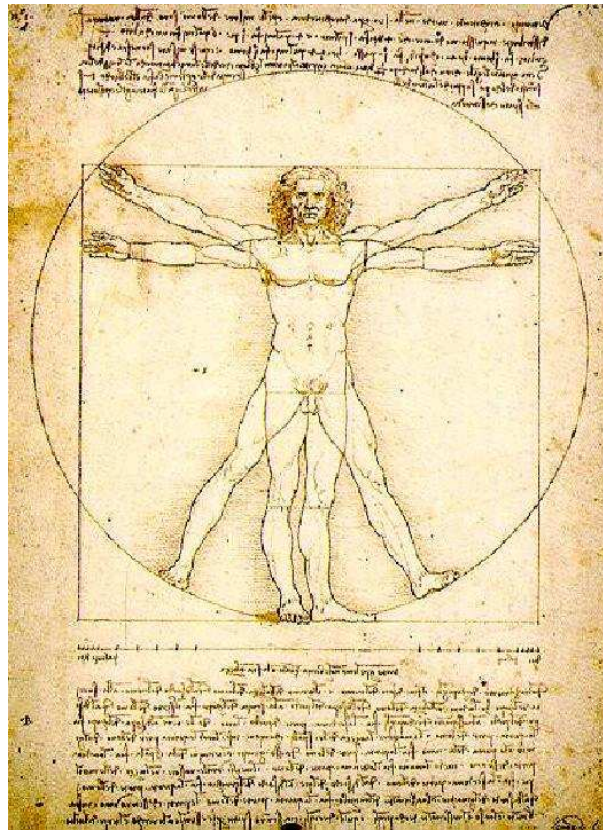


Work Systems Modeling Library

H.A. (Erik) Proper

Version of: 03-10-05

arXiv:2105.07419v1 [cs.SE] 16 May 2021



The Art & Science of Work Systems Engineering

Work Systems Modeling Library

H.A. (Erik) Proper

Contents

1	Introduction	7
2	Development Approach	11
3	WSML Domain Model	13
3.1	Documents	13
3.2	Characterizations	13
3.3	Method fragments	14
3.3.1	Modeling fragments	14
3.3.2	Models	16
3.3.3	Views	16
3.3.4	Model relations	16
4	Characterization dimensions	19
4.1	Why the model is produced	19
4.2	What the model is focussing on	20
4.3	How the models are produced	23
	Bibliography	25
	Dictionary	29
	Author Index	39
	Subject Index	41

Chapter 1

Introduction

Version:
06-09-05

As the title suggests, we are focussing on a library for the modeling of work systems. In line with [Alt99, Alt02] we regard a work system as:

Work system – An open active system in which actors perform processes using information, technologies, and other resources to produce products and/or services for internal or external actors.

In the context of these lecture notes this is refined to:

An open active system in which actors perform processes using information, technologies, and other resources to produce products and/or services for internal or external actors.

Information systems as well as organizations are regarded as specialisations of work systems. More precisely we define

Organizational system – A special kind of system, being normally active and open, and comprising the conception of how an organization is composed and how it operates (i.e. performing specific actions in pursuit of organizational goals, guided by organizational rules and informed by internal and external communication), where its systemic property are that it responds to (certain kinds of) changes caused by the system environment and, itself, causes (certain kinds of) changes in the system environment.

In the context of these lecture notes this is refined to:

A special kind of system, being normally active and open, and comprising the conception of how an organization is composed and how it operates (i.e. performing specific actions in pursuit of organizational goals, guided by organizational rules and informed by internal and external communication), where its systemic property are that it responds to (certain kinds of) changes caused by the system environment and, itself, causes (certain kinds of) changes in the system environment.

Information system – A sub-system of an organizational system, comprising the conception of how the communication and information-oriented aspects of an organization are composed and how these operate, thus leading to a description of the (explicit and/or implicit) communication-oriented and information-providing actions and arrangements existing within the organizational system.

In the context of these lecture notes this is refined to:

A sub-system of an organizational system, comprising the conception of how the communication and information-oriented aspects of an organization are composed and how these operate, thus leading to a description of the (explicit and/or implicit) communication-oriented and information-providing actions and arrangements existing within the organizational system.

Computerized information system – A sub-system of an information system, whereby all activities within that sub-system are performed by one or several computer(s).

In the context of these lecture notes this is refined to:

A sub-system of an information system, whereby all activities within that sub-system are performed by one or several computer(s).

Modeling of work systems occurs for all sorts of reasons. Requirements need to be expressed. A pre-existing situation may need to be charted and analyzed. Early design decisions may be captured using architecture principles. Detailed design may be worked out. We all regard these activities as essentially being forms of modeling. In the work systems modeling library, we consider work system engineering from a modeling perspective.

