

A solid orange vertical bar is positioned on the left side of the slide.

Modelling in Enterprise Architecture



Models and Modelling

Modelling

Describing and Representing all relevant aspects of a domain in a defined language.

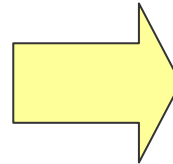
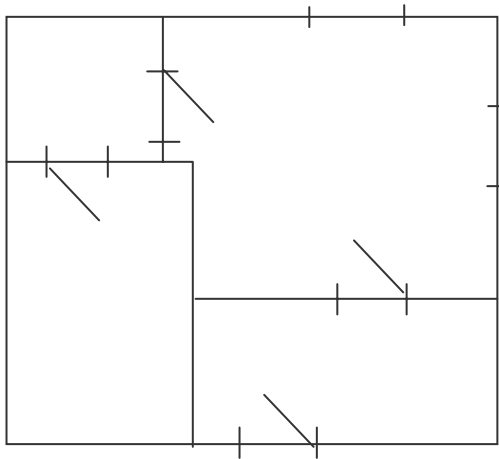
Result of modelling is a model - an exemplary reproduction of reality.

Model

A reproduction of the part of reality which contains the essential aspects to be investigated.

Model and Real Object in Architecture

modell (plan)



real object

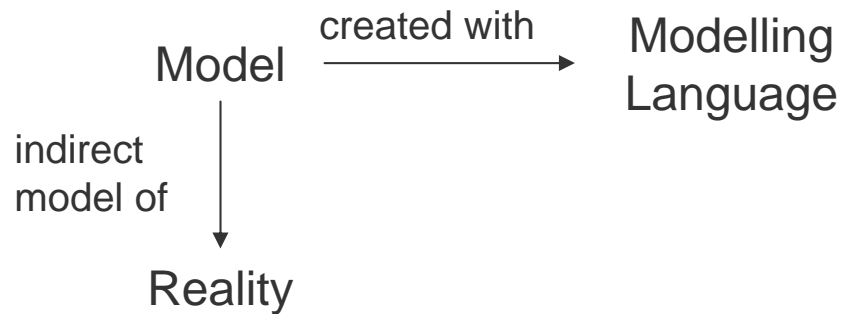


Rationale for Modelling

- Models provide abstractions of a physical system that allow engineers to reason about that system by ignoring extraneous details while focusing on relevant ones.
- All forms of engineering rely on models to understand complex, real-world systems.
- Models are used in many ways:
 - ◆ predict system qualities
 - ◆ reason about specific properties when aspects of the system are changed
 - ◆ communicate key system characteristics to various stakeholders

(Brown 2004)

Modelling Language



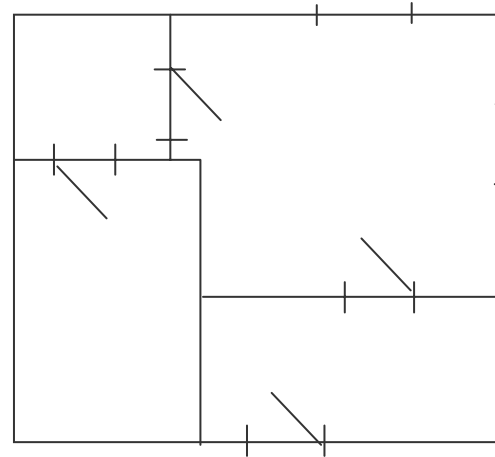
- A modelling "language" specifies the building blocks from which a model can be made.
- There can be different types of modelling languages, depending on the kind of model
 - ◆ graphical model
 - ◆ textual description
 - ◆ mathematical model
 - ◆ conceptual model
 - ◆ physical model

Model and Modelling Language in Architecture

real object



modell (plan)




modelling language

object types:

_____ wall


 door


 window

rules:

- a door is adjacent to a wall on both sides
- Windows are on outer walls.

