

MODELLEZÉS

pragmatikus szemléletben

THEISZ Zoltán, software architect, Vehicle Motion Control

engineering.tomorrow.together.

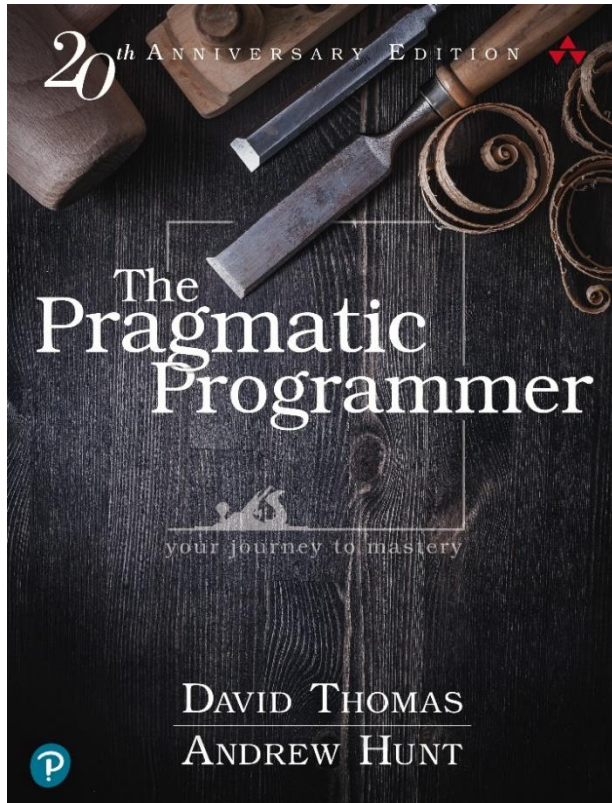


thyssenkrupp

pragmatic: "relating to matters of fact or practical affairs often to the exclusion of intellectual or artistic matters : practical as opposed to idealistic"



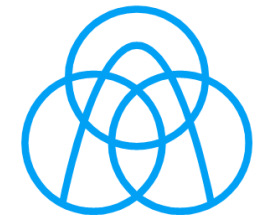
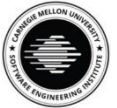
The Pragmatic Bookshelf



software architect: "software engineer responsible for high-level design choices related to overall system structure and behavior"

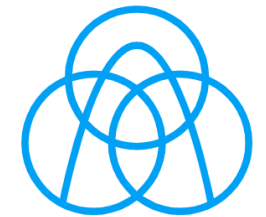
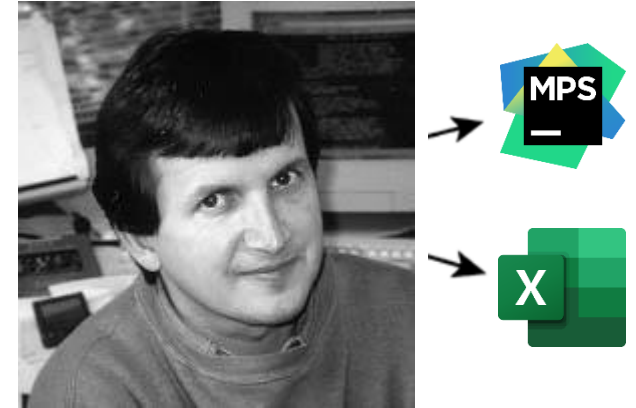
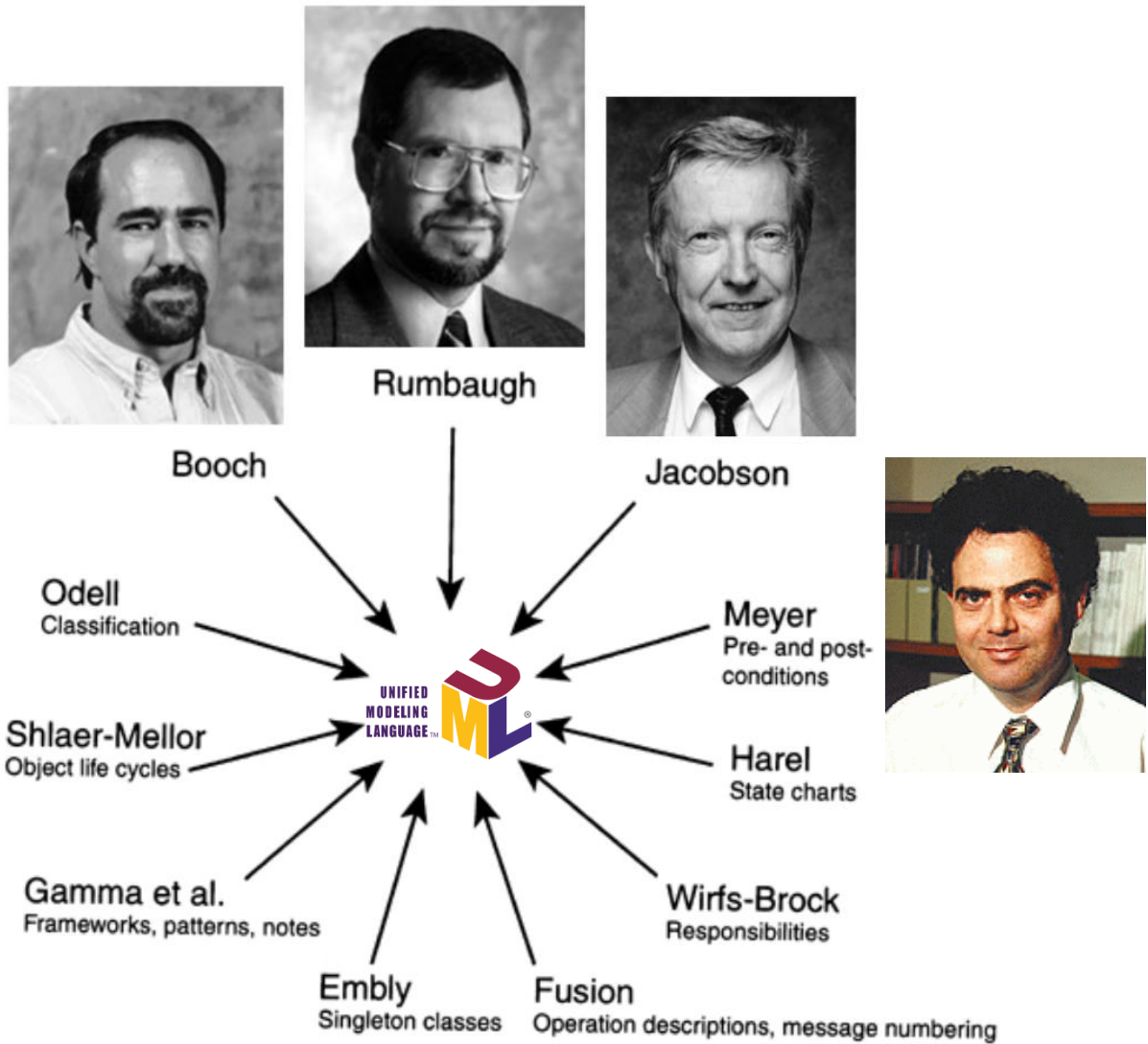


model-based systems engineering: "formalized methodology that is used to support the requirements, design, analysis, verification, and validation associated with the development of complex systems"



