Hochschule **RheinMain**

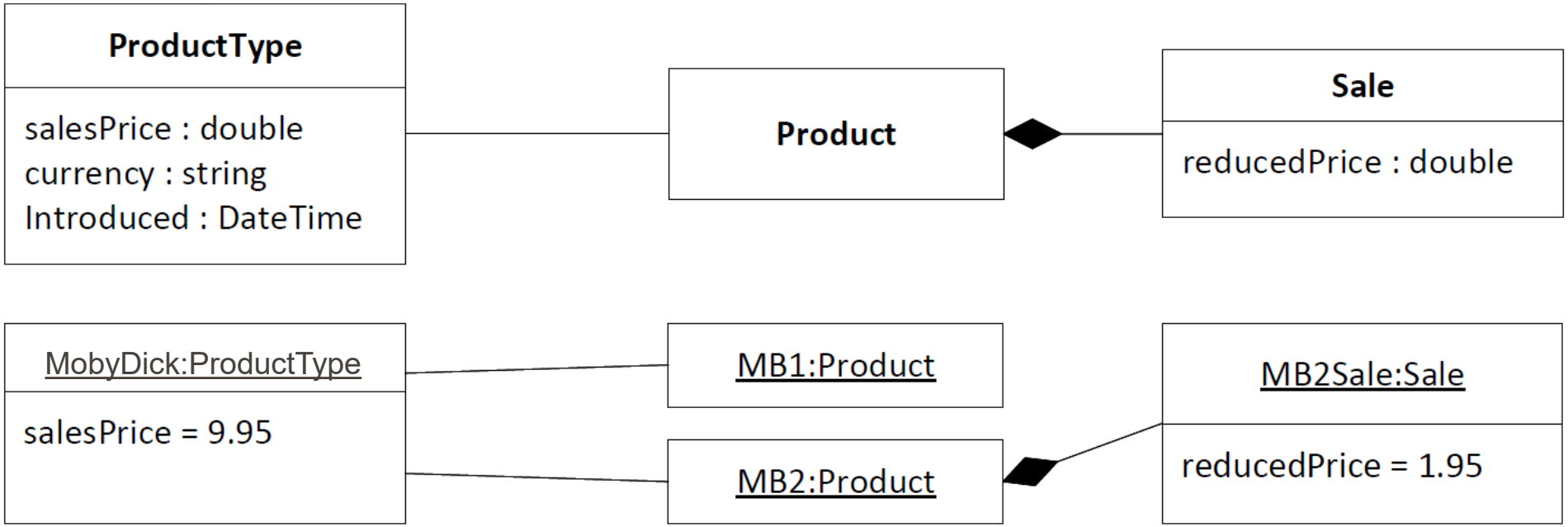
MODELING A WAREHOUSE SYSTEM USING REFINEMENTS AND DECOMPOSITION

A contribution to the MULTI Warehouse challenge

Georg Hinkel
24.09.2024

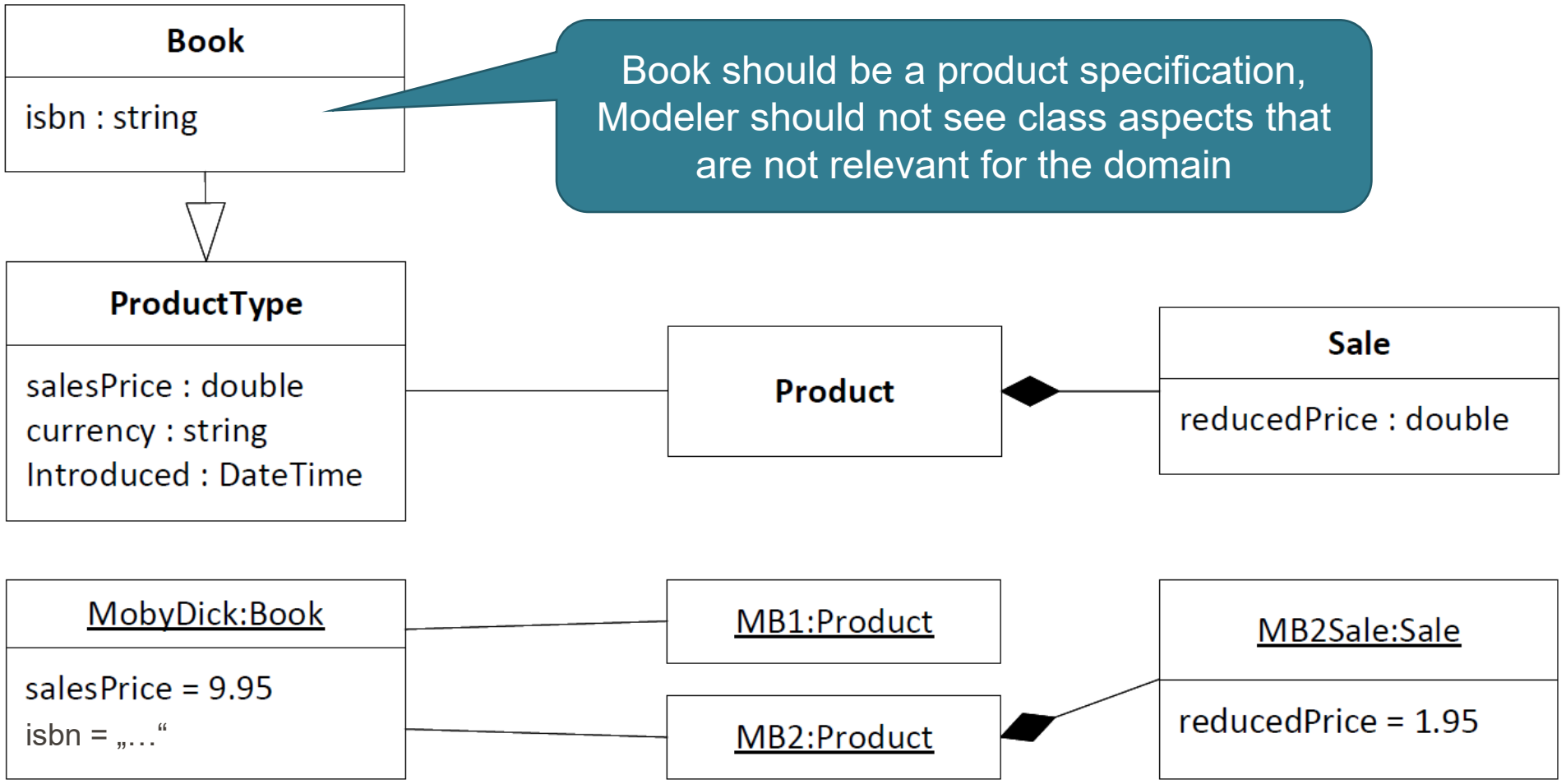
A WAREHOUSE-MODEL IN TWO-LEVEL MODELING

Hochschule **RheinMain**



A WAREHOUSE-MODEL IN TWO-LEVEL MODELING

Hochschule **RheinMain**



REFINEMENTS AND DECOMPOSITION