Model Types

- For complex systems many aspects might be of interest.
- One can use various modeling concepts and notations to highlight one or more particular perspectives, or views, of that system, depending on what is relevant
 - ◆ at any point in time
 - ◆ for a particular stakeholder

Model Types in Enterprise Architecture

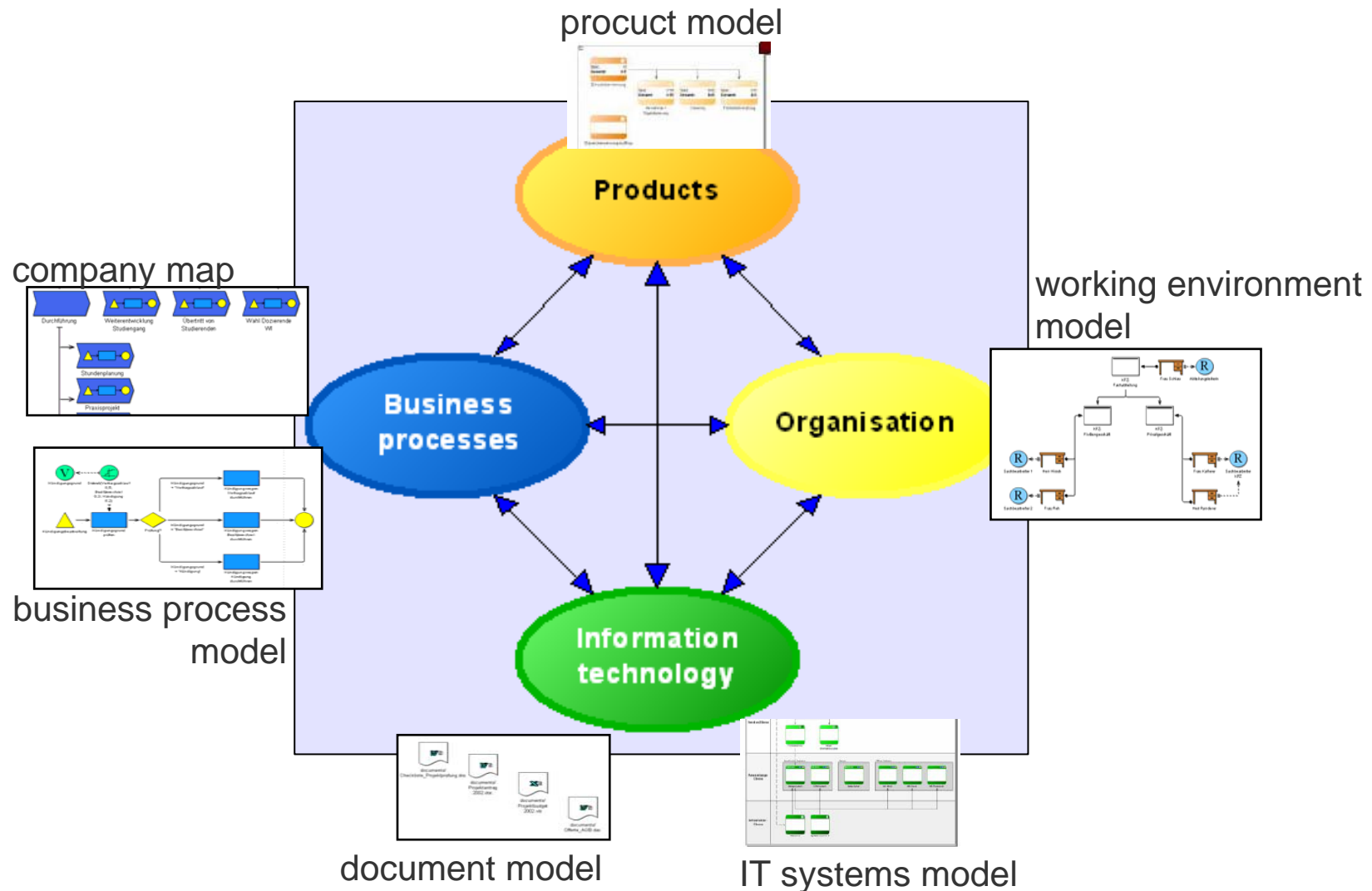
- There exist different model types for different perspectives and aspects (boxes in the Zachman framework, views in ARIS etc.), e.g.
 - ◆ Process
 - ◆ Organisation
 - ◆ Data
- For the same view or aspect there can again be different model types, e.g.
 - ◆ event-driven proces chains
 - ◆ flow diagrams
 - ◆ petri nets
 - ◆ activity diagrams (UML)
 - ◆ BPMN

depending on required level of detail, relevant aspects, preferences of users etc.