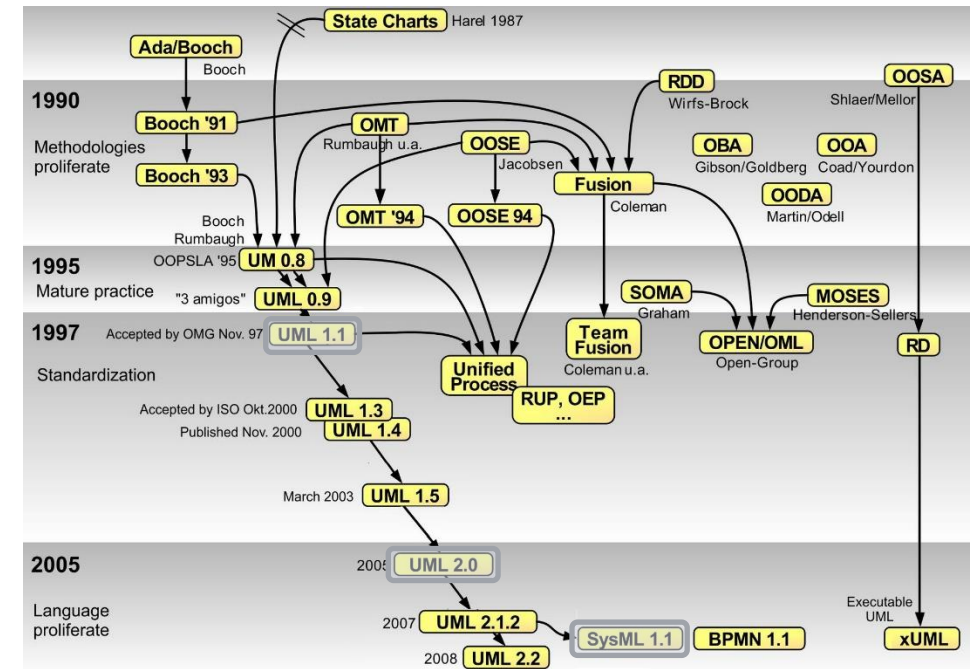
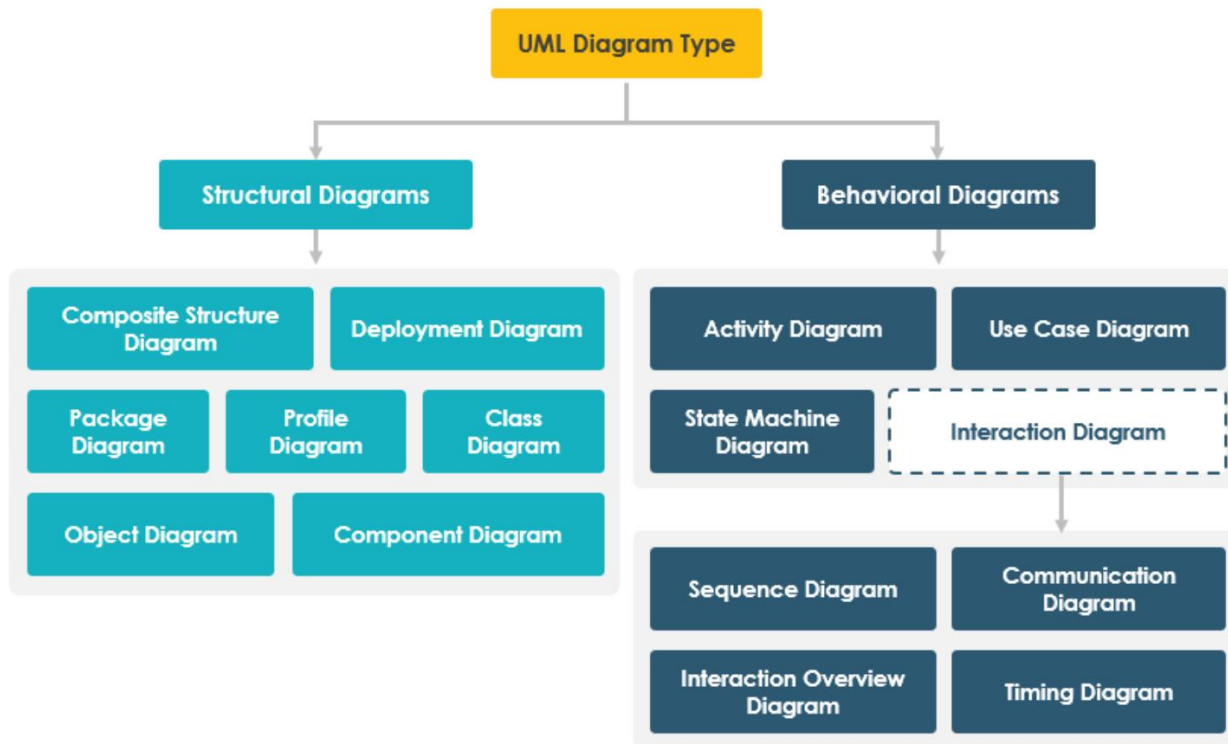
thyssenkrupp

engineering.tomorrow.together.



engineering.tomorrow.together.