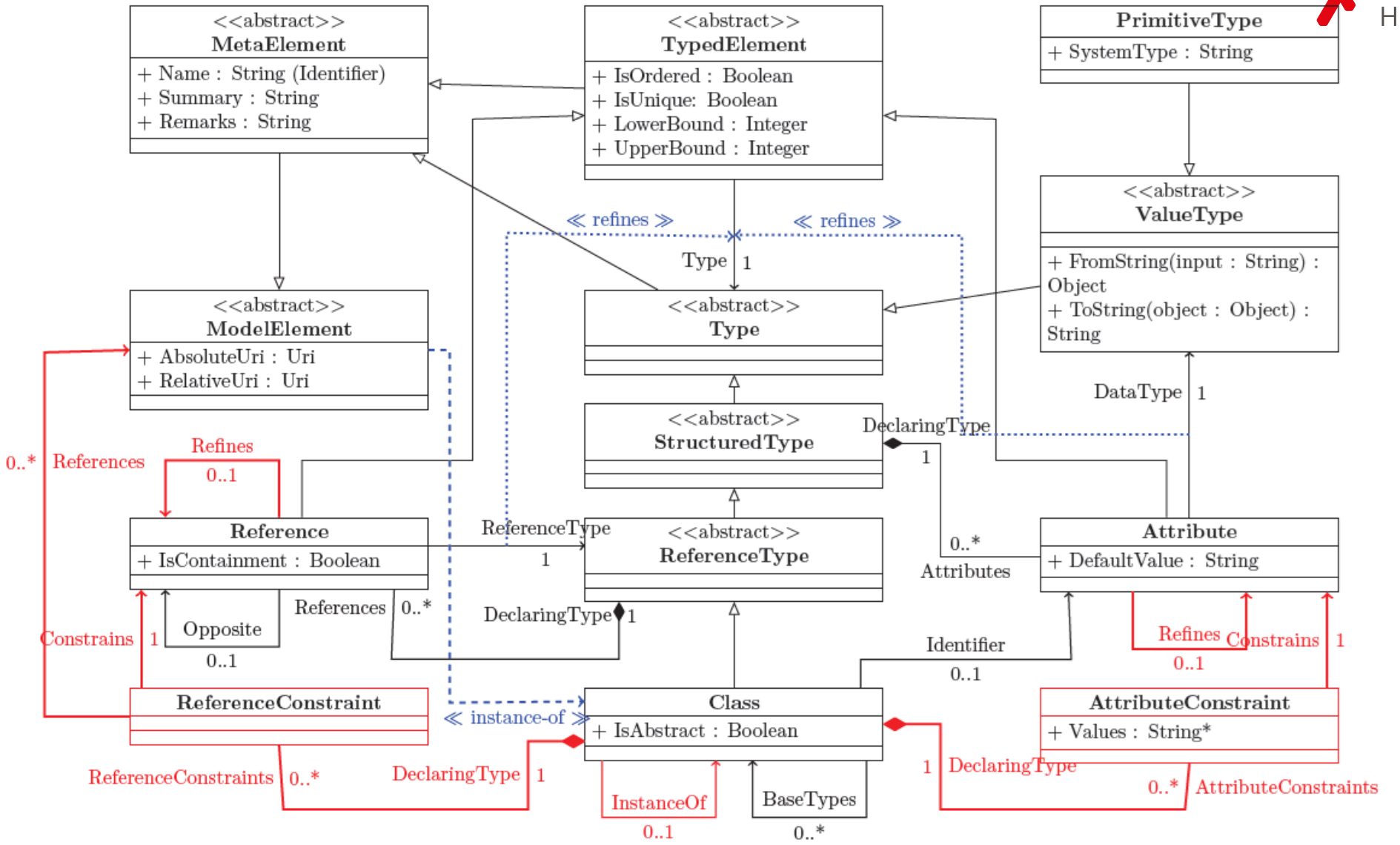
Hochschule **RheinMain**

- Closest concept: Redefinition in UML or MOF (but not defined properly)
- Refinements and decomposition formally defined and implemented [1]
  - $f: A \to B^*$ is decomposed into $f_1, \ldots, f_n: A \to B^*: \forall a \in A: f \nearrow (a) = f_1 \nearrow (a); \ldots; f_n \nearrow (a)$
  - $\hat{f}: \hat{A} \to \widehat{B}$ is refinement of $f: A \to B: \forall a \in \hat{A}, b \in \widehat{B}: \hat{f} \nearrow (a) = f \nearrow (a), \hat{f} \searrow (a, b) = f \searrow (a, b)$
  - Idea: generated code only stores refined feature / feature component via backing field
- Refinements and decomposition sufficient to realize deep modeling [2]
  - Minor extension to two-level metametamodel
  - Helpful definition: A categorizes B: $a \in A \Rightarrow a \prec B$ (an instance of A is a subtype of B)