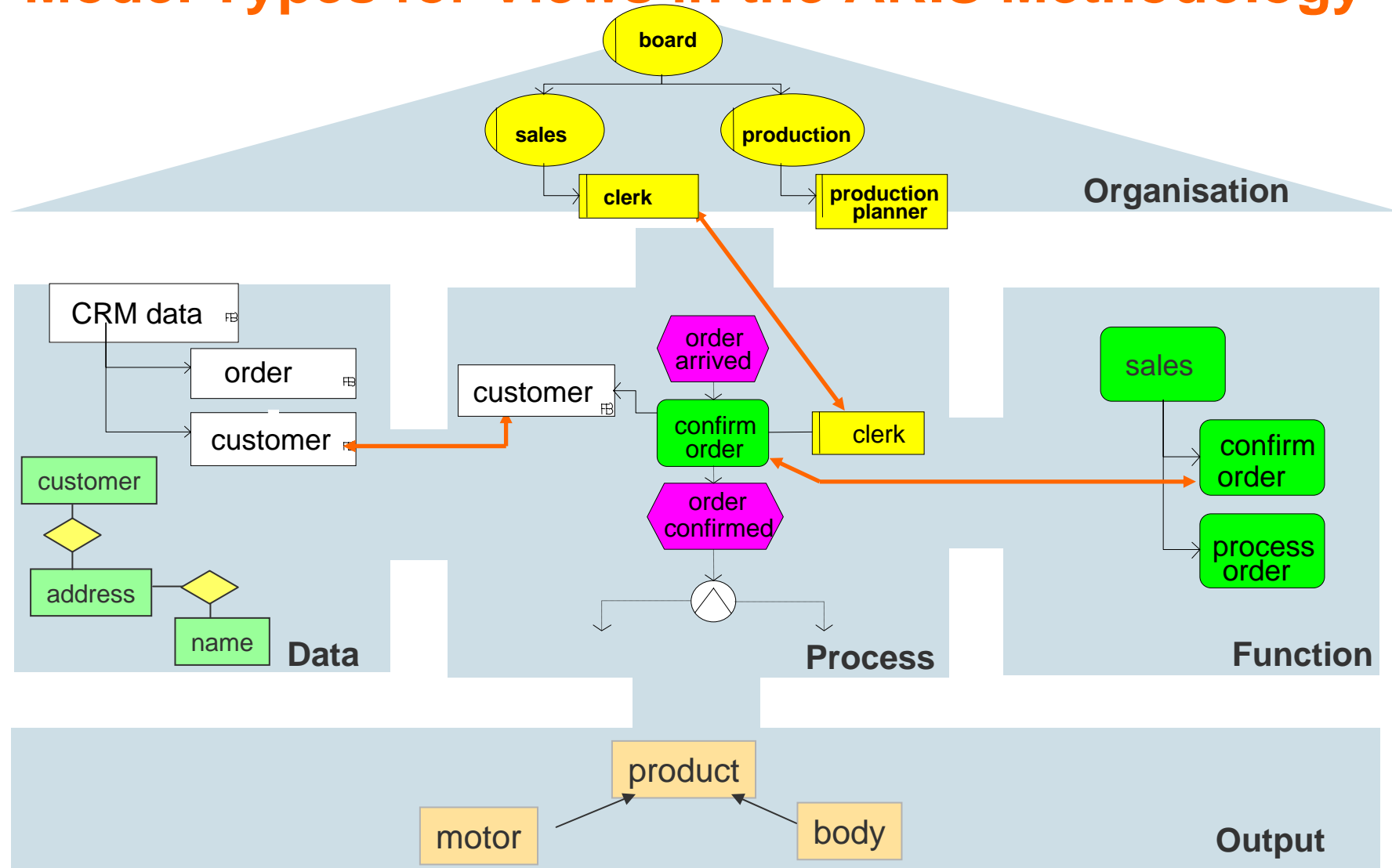
Model Types of the Zachman Framework

	DATA <i>What</i>	FUNCTION <i>How</i>	NETWORK <i>Where</i>	PEOPLE <i>Who</i>	TIME <i>When</i>	MOTIVATION <i>Why</i>
Objective/Scope (Contextual) → Role: Planner	List of Things important in the Business	List of Core Business Processes	List of Business Locations	List of important Organizations	List of Events	List of Business Goals/Strategies
Enterprise Model (Conceptual) → Role: Owner	Conceptual Data/ Object Model	Business Process Model	Business Logistics System	Work Flow Model	Master Schedule	Business Plan
System Model (Logical) → Role: Designer	Logical Data Model	System Architecture Model	Distributed Systems Architecture	Human Interface Architecture	Processing Structure	Business Rule Model
Technology Model (Physical) → Role: Builder	Physical Data/ Class Model	Technology Design Model	Technology Architecture	Presentation Architecture	Control Structure	Rule Design