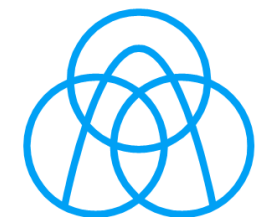
thyssenkrupp



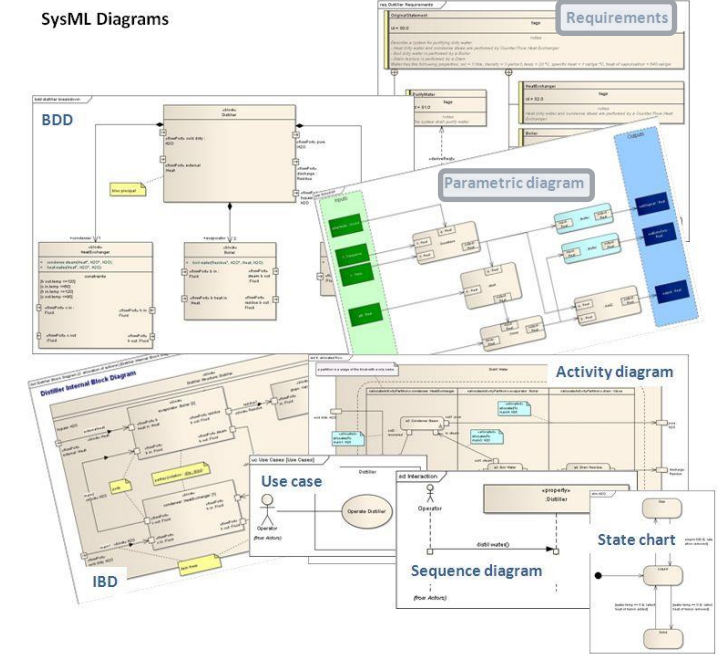
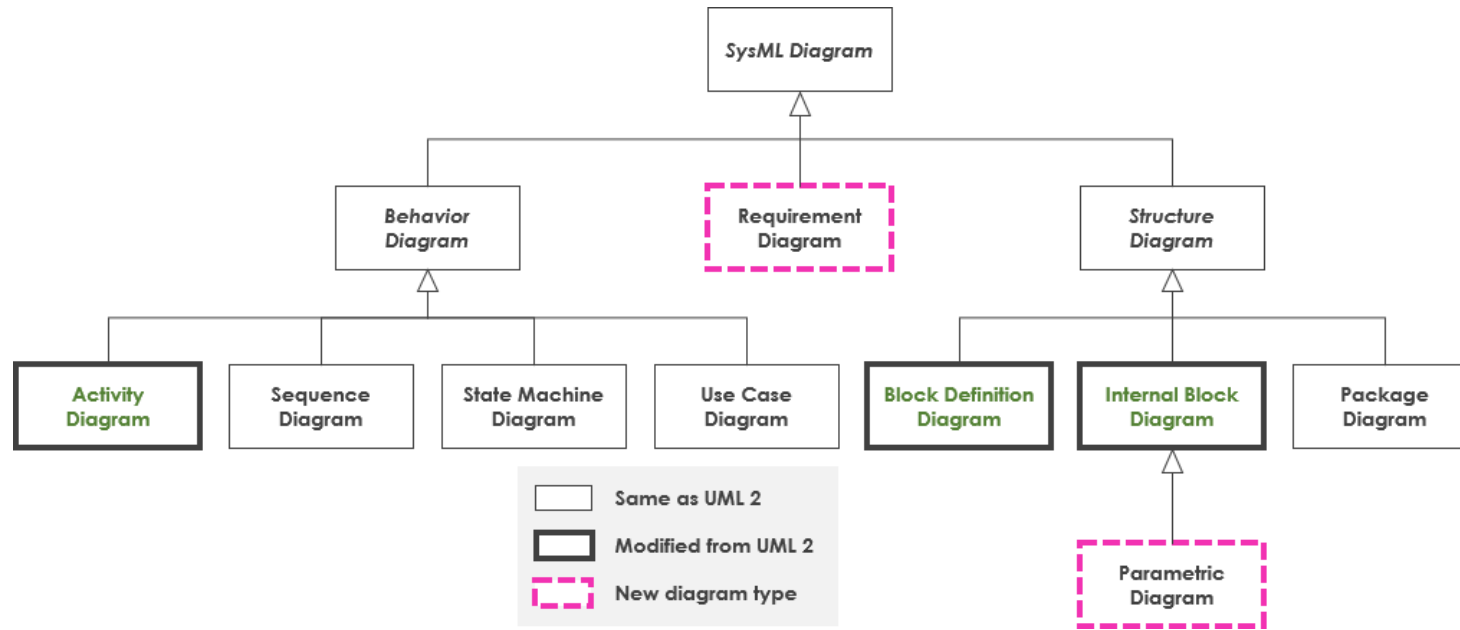
software architect: "software engineer responsible for high-level design choices related to overall system structure and behavior"



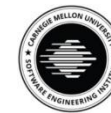
engineering.tomorrow.together.



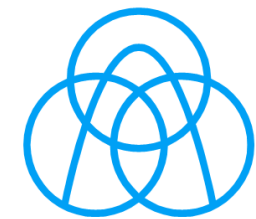
thyssenkrupp



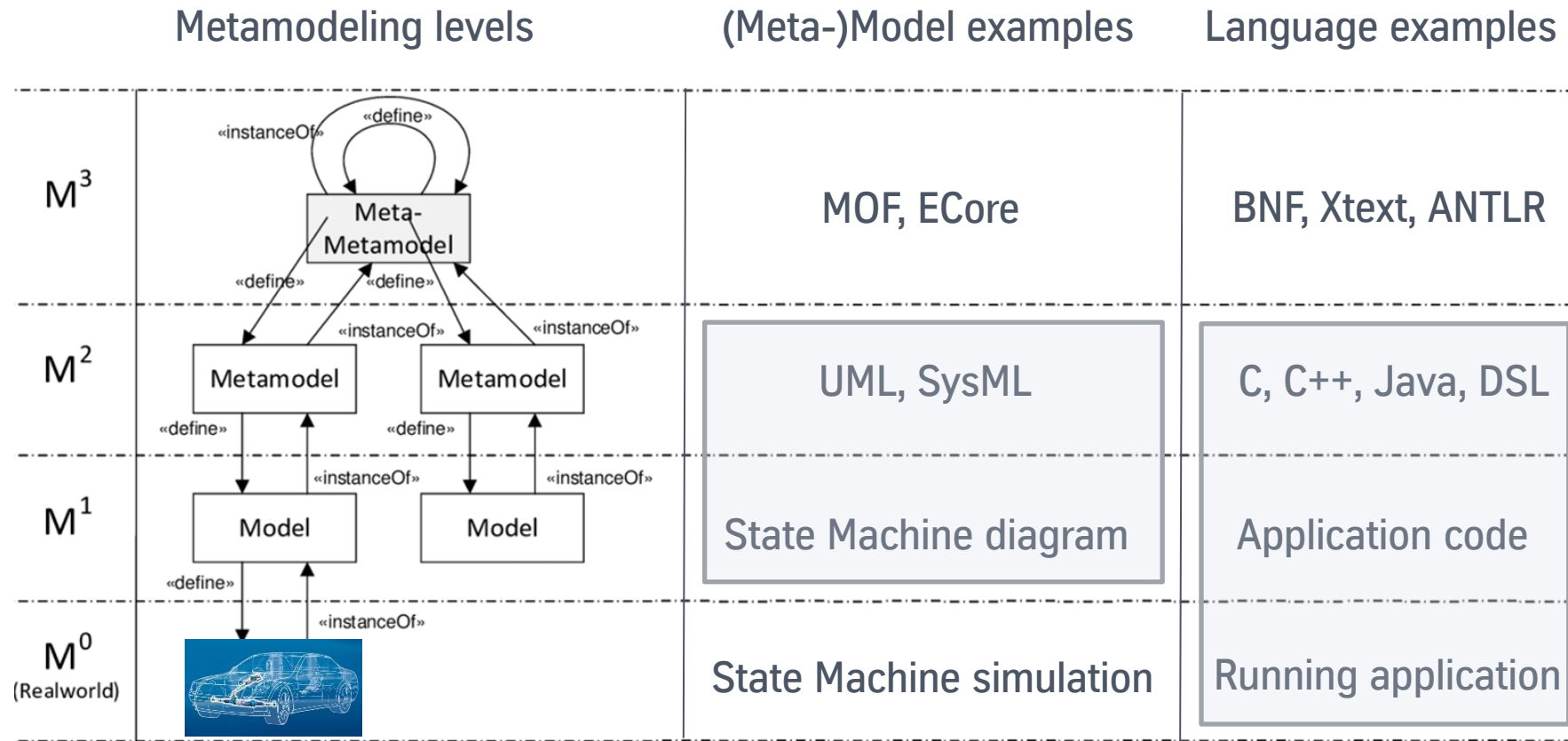
model-based systems engineering: "formalized methodology that is used to support the requirements, design, analysis, verification, and validation associated with the development of complex systems"



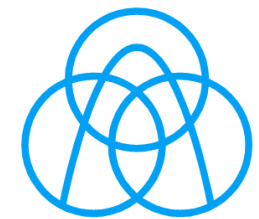
engineering.tomorrow.together.