[1] Georg Hinkel, Kiana Busch, and Robert Heinrich. 2018. **Refinements and Structural Decompositions in Generated Code.** In *Proceedings of the 6th International Conference on Model-Driven Engineering and Software Development, MODELSWARD 2018*, Funchal, Madeira - Portugal, January 22-24, 2018, Slimane Hammoudi, Luís Ferreira Pires, and Bran Selic (Eds.). SciTePress, 303–310. https://doi.org/10.5220/0006549403030310
[2] Georg Hinkel. 2019. **Using structural decomposition and refinements for deep modeling of software architectures.** *Softw. Syst. Model.* 18, 5, 2787–2819. https://doi.org/10.1007/S10270-018-0701-6

NMETA

Hochschule **RheinMain**



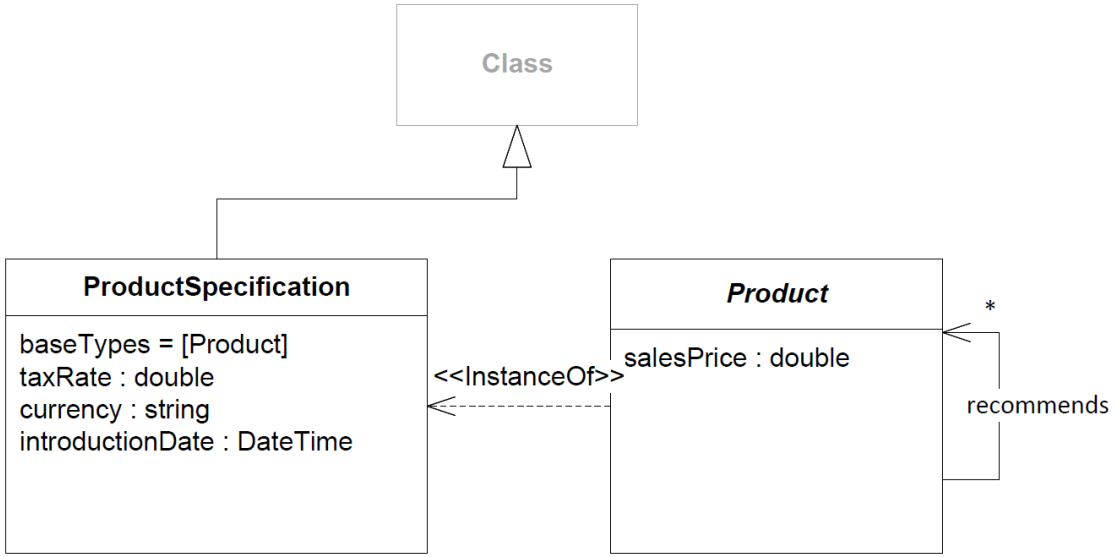
A WAREHOUSE MODEL V1

Hochschule **RheinMain**

- A product specification can have instances
  - Properties of a product are determined by the product specification
    ➔ is a class (inherits `Class`)

