What is a Model? (And if yes, how many?)

Version 1.01

This document developed by
Dr. Juergen Pitschke, BCS-Dr. Juergen Pitschke, www.enterprise-design.eu

Table of Content

1 What is a Model?

The easiest terms often cause the biggest problems. We use the term „Model" quite often talking about visual modeling using BPMN, UML, SysML and other standard notations. Sometimes the term "Diagram" is used as a synonym caused by an uncritical view and sometimes caused by used tools. But this is wrong: Content and Presentation are mixed up. The content is most important. In most cases the same content can be presented using different presentation forms. The same model can be presented using different diagram types or different notations.

The document was motivated from questions of my students. Understanding the term "Model" is important in Enterprise Modeling or Software Engineering. If you e.g. find a work package in the project plan named "Develop Class Diagram" you easily recognize that the understanding of the term "Model" is not given. If the project team has an understanding of "Model" the same work package would maybe be named "Develop Information Model for End User Portal".

The discussion is done in the context of ***visual models***. This means in the context of graphical presentation of processes, systems, requirements, etc. using standard notations as BPMN, UML, SysML and other notations.

1.1 The Term "Model"

What does the term "Model" really mean? Wiktionary gives 7 different meanings for the word "Model" in English:

"model (*plural* **models**)
1. A person who serves as a subject for artwork or fashion, usually in the medium of photography but also for painting or drawing.
   The beautiful model had her face on the cover of almost every fashion magazine imaginable.
2. A miniature representation of a physical object.
   The boy played with a model of a World War II fighter plane.
3. A simplified representation (usually mathematical) used to explain the workings of a real world system or event.
   The computer weather model did not correctly predict the path of the hurricane.
4. A style, type, or design.
   He decided to buy the turbo engine model of the sports car.
5. The structural design of a complex system.
   The team developed a sound business model.
6. A praiseworthy example to be copied, with or without modifications
   British parliamentary democracy was seen as a model for other countries to follow
7. (*logic*) An interpretation function which assigns a truth value to each atomic proposition."[1]

Definitions 3, 5 and 6 are of interest in the context of "Visual Modeling". But also definitions 2 and 7 are maybe related.

[1] http://en.wiktionary.org/wiki/model, 31.08.2009

1.2 Model ("Simple Model")

We want to use the term "Model" always for a representation in the sense of definition 3. A model is a simplified representation of a subject of interest. A model is created through abstraction – we ignore some aspects of the subject to highlight another aspect or attribute. To distinguish this kind of model from a "composite model" (see paragraph 1.3) we also use the term "simple model".

A good simple model shows exactly one aspect or perspective of the subject of interest – and really only one. Mixing different views in one model always causes models which are ambiguous, difficult to understand and not maintainable. E.g. if we are dealing with business process modeling the model should show either the As-Is or the To-Be Process, but not mixing both views in one model. If we are doing process decomposition a single model should show one level of detail, but not mixing e.g. Macro-Process View and Task-Level View. Different aspects are important when analyzing business processes – the sequence flow within one process, the information exchange between process participants, and others. These views should be separated into different models. This is addressed by version 2.0 of the BPMN. The new diagram types support the separation of the different concerns better than the previous version.

We can summarize some basic modeling principles from this:
- A good (simple) model shows exactly one aspect of the subject of interest.
- Independent concepts are represented independently, e.g. separation of business process and business rules.
- Stable concepts are shown separated from instable concepts, e.g. separation of roles in a business process and organizational structure of the business unit.

1.3 Composite Models

A single view (one simple model) is never enough to describe a business area or any relevant subject. Therefore a sufficient description of a "subject of interest" will always include a combination of different simple models. We will name such combinations of simple models "Composite Model". You will know such models already. Names like "Enterprise Model" or "System Model" point to composite models. The term "Composite Model" refers to definition 5 in Wiktionary.

Our modeling approach uses the Zachman Framework as the underlying architecture pattern. We assign simple and composite models to the perspectives, views and cells of the framework. A simple model is always assigned to a single cell and to a single level of detail. Paragraph 1.4 will discuss this in more detail.