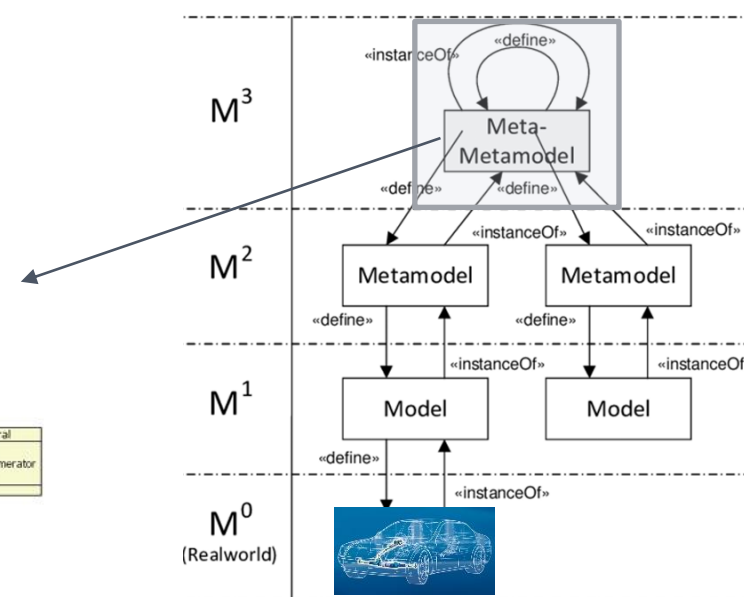
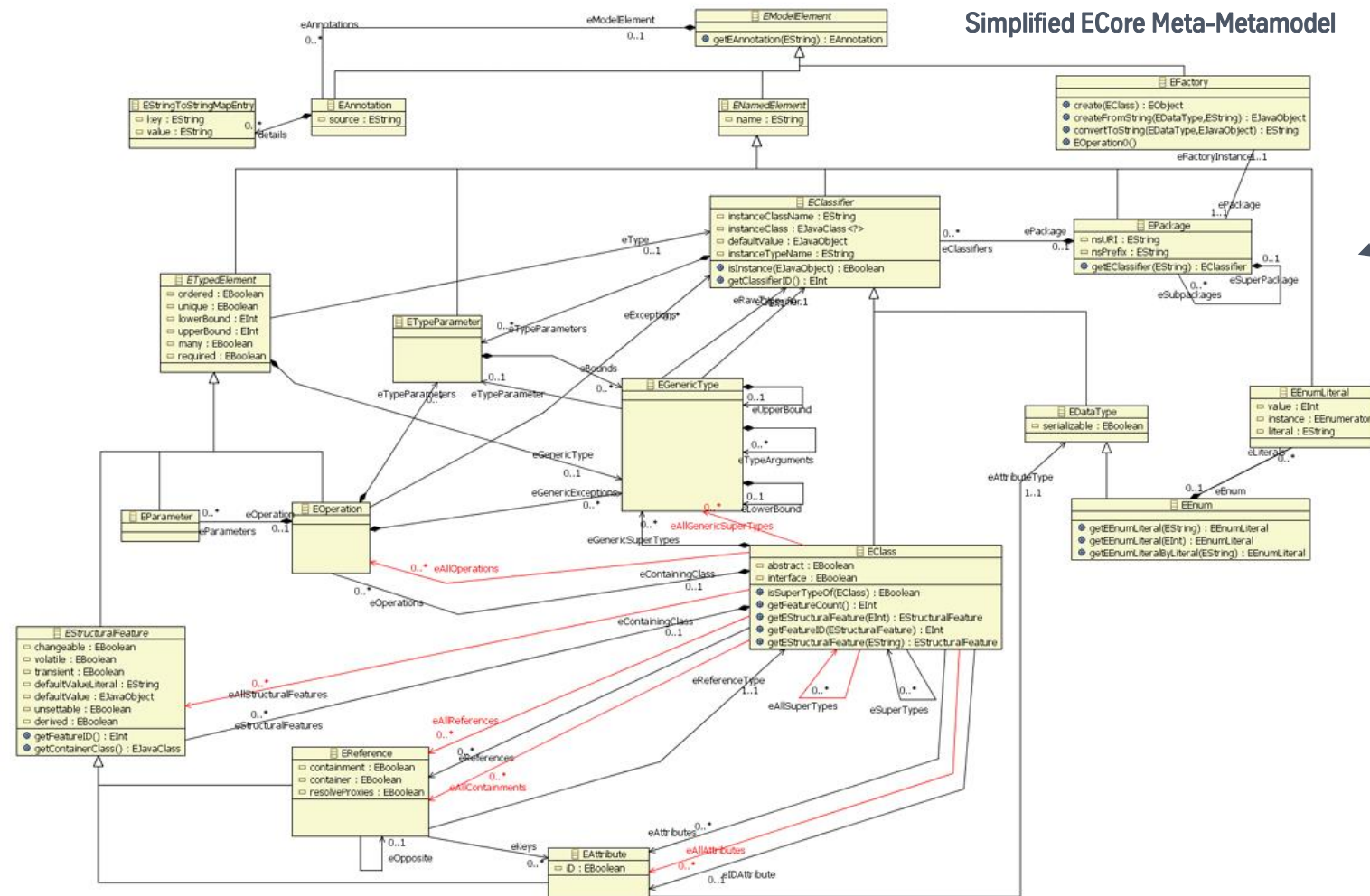
thyssenkrupp



engineering.tomorrow.together.



thyssenkrupp



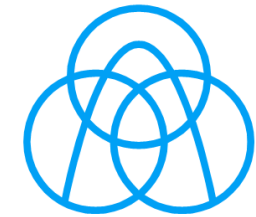
engineering.tomorrow.together.