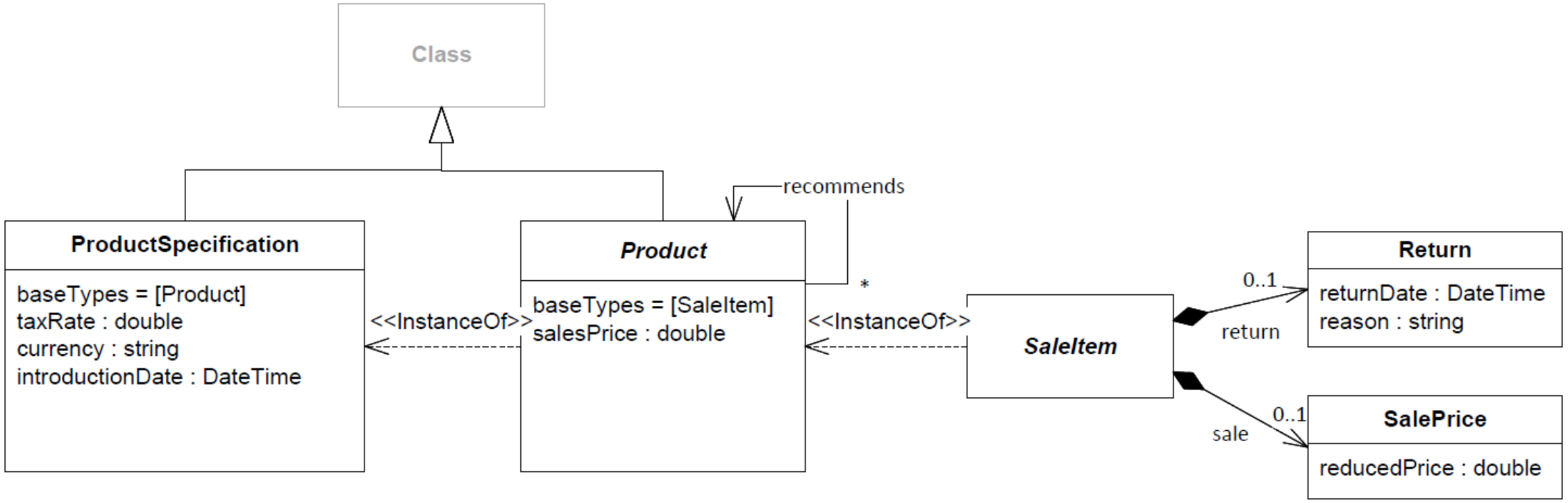
- A product is always an instance of a product specification
  - `ProductSpecification` categorizes `Product`
    ➔ <<instanceOf>>

```
1  public interface IProduct
2  {
3    ...
4    /// <summary>
5    /// Gets the ProductSpecification for this model element
6    /// </summary>
7    IProductSpecification GetProductSpecification();
8  }
```