Samples for composite models:
- Enterprise Model: includes all perspectives and views of the Zachman Framework.
- Business Model: includes all simple models of the perspective "Business Concepts"
- System Model: includes all simple models of the perspectives "System Logic" and "Technology Physics"
- Model of a „Service"[2]: Combination of simple models of the perspectives "System Logic" and "Technology Physics"

Composite Models are always composed of multiple simple models. They can also combine multiple other composite models. We need to describe the relations between models (simple and composite): A model as a refinement of another model, the mapping from As-Is to To-Be Process, or the combination of simple models establishing the "Business Model". Such aspects

[2] Service in the context of Service Oriented Architecture (SOA)

are also often described using visual models. A "Process Map" shows the connection between different business process models, an "Overview Model" shows relations between different views on a subject.

1.4 Models and the Zachman Framework

The Zachman Framework is not a method or a process. It defines an architecture pattern, presented by a matrix. It defines perspectives (in the rows) and views (in the columns) to define an enterprise architecture. Figure 1 shows the framework.

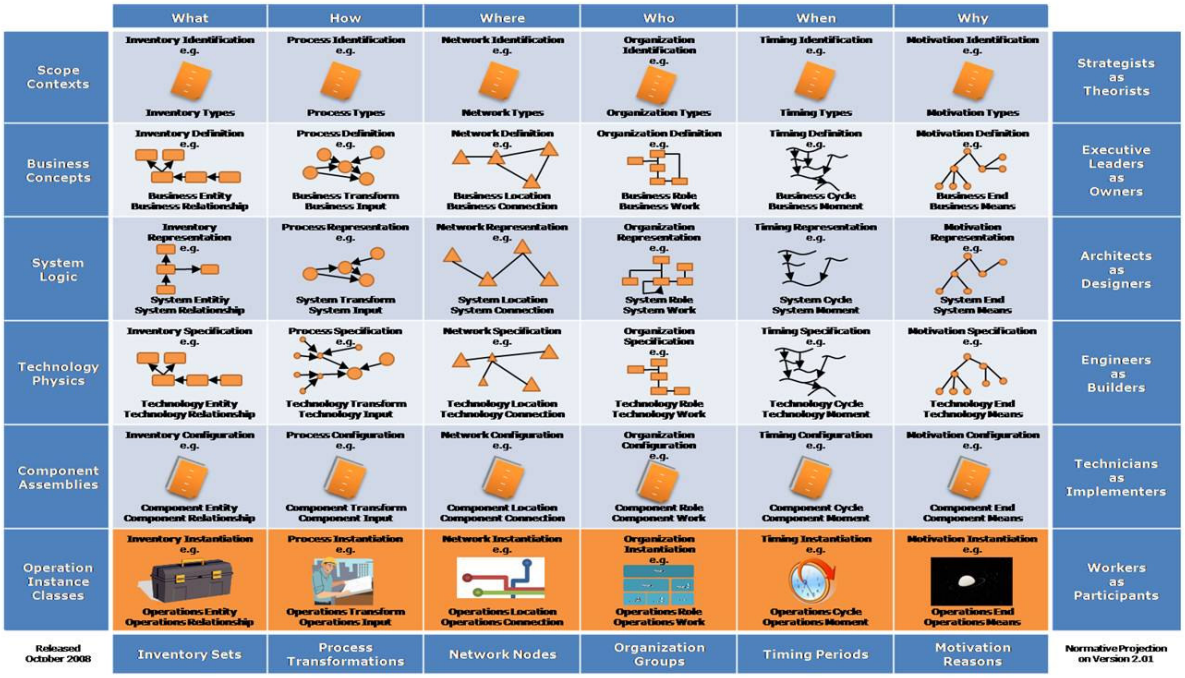

Figure 1: Zachman™-Framework[3], Source: www.zachmaninternational.com

A common misunderstanding about the Zachman Framework is that the perspectives (rows) contain more detail from top to bottom. That's wrong. Each row defines a new perspective. An increase in detail happens within each cell. The Framework is three dimensional. Figure 2 tries to make this point.