thyssenkrupp

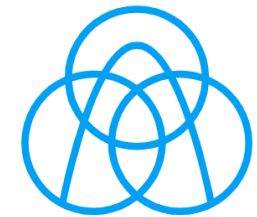
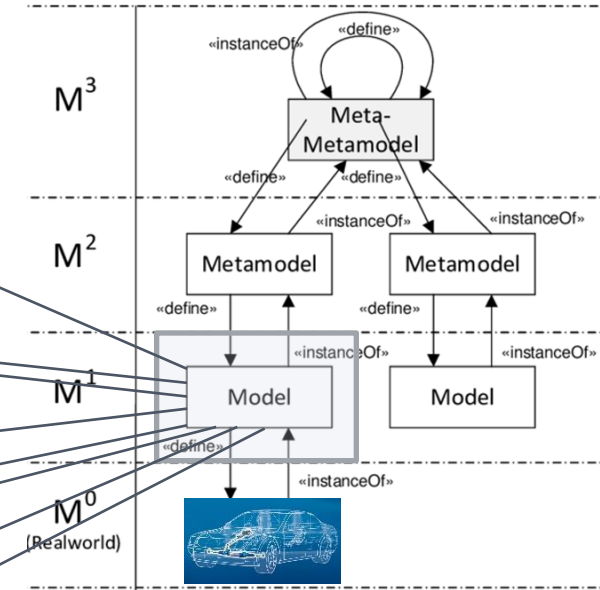
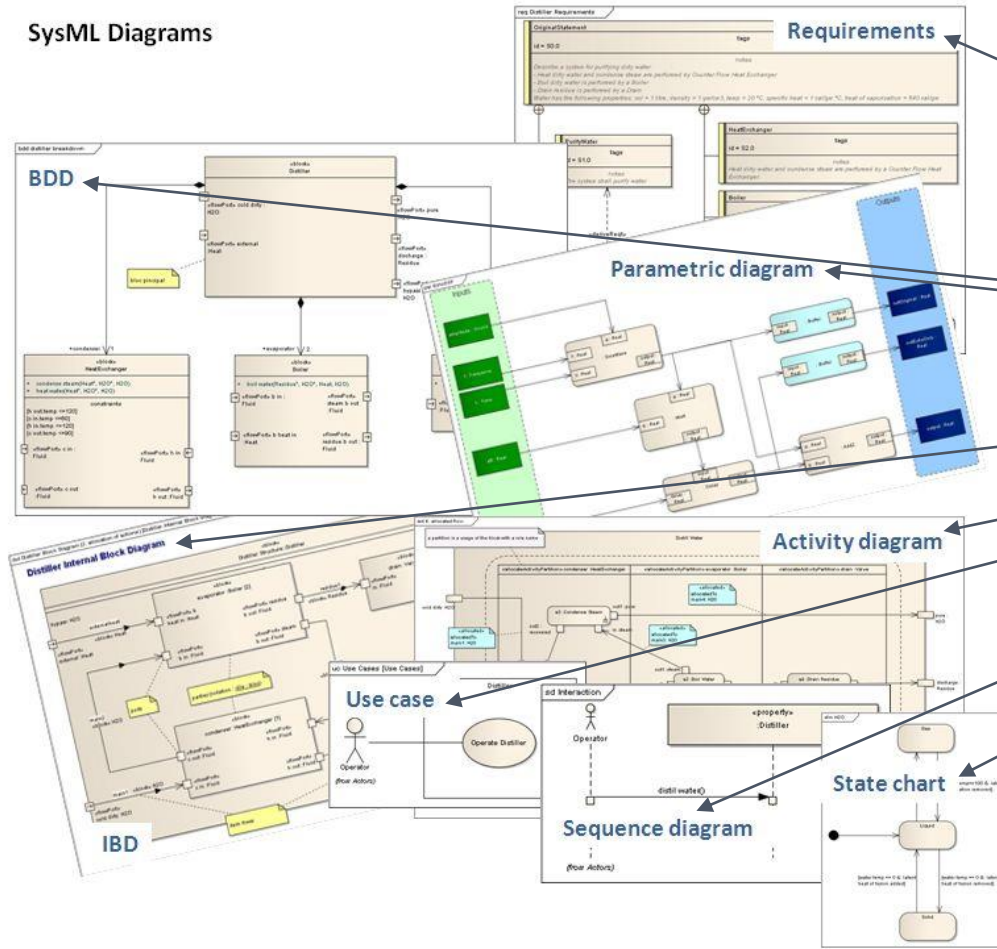
The diagram illustrates the SysML Metamodel structure, showing the relationships between various classes and their attributes. Key components include:

- RangeCardinality**: Attributes include `minCardinality: integer` and `maxCardinality: integer`.
- DomainCardinality**: Attributes include `minCardinality: integer` and `maxCardinality: integer`.
- Relation**: Attributes include `name: string`, `isVariable: boolean`, `<<opt>> isTransitive: boolean`, `<<opt>> isSymmetric: boolean`, `<<opt>> isReflexive: boolean`, and `<<opt>> isReflExive: boolean`.
- Concept**: Attributes include `name: string` and `isVariable: boolean`.
- Individual**: Attribute includes `name`.
- Attribute**: Attributes include `name: string`, `isVariable: boolean`, and `<<opt>> isFunctional: boolean`.
- GluingVariant**: Attribute includes `abstractDomainModel`.
- DomainModel**: Attribute includes `name: string`.
- Predicate**: Attributes include `consequent` and `antecedent`.
- Atom**: Attributes include `head` and `body`.
- Head**: Attributes include `consequent` and `antecedent`.
- DefaultDataSet**: Attribute includes `name: string`.
- CustomDataSet**: Attribute includes `name: string`.
- EnumerateDataSet**: Attribute includes `name: string`.
- DataSet**: Attributes include `range` and `valueOf`.
- DataValue**: Attributes include `lexicalForm: string` and `elements`.
- RelationMapset**: Attributes include `image` and `mapsetOf`.
- AttributeMapset**: Attributes include `image` and `mapsetOf`.

The diagram also includes several notes explaining the relationships and constraints between these classes, such as the requirement for `abstractDomainModel` to be a transitive parent and the constraints on `mapsetOf` and `image` relationships.

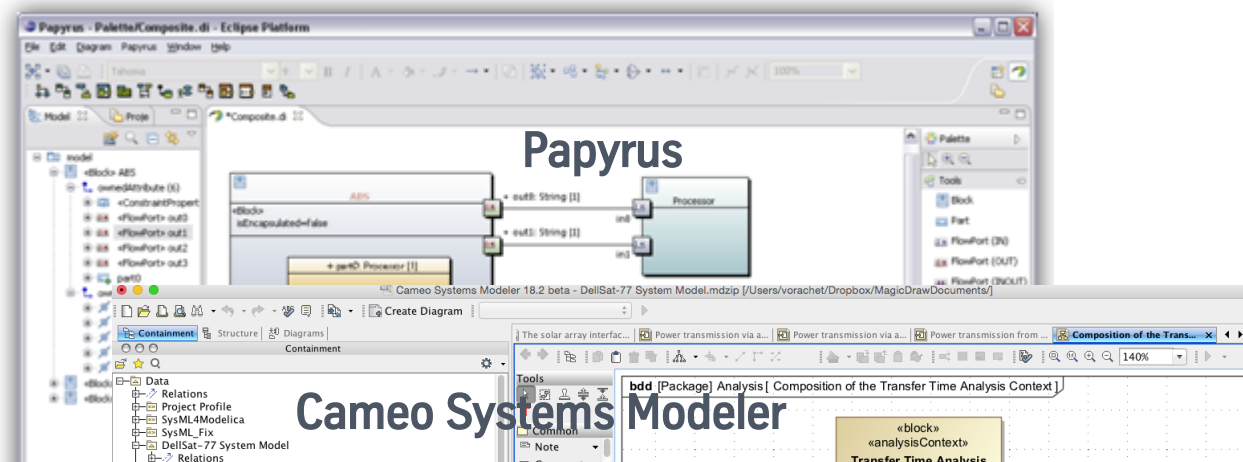


SysML Diagrams



engineering.tomorrow.together.