Detailed Representations (Out of Context) → Role: Programmer	Data Definitions	Program	Network Architecture	Security Architecture	Timing Definition	Rule Specification
Functioning Enterprise → Role: User	Usable Data	Working Function	Usable Network	Functioning Organization	Implemented Schedule	Working Strategy

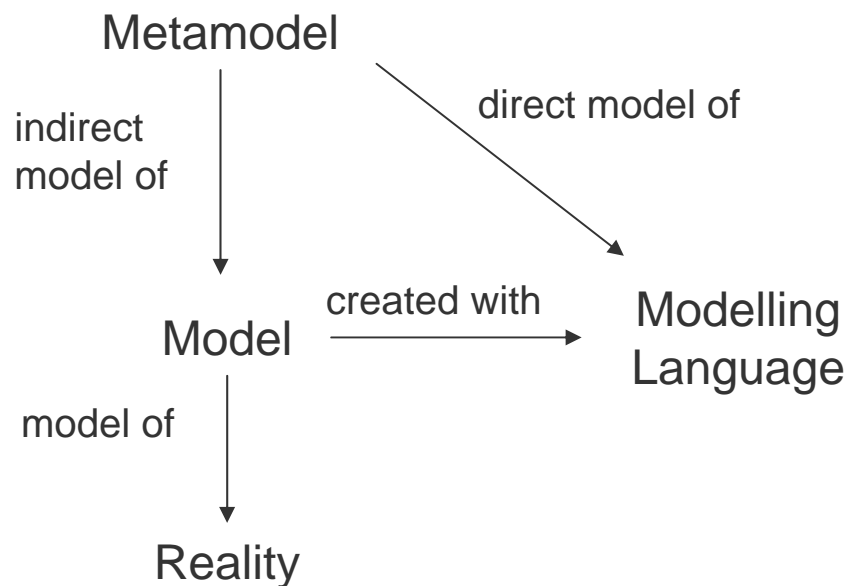
Model Types of the BPMS Methodology: Views



Model Types for Views in the ARIS Methodology



Meta Model



A meta model defines the modelling language, i.e. the building blocks that can be used to make a model. It defines the

- ◆ the object types that can be used to represent a model
- ◆ relations between object types
- ◆ attributes of the object types
- ◆ the meaning of the object types
- ◆ rules to combine object types and relations

Model Types, Meta Models and Models

Model types:

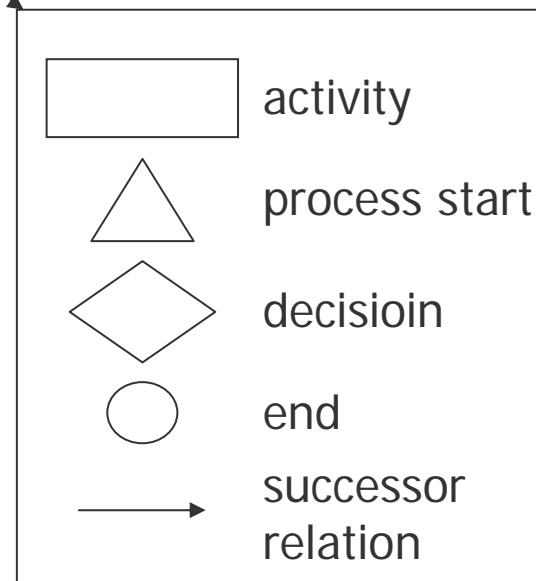
Specify a kind of model for a particular view or perspective:

- flow diagram
- organisational diagram
- activity diagram
- entity-relationship model
- ...

