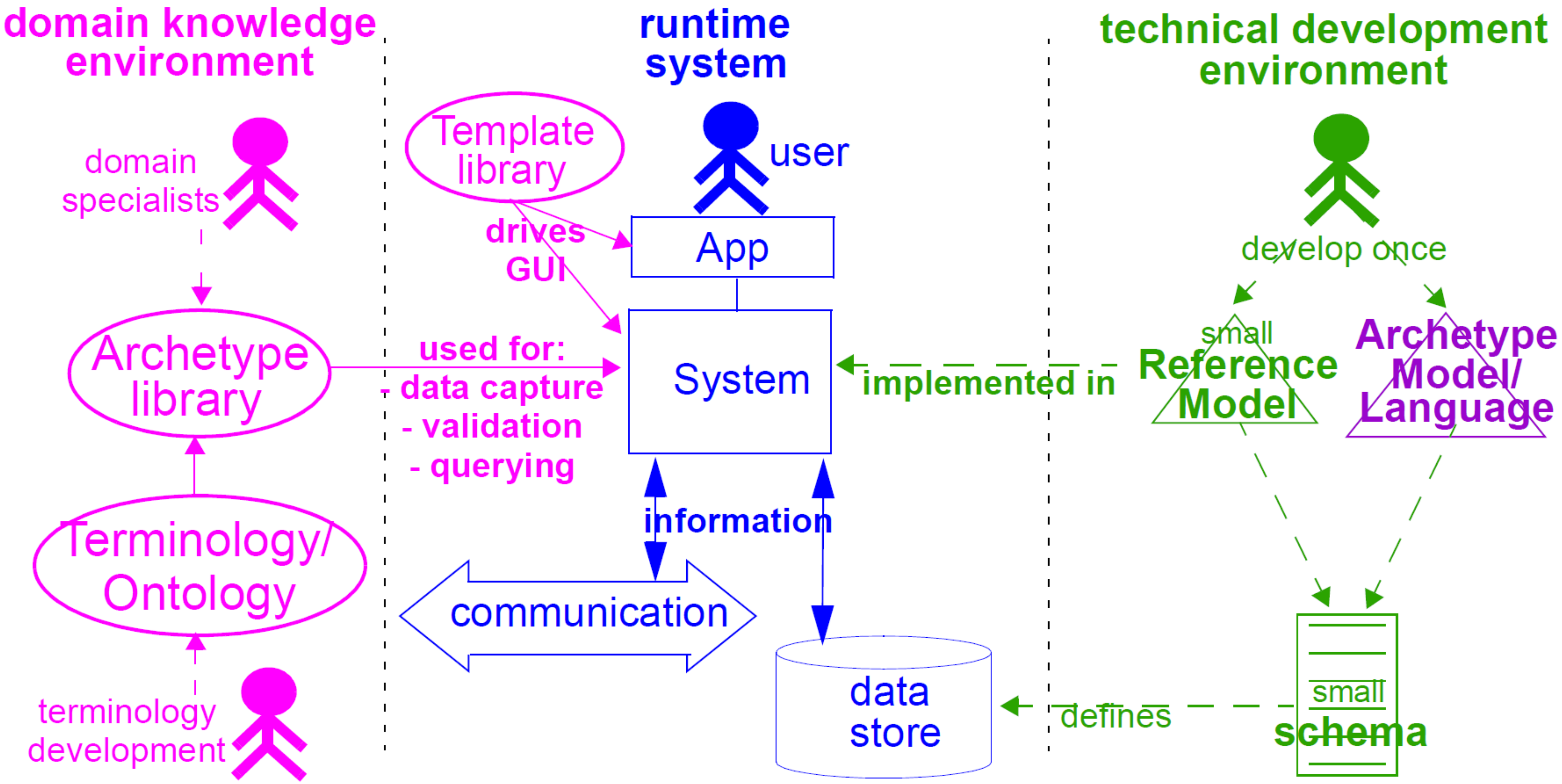
UML

XML schema languages

*open*EHR Archetype Model

OWL

openEHR Architecture



Conclusions

- **Multi-Level Modelling**
  - Potency
  - Orthogonal Classification Architecture
- **Design Patterns**
  - Type-Object Pattern
  - Dynamic Features
  - Dynamic Auxiliary Domain Concepts
  - Relation Configurator Pattern
  - Element Classification
- **Discussion**
  - When to use multi-level modelling?