thyssenkrupp



```

Money discount;
discount = createPerson();


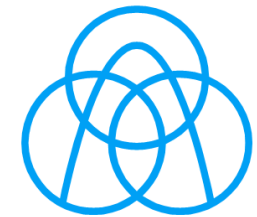
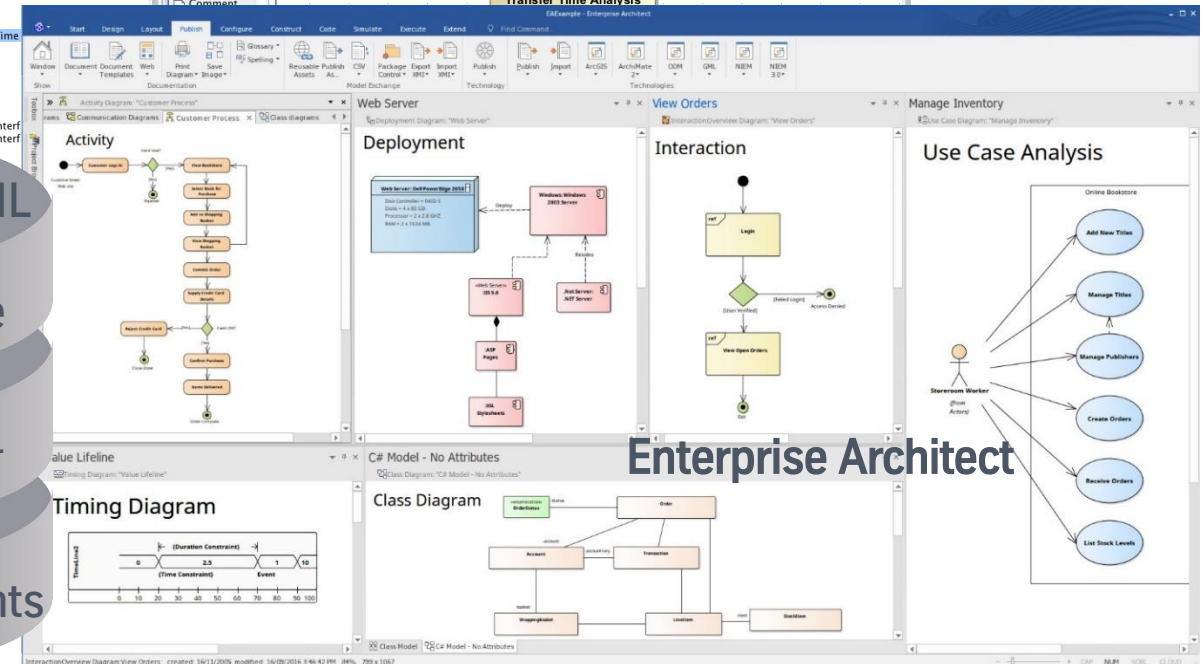
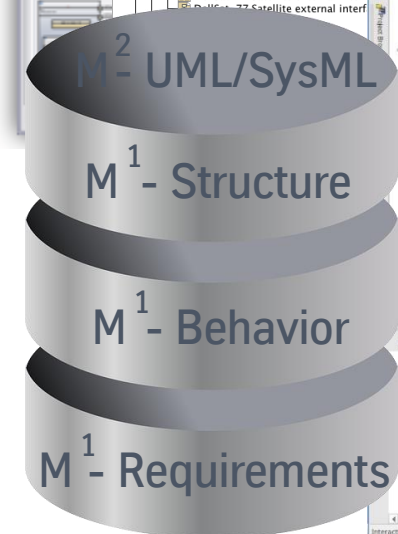
if (discount > 400 USD || discount >= 350 EUR) {
    discount = 300 EUR;
}

System.out.println("Your name: " + createPerson());
System.out.println("Your discount: " + discount);
}

public Money create(map<string, Object> person) {
    return Money Default: 0 EUR
    isChild(person)      isLevel_1(person)      isLevel_2(person)
    isAdult(person)      500 EUR              1000 EUR
    isRetired(person)     50 EUR + this.seasonalBonus()  100 EUR + this.seasonalBonus()
    200 EUR              250 EUR + (person["name"] == "Susan" ?
                        this.seasonalBonus() : 0 EUR)

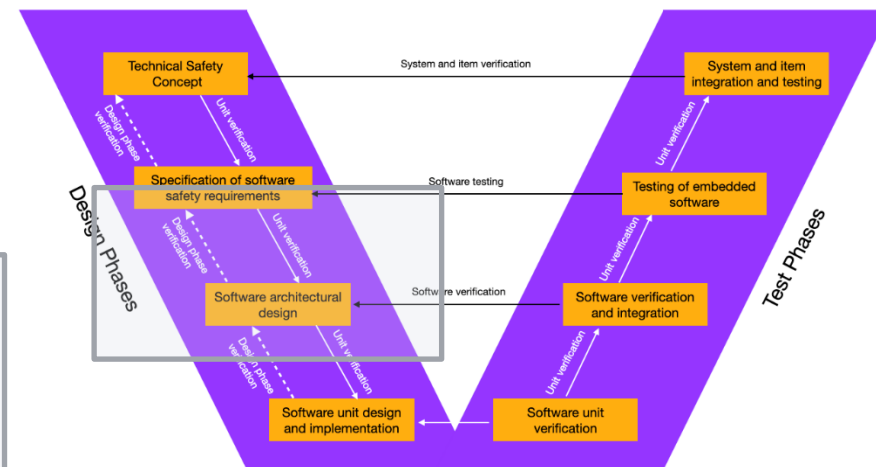
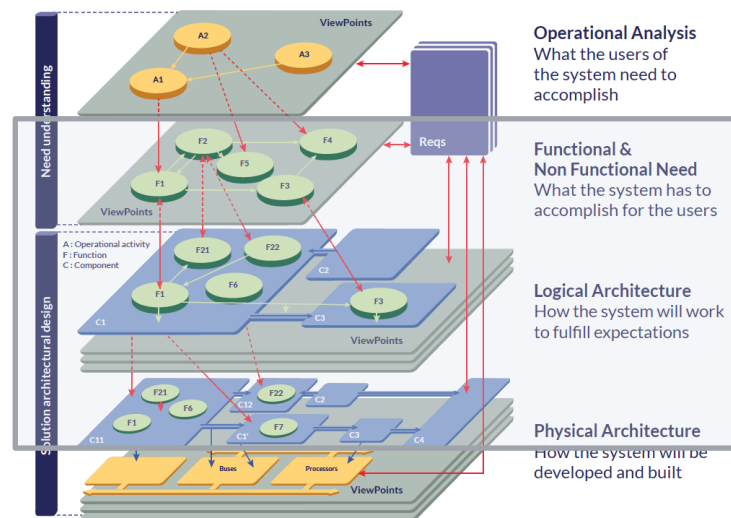
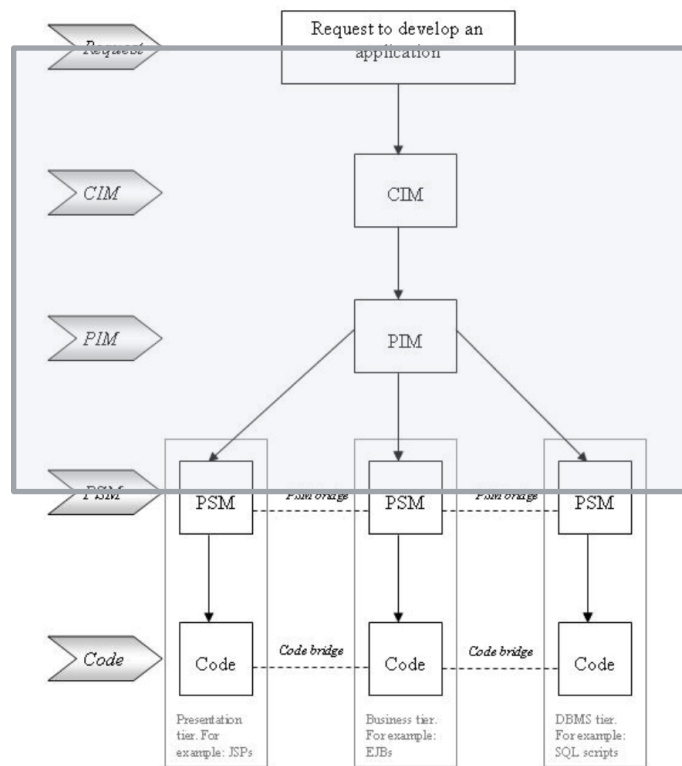
private Money seasonalBonus() {
    return 100 EUR;
}