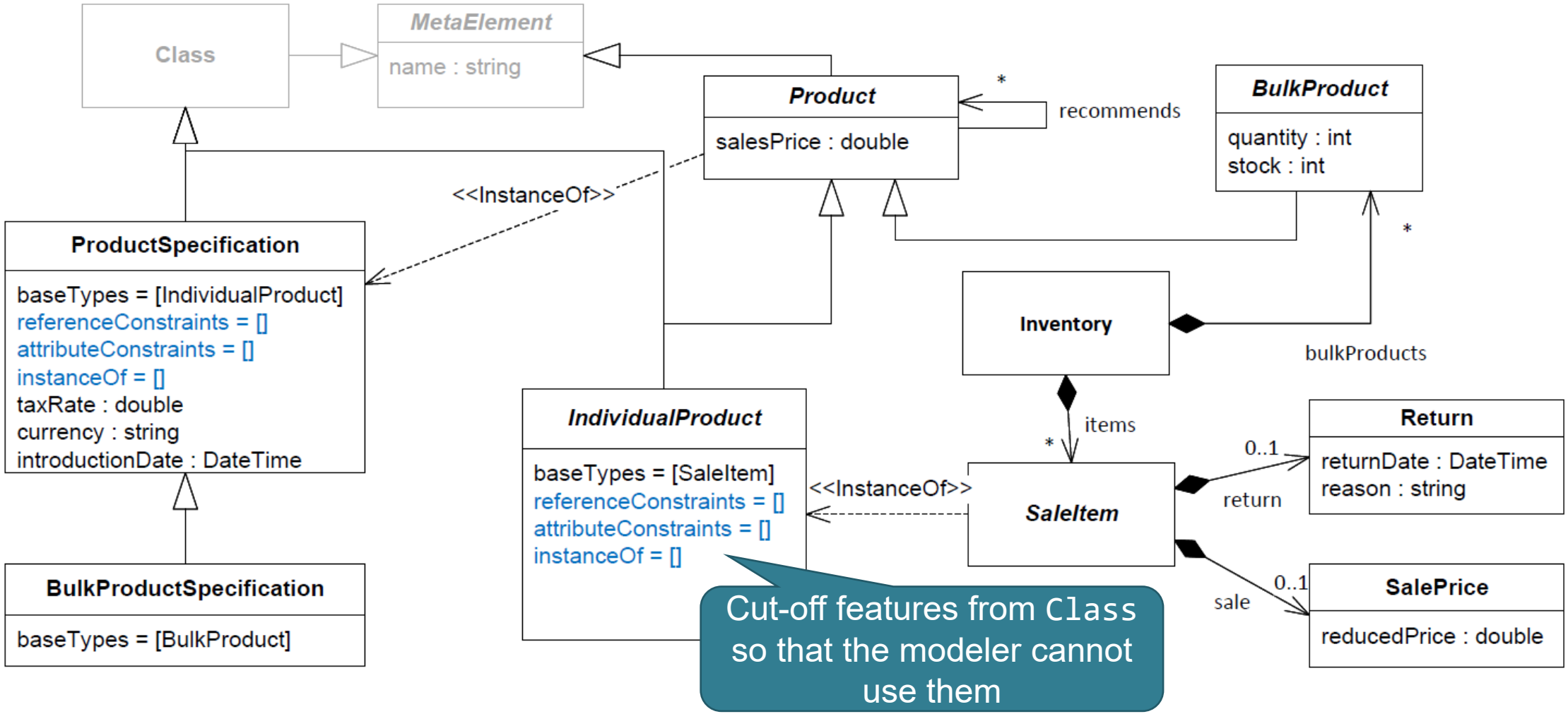
A WAREHOUSE MODEL V2

Hochschule **RheinMain**



A WAREHOUSE MODEL V3

Hochschule RheinMain



CALCULATING THE SUMMED SALES PRICES FOR ALL ITEMS IN THE INVENTORY

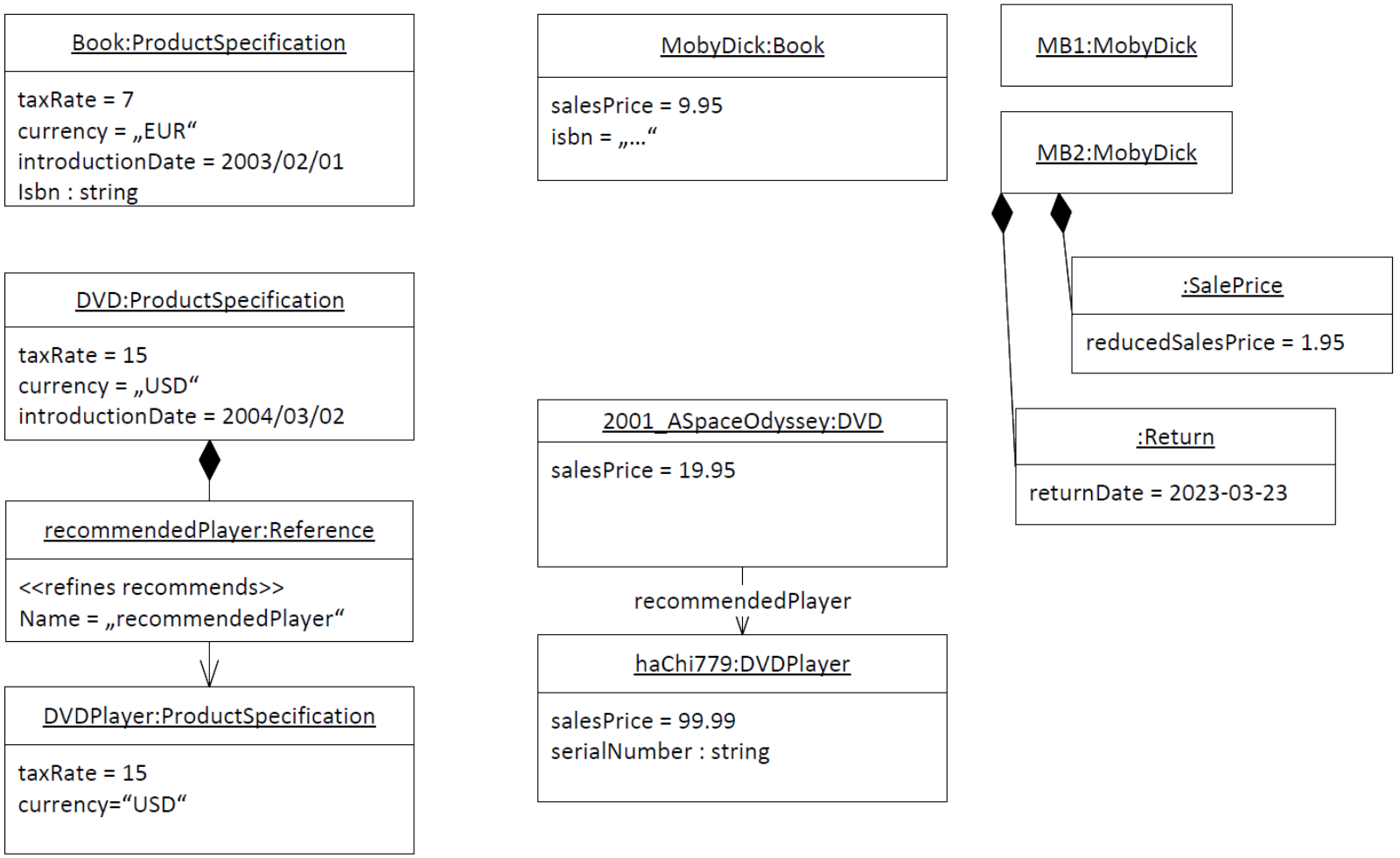
Hochschule RheinMain

```
double EffectiveSalesPrice(ISaleItem saleItem)
    => saleItem.Sale?.ReducedPrice ?? saleItem.GetIndividualProduct().SalesPrice;

var pricesPerProduct = from item in inventory.Items
                       group item by item.GetIndividualProduct() into itemsOfProduct
                       select new
                       {
                           Product = itemsOfProduct.Key,
                           SummedSalesPrices = itemsOfProduct.Sum(EffectiveSalesPrice)
                       };

foreach (var prod in pricesPerProduct)
{
    Console.WriteLine($"Total for {prod.Product}: {prod.Summe
{prod.Product.GetProductSpecification().Currency}");
}
```