In the field of work system engineering, a whole plethora of modeling methods is available to system engineers and architects. Each of these methods can be used to model some (aspects) of a domain related to an existing and/or a planned work system. The aspects may refer to requirements, architecture, design, processing, data, etc, etc. In other words, these methods are essentially all intended to model different aspects of work systems and/or their context.

The aim of the work systems modeling library (WSML) is to bring together methodical knowledge concerning the modeling of work systems. This knowledge may, for example, pertain to:

- the syntax and/or semantics of modeling techniques to be used,
- guidelines or procedures describing how to produce the different models,
- frameworks defining different foci for viewing a work system,
- partial models (acting as pre-fabricated building blocks for more complex models) such as patterns and reference models,
- formal associations between models such as transformations (for example mapping from a conceptual database schemas to a relational database schemas), refinements (from high level design to detailed design), etc.

To refer to any of such ‘fragments’ of methodical knowledge, we shall use the term method fragment, which we define as:

Method fragment – An identifiable part of methodical knowledge.

In the context of these lecture notes this is refined to:

An identifiable part of methodical knowledge.

The WSML will be filled with method fragments pertaining to modeling of work systems.

The motivations behind the WSML are threefold:

- Our research group is involved in research into work system modeling in the broadest sense. Having a library of method fragments pertaining to the modeling of work systems, will aid us in our research activities. Even more, the high level structures for structuring the library, as well as the classification scheme to classify the available method fragments, are theoretical results in themselves.
- Having, and developing, a WSML will be a benefit to students. Both once the library is filled with method fragments, as well as during the actual filling and structuring of the library. As parts of assignments/projects in the context of courses and/or Bachelor/Master thesis projects, students can contribute to the creation and evolution of the library.
- Practitioners are confronted with a plethora of methods and techniques to model an even wider spectrum of distinct aspects of work systems.

Having a WSML would enable practitioners to better judge which method fragment to apply in a specific situation. The WSML should provide them with a roadmap to find their way around the jungle of methods, techniques, tools, etc.

The development of the WSML extends earlier work on similar libraries [PBHJ00, JPB⁺01]. Now that our theoretical perspective on work systems modeling has matured, and has become a core part of the DAVINCI series of courses and lecture notes [Pro05], we are able to build further on these initial results.

The notion of method fragment is borrowed from the field of method engineering [RB96, RP96]. This field of research certainly holds relevance to the WSML effort. However, method engineering aims to develop theories to (situationally) construct full fledged system engineering methods. We are 'merely' interested in the modeling aspects of system engineering, with the underlying goal of obtaining a fundamental insight into modeling as such.

In the remainder of this document, we will first discuss (chapter 2) our approach to the development on the WSML. We then continue (chapter 3) with a discussion of a domain model for the WSML which amongst other things will identify the types of method fragments that will be stored in the library. This is followed by the initial classification scheme (chapter 4) to classify method fragments. In subsequent versions of the WSML we expect these latter two to evolve. The version as presented in this document is based on several sources from literature and some experiences from practice. It will serve as a starting point to codify/classify pre-existing modeling methods and techniques. In performing the latter exercise, the classification scheme is likely to evolve.

Chapter 2

Development Approach

Version:
07-09-05

Considering the experimental nature of the WSML and the uncertainty about the precise structures of a classification scheme for the method fragments, it is most sensible to take an iterative approach in the development of the above four elements of the library. Even more, in the context of several courses in the DAVINCI series as well as thesis projects, students have already done partial classifications of modeling elements. The WSML project aims to incorporate these results from the past as well.

The basic idea of the WSML project is to iteratively create a library of method fragments. In creating this library, the following results need to be created and iterated:

Typing of method fragments – An overall typing of the kind of method fragments which we will discern.

The structure of method fragments will be represented as an ORM [Hal01] schema together with an explanation of the types involved.

Classification scheme – A classification scheme which can be used to classify the method fragments.

This classification scheme will be represented as a refinement of the previous ORM schema, together with an explanation of the identified classification dimensions.

Classification procedure – When adding elements to a library, the classification should be done in a *reproducible* way. This requires the explicit identification of a procedure that should be followed when identifying method fragments and consequently classifying a given method fragment.

Library infrastructure – A database system based on the classification schema, that can be used to actually store the classification of the method fragments. This could be an automated system, but initially, this can quite well be a manual (e.g. one document per fragment) system.

Library content – This is the actual library of classifications of modeling elements.

In the development of an initial version the WSML we identify four main iterations:

- 0 As a first step a provisional classification scheme should be provided and agreed upon. This document contains the proposed classification scheme.
In addition pre-existing classifications/descriptions of relevant method fragments should be gathered.
- 1 The pre-existing classifications should be enriched/transformed to match the new classification scheme, and possibly modifications should be made to the classification scheme based on the classification schemes used in the pre-existing classifications.