```

engineering.tomorrow.together.

thyssenkrupp

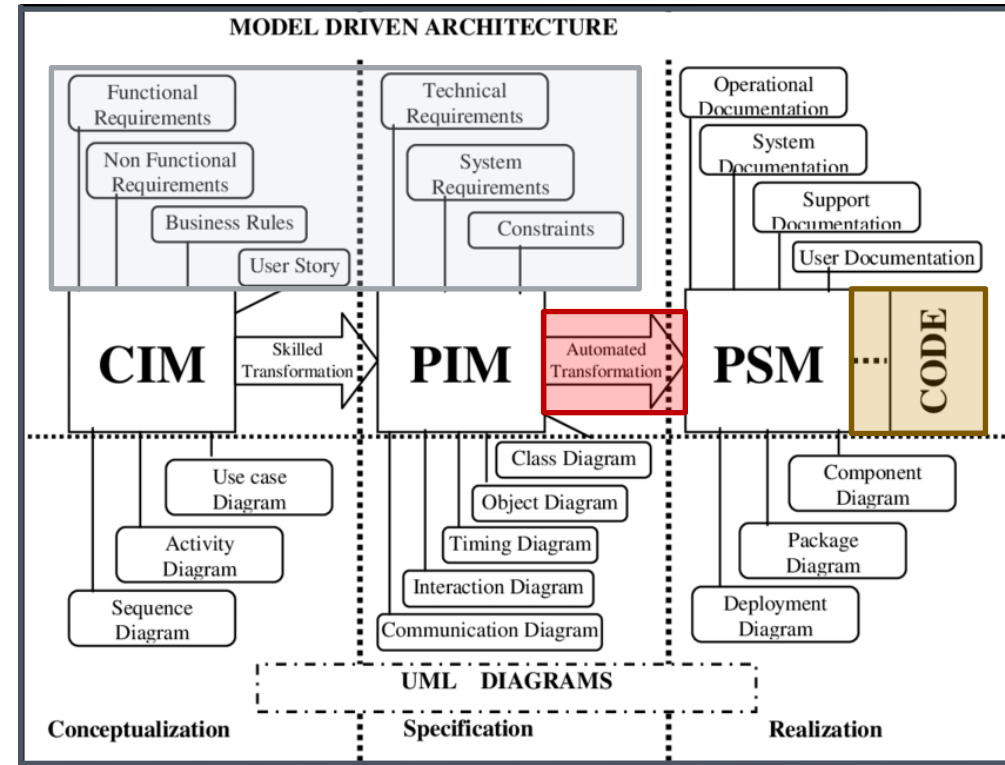
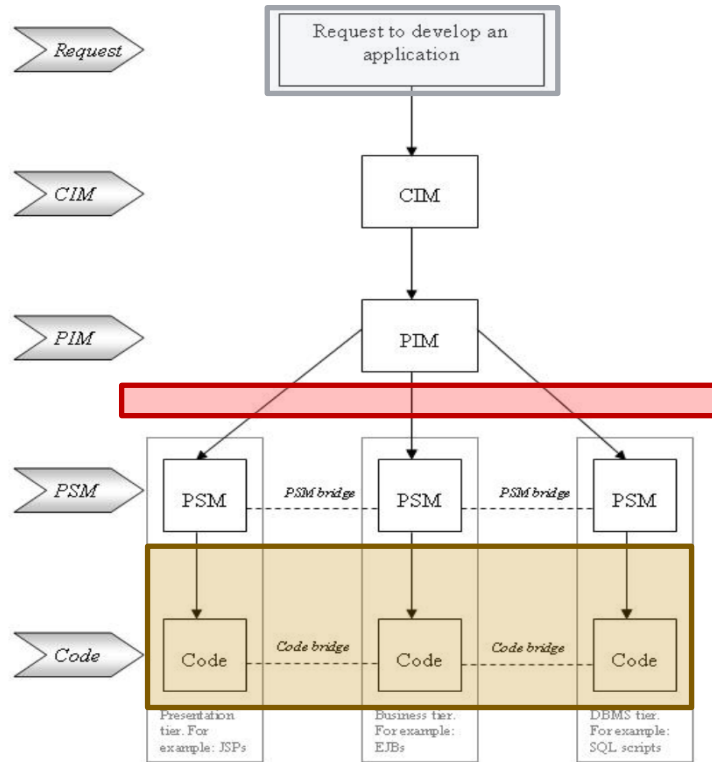


software architect: "software engineer responsible for high-level design choices related to overall system structure and behavior"

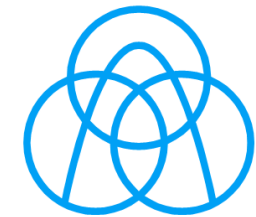
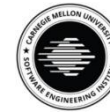


engineering.tomorrow.together.