[3] John Zachman, The Zachman Framework For Enterprise Architecture: Primer for Enterprise Engineering and Manufacturing, Zachman International, 2006, Electronic Book

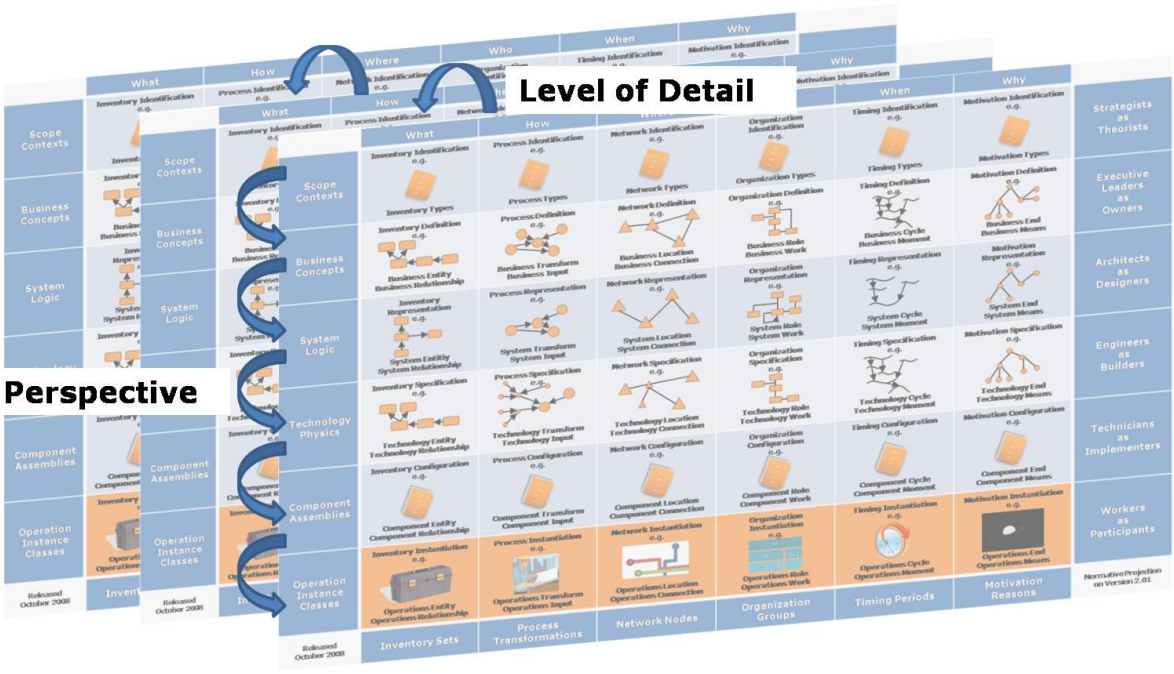

Figure 2: Zachman™-Framework – Perspective versus Level of Detail

There are relations between the perspectives. The perspective "Business Model" (row 2) defines requirements and preconditions for the perspective "System Logic" (row 3).

We speak about a composite model which contains all simple models in all perspectives, views and levels of details when we use the term "Enterprise Model". The "Business Model" includes all simple models of the perspective "Business Concepts". The German term "Fachmodell" usually includes models from the perspectives "Scope" and "Business Concepts". A "System Model" (for an IT system) contains simple models from the "System Logic" and "Technology Physics" perspectives.

Table 1 assigns common model types and artifacts to cells in the Zachman framework. The table gives only a rough idea which model types are used for a view usually. For a detailed discussion a precise definition of the model content is needed for each model type. Table 1 shows simple models only. It mixes models and artifacts from different levels of detail. It was not the intention to give examples for all cells.

	What	How	Where	Who	When	Why
Scope Contexts	List of Business Concepts	Process Map		Stakeholder model		Project Charter Vision Problem Statement
Business Concepts	Fact Model Rule Model	Business Process Model Business Activity Model	List of Business Locations	Organizational Chart Role Model (Business)	Timing Model	Business Requirements
System Logic	Information Model Logical Data Model	Functional Model Workflow Model Program Design	Deployment Model (logical) Component Model	Role Model (System)	Timing Model	Design Requirements
Technology Physics	Physical Data Model	Implementation Model	Deployment Model (physical)			System Requirements
Component Assemblies	Database Structure	Program Code				
Operations Instance Classes						

Table 1: Simple Models and Artifacts and the Zachman Framework

Recently you find the term "Capability" more often in publications. "Business Capabilities" and "System Capabilities" are differentiated. A capability means in the most cases a combination of simple models of the "What" and "How" views in the given perspective.