Standard C# code, understood by millions of developers

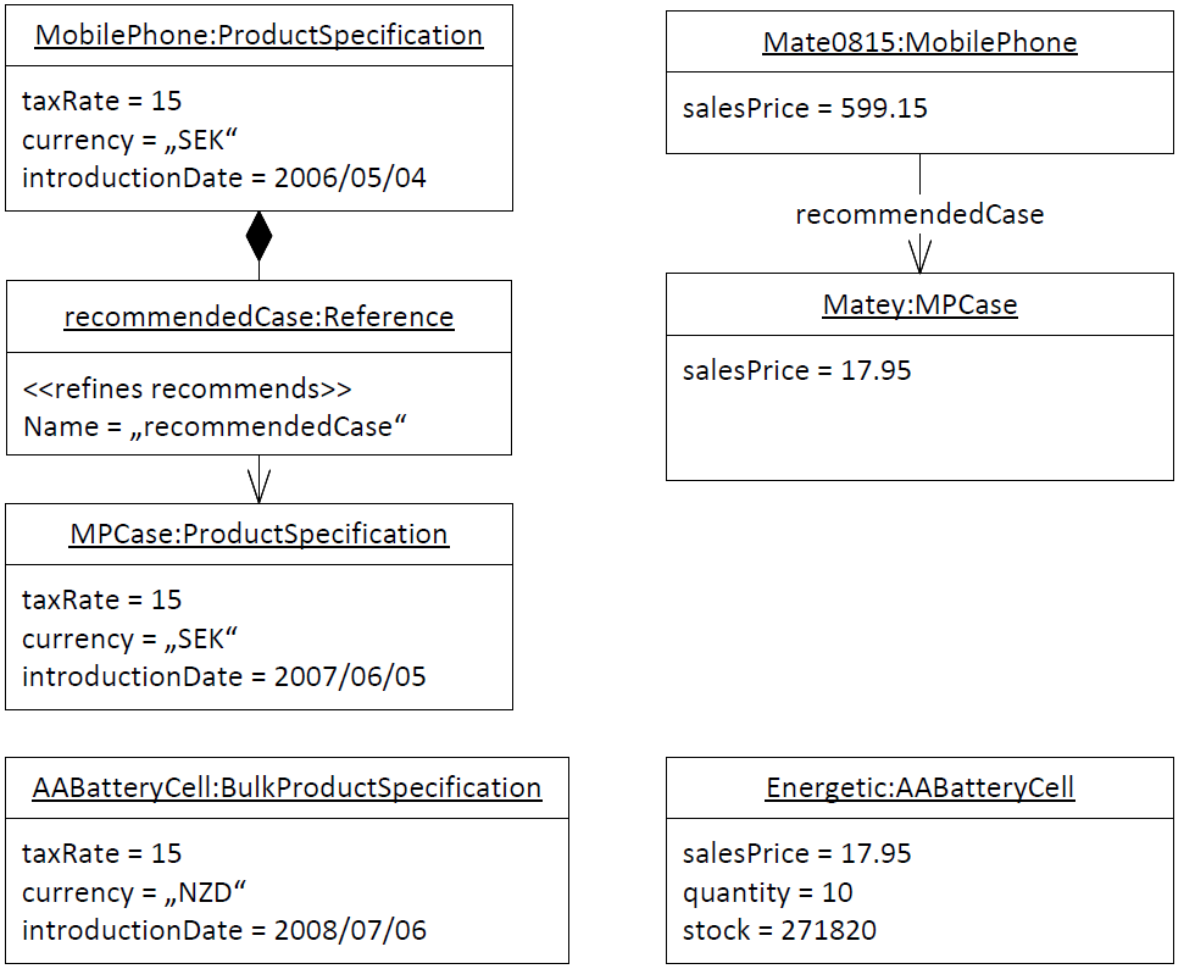
THE EXAMPLE WEB SHOP I

Hochschule **RheinMain**



THE EXAMPLE WEB SHOP II

Hochschule **RheinMain**



LIMITATIONS

Hochschule **RheinMain**

- No polymorphism for value semantics
  - Example: prices should by type-safe
- Instances in NMF cannot override slots of their classes
  - Example: AA batteries to be sold in NZD
- Instantiating a type currently requires code generation
  - Dynamic models exist in NMF, but currently do not support implementing static interfaces
- Currently no end-user-friendly modeling tool
  - Working on collaborative, graphical editors based on GLSP
- Currently no support for constraints
  - Planned to integrate into editor rather than into the model

CONCLUSIONS

Hochschule **RheinMain**

- The solution uses few modeling constructs on top of two-level modeling
  - Refinements
  - Decomposition
  - Categorization
- No notion of levels or potencies
- Few constraints necessary

REFERENCES

Hochschule **RheinMain**

1. Georg Hinkel, Kiana Busch, and Robert Heinrich. 2018. **Refinements and Structural Decompositions in Generated Code.** In *Proceedings of the 6th International Conference on Model-Driven Engineering and Software Development, MODELSWARD 2018*, Funchal, Madeira - Portugal, January 22-24, 2018, Slimane Hammoudi, Luís Ferreira Pires, and Bran Selic (Eds.). SciTePress, 303–310. https://doi.org/10.5220/0006549403030310
2. Georg Hinkel. 2019. **Using structural decomposition and refinements for deep modeling of software architectures.** *Softw. Syst. Model.* 18, 5, 2787–2819. https://doi.org/10.1007/S10270-018-0701-6