thyssenkrupp

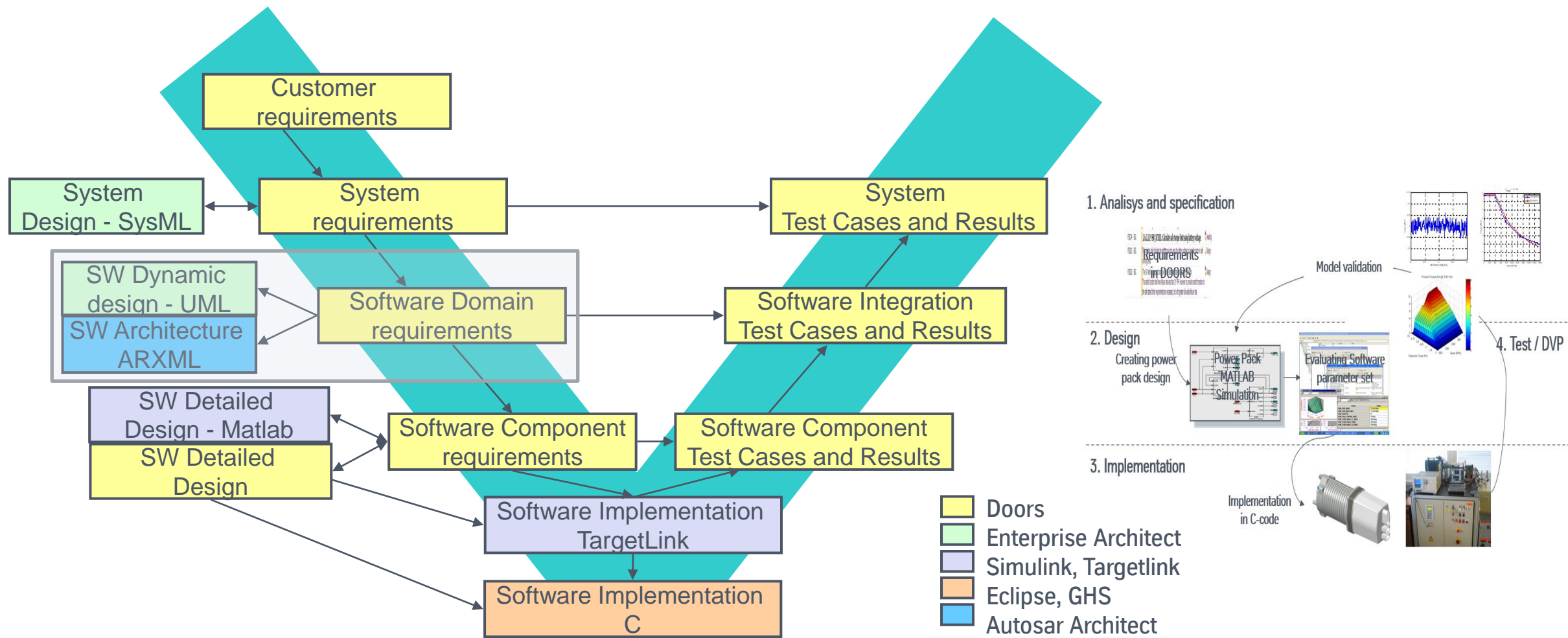


model-based systems engineering: "formalized methodology that is used to support the requirements, design, analysis, verification, and validation associated with the development of complex systems"



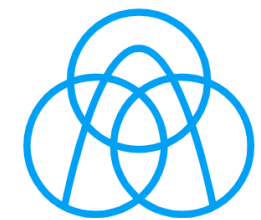
engineering.tomorrow.together.

thyssenkrupp

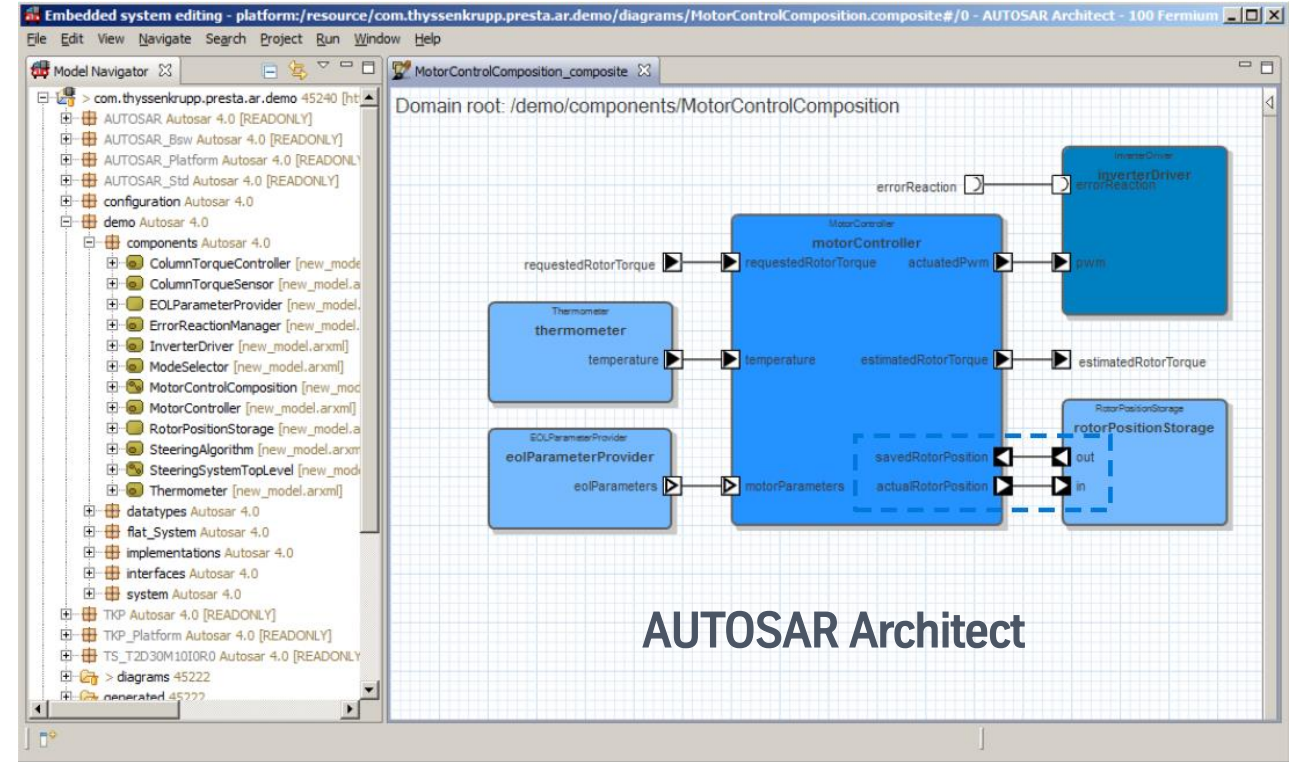
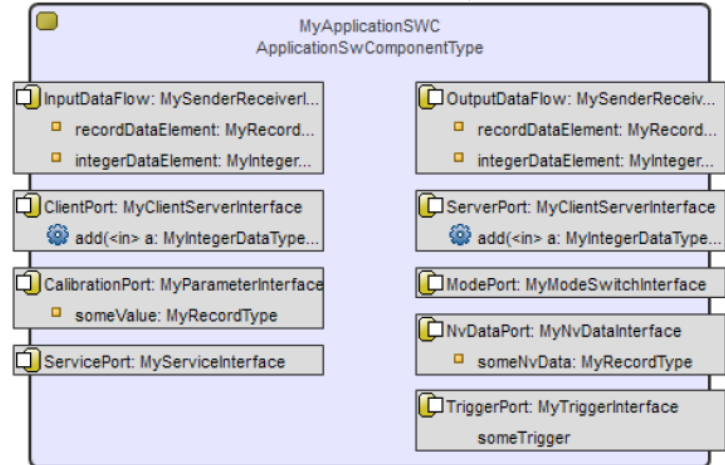
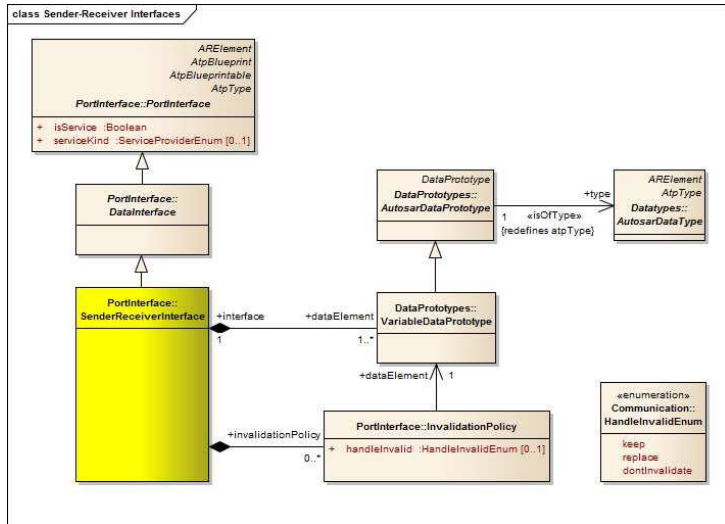


model-based systems engineering: "formalized methodology that is used to support the requirements, design, analysis, verification, and validation associated with the development of complex systems"

engineering.tomorrow.together.



thyssenkrupp



engineering.tomorrow.together.