1.5 Model and Pattern

Wiktionary shows under number 6 another interesting definition: "A praiseworthy example to be copied, with or without modifications." More typically the term "Pattern" is used for this in the context of Enterprise Modeling.

The definition from Wiktionary reflects the intention quite well. We talk about pattern or templates which can be reused. In system design and implementation we know architecture patterns for IT systems, e.g. the model-view-controller pattern. Such patterns can be found in all perspectives and views of the Zachman Framework and also for composite models. For business process models a pattern "Follow-Up" can be imagined showing a generic, reusable part of a business process. The usage of patterns can significantly improve model quality.

2 Models and Communication

A good model serves a defined purpose – and in the best case exactly one purpose!

Models in the perspectives "System Logic" and "Technology Physics" can be used to generate source code for program or workflow engines automatically. You can generate Java source code from a program model represented by a UML Class Diagram, a physical data model represented by an ER Diagram can be used to generate database structures or a BPMN based process model can be used to generate BPEL code for a workflow engine. The model has to fulfill different formal requirements and must contain many detail information to make this possible. The discussion about visual models and standard notations seems mainly driven by this aspect: models as a basis to generate system components. Reality most often shows a different picture. Scott Ambler found in one of his surveys that models are NOT used to generate system components by the majority of project teams. The main purpose of the models is the documentation of requirements, system design, etc.[4]. Scott Amblers survey was focused on software development teams. I found the results for "agile" and "traditional" teams quite interesting.

It can be questioned if the survey is really representative. But my experiences show a very similar pattern in my customer base. This doesn't mean that generating system components from system models automatically (transfer from row 3 to row 4) is not of relevance. This question will gain momentum in the future with better techniques and with a wider availability of industrial components for different application areas.

Given this: What is the main intention of models today? Answers from customers vary from "system documentation", "system design", "process documentation", "process analysis and optimization" to "requirements management", "creating work orders" and "creating requests for proposals". Questioning these statements we realize that one important aspect is behind all these answers:

Models support Communication!

The first concern of all models in the perspective "Business Concepts" is the communication between business people. They also support communication between business people and IT stuff. For business models this will not change in the future – communication is the main purpose to support analysis, optimization, requirements definition, etc.

Many discussions change drastically when we accept that models are used for communication first in some perspectives. The "reader" or "addressee" of the model has to understand the model to make communication successful. The criteria "formal correctness" becomes less important. "Understandability" and "non ambiguity" are much more important.
For sure formal correctness is an issue here too to avoid ambiguity and to support the transformation into other models. But "understandability" is more important if we need to judge between "formal correctness" and "understandability". We maybe comment a model element to support a later formalization when necessary.

The question which model elements from the standard notations as UML or BPMN should be used is also changing direction: It is not important to use each element or to use every attribute the notation defines. Only elements an addressee of the model is able to understand are making sense. A good modeling style guide, consistent naming conventions, legends, or additional information for model elements are needed to support the reader of the model.

[4] Results from Scott Ambler's July 2008 Modeling and Documentation Survey posted at www.agilemodeling.com/surveys/

You will maybe also question your modeling tools. Which features are offered to support "communication"? Is it easy to add more information? Which output is created? This includes publishing models in an Intranet, creating work orders, standard reports or also legal documents as a Request for Proposal. Is it easy to gather information e.g. during a workshop? Are different media supported, e.g. audio information?

3 What next?

We need to go into more detail:

- Which simple models are relevant for a typical project?
- Which content do these models have exactly? Which perspective of the Zachman Framework is covered by these models?
- Which notation is best suited for each model?
- How are the models related? Which composite models do we need?
- Which quality criteria are relevant to support communication?

Also the tool question needs more attention. We will discuss this in further articles.