Meta model

(modelling language):

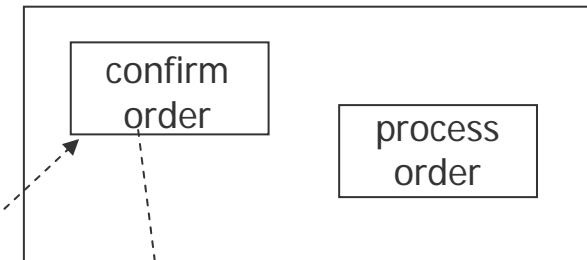
Object types with predefined meaning that can be used to make a model of a particular type



Object „confirm order“ is an instance of the object type "activity"

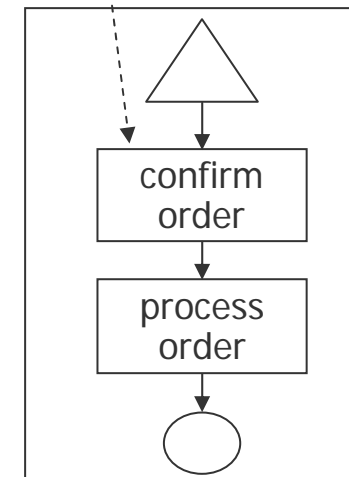
Objects:

Instances of object types representing real entities



Instance of "confirm order" in a particular model

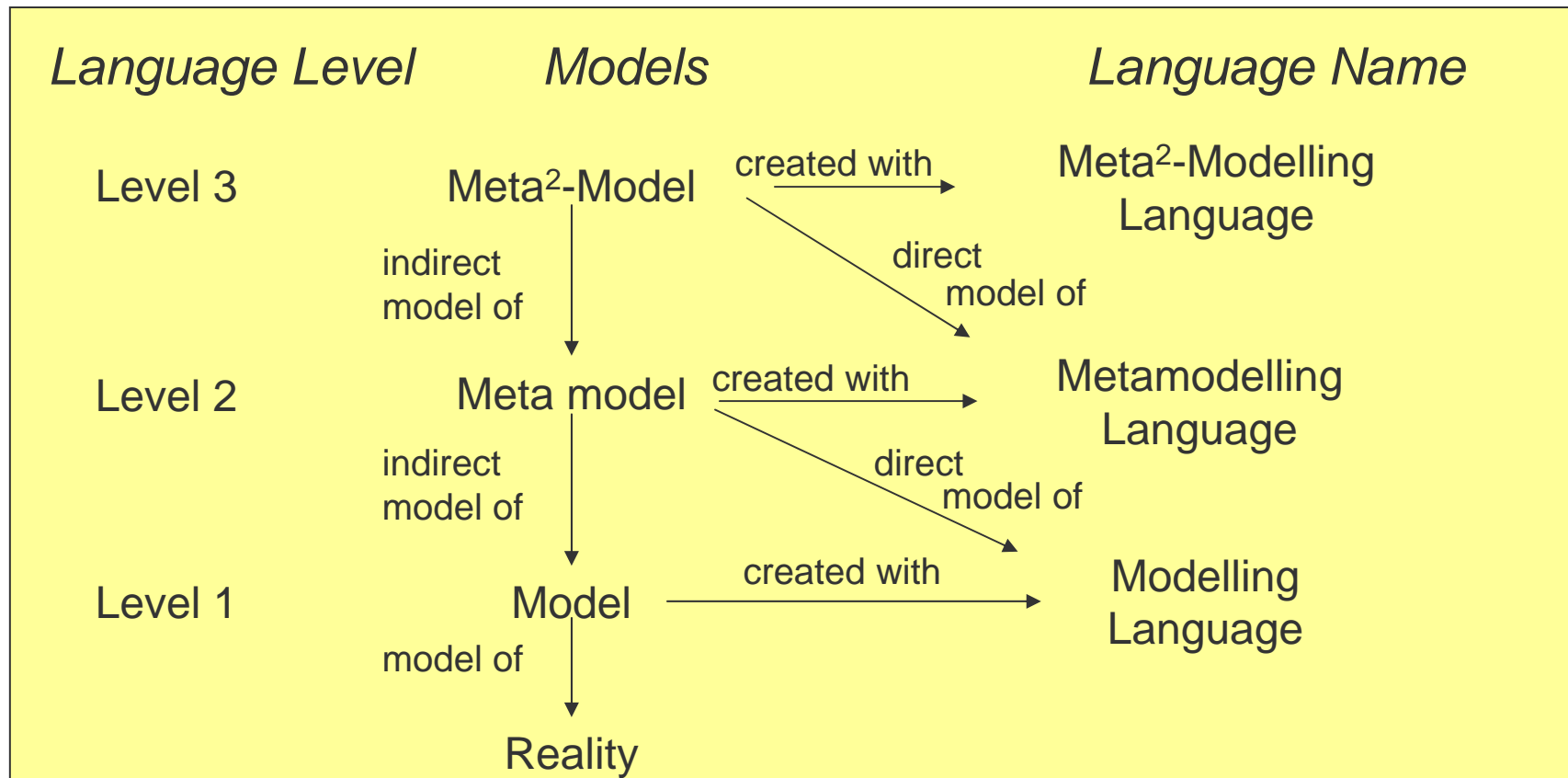
Model:



combination of object representing the whole system

Meta Model Hierarchy

The meta-model must again be described in some language, which has to be specified in a meta model



Often the metamodelling language is a kind of natural language, represented in a manual.

Model-driven Architecture MDA

- Model-driven Architecture MDA is defined by the Object Management Group OMG.
- It is a way
 - ◆ to organize and manage enterprise architectures
 - ◆ supported by automated tools and services
 - ◆ for defining models and facilitating transformations between different types of models

<http://www.omg.org>

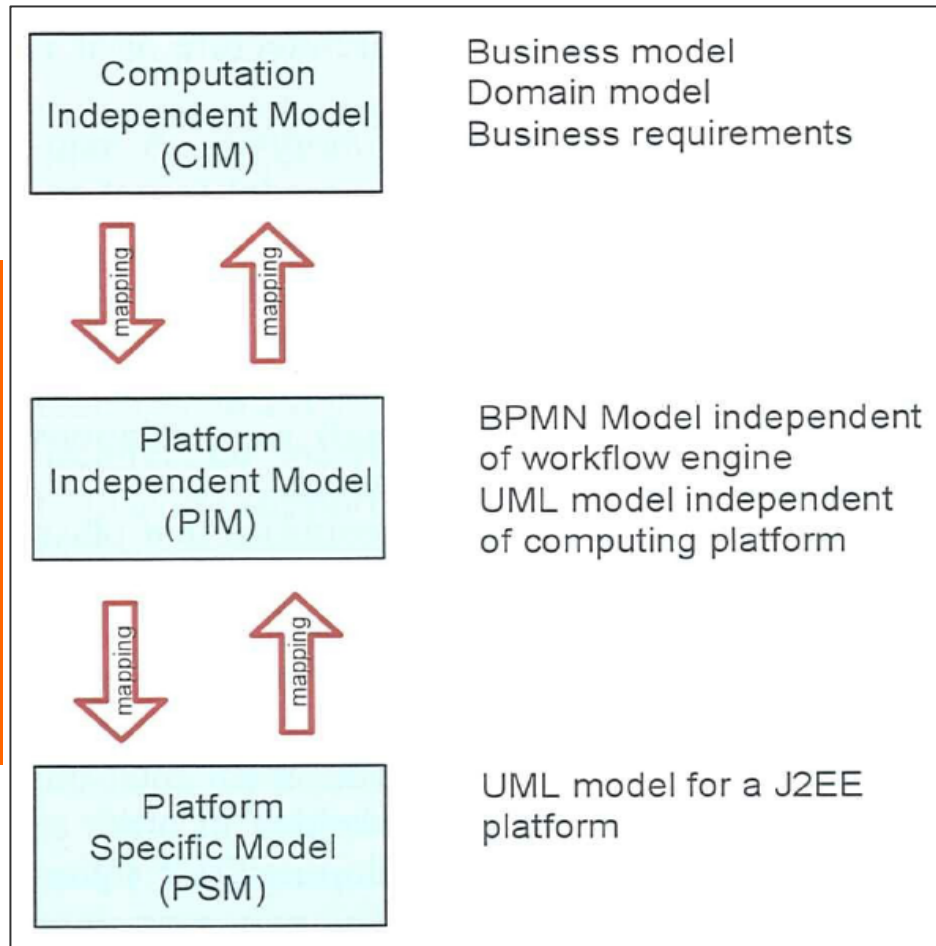
(Brown 2004)



Models and Transformation

- Three ideas are important here with regard to the abstract nature of a model and the detailed implementation it represents:
- **Model classification.** In all software and system development there are important constraints implied by the choice of languages, hardware, network topology, communications protocols and infrastructure, and so on. Each of these can be considered elements of a solution "platform." An MDA approach helps us to focus on what is essential to the business aspects of a solution being designed, separate from the details of that "platform."
- **Platform independence.** The notion of a "platform" is rather complex and highly context dependent. For example, in some situations the platform may be the operating system; in some situations it may be a technology infrastructure. In any case, it is more important to think in terms of what models at different levels of abstraction are used for what different purposes, rather than to be distracted with defining the "platform."
- **Model transformation and refinement.** transformations between models become first class elements of the development process. This is important because a great deal of work takes places in defining these transformations, often requiring specialized knowledge of the business domain, the technologies being used for implementation, or both. We can improve the efficiency and quality of systems by capturing these transformations explicitly and reusing them consistently across solutions.

Model-Driven Architecture MDA



MDA comprises three levels of abstraction with mappings between them

CIM Computation-Independent Model

- ♦ modelling the requirements for the system describing the situation in which the system will be used
- ♦ hiding much or all information about the use of IT systems

PIM Platform-Independent Model

- ♦ describing operations of the system while hiding details for a particular platform
- ♦ describing those parts of the system specification that do not change from one platform to another

PSM Platform-Specific Model

- ♦ Combines specifications of PIM with details about a particular type of platform

Principles of MDA

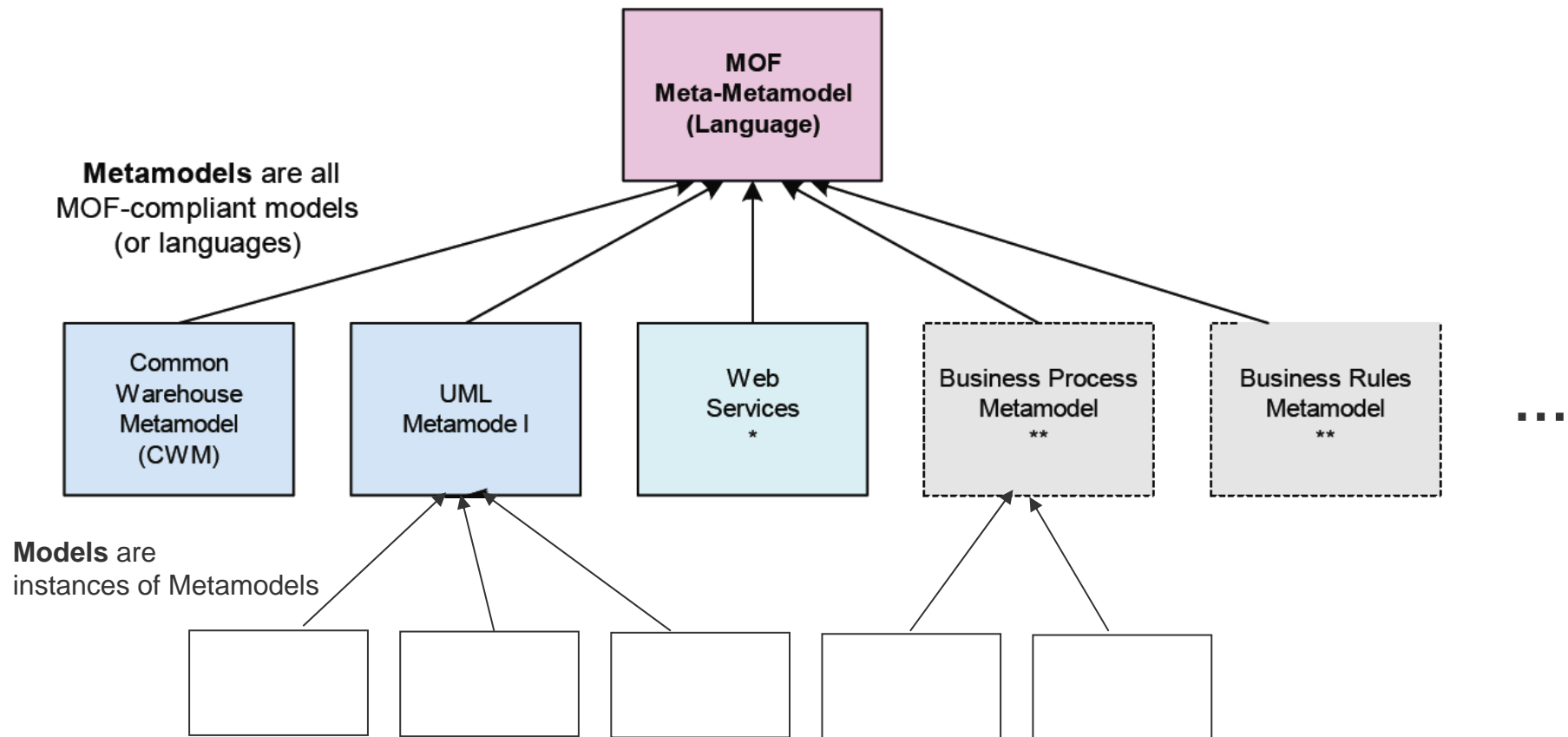
- Models expressed in a well-defined notation are a cornerstone to understanding systems for enterprise-scale solutions.
- The building of systems can be organized around a set of **models** by
 - ◆ imposing a series of transformations between models,
 - ◆ organized into an architectural framework of layers and transformations.
- A formal underpinning for describing models in a set of **metamodels**
 - ◆ facilitates meaningful integration and transformation among models
 - ◆ and is the basis for automation through tools.
- Acceptance and broad adoption of this model-based approach requires industry standards to provide openness to consumers, and foster competition among vendors.

(Brown 2004)

Metamodelling in MDA

- Underlying these model representations, and supporting the transformations, is a set of metamodels.
- The ability to analyze, automate, and transform models requires a clear, unambiguous way to describe the semantics of the models.
- Hence, the models intrinsic to a modeling approach are described in a model, which is called a metamodel. For example, the UML metamodel describes in precise detail the meaning of a class, an attribute, and the relationships between these two concepts.
- The OMG has defined a set of metamodeling levels as well as a standard language for expressing metamodels: the Meta Object Facility (MOF).
- A metamodel uses MOF to formally define the abstract syntax of a set of modeling constructs.

Model Levels of MDA



OMG business modeling specifications

OMG business modeling specifications that provide automated and interoperability support for Business Process Management (BPM) and business modeling:

- ◆ Business Motivation Model (BMM)
- ◆ Business Process Modeling Notation (BPMN)
- ◆ Business Process Definition Metamodel (BPDM)
- ◆ Semantics of Business Vocabulary and Business Rules (SBVR)
- ◆ Business Process Maturity Model (BPMM)
- ◆ Production Rule Representation (PRR)
- ◆ Workflow Management Facility

OMG Modeling and Metadata Specifications

- Meta Object Facility (MOF™)
- Unified Modelling Language™ (UML®)
- XML Metadata Interchange (XMI®)
- Common Warehouse Metamodel (CWM™)
- Object Constraint Language (OCL)
- Systems Modelling Language (SysML)
- Ontology Definition Metamodel (ODM)
- Reusable Asset Specification (RAS)
- Software Process Engineering Metamodel (SPEM)