- 2 Additional method fragments should be classified and a proper library infrastructure should be set up, which is to be filled with the classifications.
- 3 More method fragments elements should be classified and the classification scheme should be validated against selection criteria as used in practical modeling situation.

In the further development of the WSML we will continue to rely on student projects. However, we will also seek collaborations with industrial partners, as well as organizations such as the NGI (Netherlands Society for Informatics), GIA (Society for Information Architects) and the NAF (Netherlands Architecture Forum).

Chapter 3

WSML Domain Model

Version:
03-10-05

In this chapter we discuss the domain model of the WSML. The complete model is depicted in Figure 3.1. Below we provide a textual discussion of the core concepts as identified in the model. Using sans-serif font we will refer to object types in the model depicted in Figure 3.1.

3.1 Documents

The library not only comprises facts about method fragments, but (in particular in the earlier iterations) comprises documents supporting these facts. These Library Source Documents are Publications from the IRIS publication management system, which may have a PDF formatted content and which *must* have their Publication Details specified as a BiBTeX entry. Publications may refer to other documents. We distinguish between two classes of Library Source Documents:

1. The Base Documents are original documents discussing one or more method fragments and/or characterisation dimensions, such as Object-Role Modeling (ORM) [Hal01], the Zachman framework [Zac87], Unified Modeling Language (UML) [BRJ99], etc.
2. The Description Documents are descriptions of Library Fragments. As time progresses, the description of a Library Fragment may be updated. A newer Description Document can therefore be preceded by an older one.

Initially, the library will be filled mainly with description documents only (potentially even violating some of the constraints in Figure 3.1). As time goes on, the characterizations of the method fragments will be more and more explicit, which can then be stored in the library explicitly conform the structures as described below.

In the library we identify two main classes of Library Fragments. The first class deals with Characterization Dimensions, which are used to characterize the actual Method Fragments. The latter ones form the other class of Library Fragments. Below, we discuss these two classes in more detail, including a further classification of Method Fragments.

3.2 Characterizations

Method Fragments in the library must be characterized by means of Characterization Properties, which are defined as:

Characterization property – A specific value from a characterization dimension that an artefact may exhibit.

In the context of these lecture notes this is refined to:

A specific value from a characterization dimension that an artefact may exhibit.

These characterization properties, which will be discussed in the next chapter, may refer to such things as the system aspect (business, information, infrastructure, etc.) which a method fragment is geared towards, the kind (prescriptive or descriptive) of model it aims to produce, etc.

Each Characterization Property may be associated to a Method Fragment as either being: intended for or suitable for. One would expect the set of properties for which a fragment is intended for to be a subset of the properties it is suitable for, but that does not necessarily have to be the case. In each case, a Motivation for the property must be provided. A Motivation may refer to publications (by means of a BiBTeXKey) substantiating this motivation.

A Characterization Property really consists of two parts: a Characterization Dimension and a Characterization Value. When characterizing method fragments, a well defined method should be used in determining the characterization values. This is described in the Characterization Method. The textual description may refer to other documents by means of a BiBTeXKey. Each Characterization Dimension must be described by some Dimension Description Document

Note: What is currently missing from the domain model is the fact that once a larger set of Characterization Dimensions becomes available we can identify for which types of Method Fragments they are relevant. As a next step we can then require a characterization of fragments for this set of dimensions to be mandatory. This will lead to the introduction of a fact-type:

Characterization dimension (name) is mandatory for Fragment type (name)

with constraints:

EACH Characterization dimension MUST BE mandatory for SOME Fragment type

EACH Fragment type IS ONE OF: Viewing framework, Viewpoint, ...

EACH Method Fragment of type X is characterized by EACH Characterization dimension which is mandatory for type X

3.3 Method fragments

Each Method Fragment must be described by some Fragment Description Document, and must receive some Characterization Property defining what the intended/suitable focus of the fragment is. We distinguish four main classes of Method Fragments:

1. Modeling Fragments which are the modeling techniques, frameworks, viewpoints, types of mappings/relations between models, etc, used in the creation of actual models.
2. Models representing complete models, cases, reference models, patterns, etc.
3. Views comprising sets of models, representing complete cases, reference models, patterns, etc.
4. Model Relations between Partial Models present in the library using one of the identified Model Relation Types.

Each of these classes is discussed in more detail in the remainder of this section.

3.3.1 Modeling fragments

In [SWS89, WH90] a system development method is dissected into a way of thinking, way of modeling, way of working, way of controlling and a way of supporting. By combining this terminology with the notion of a viewpoint taken from [IEE00], and adding viewing framework, viewing cell and model relation type, we define the notion of a modeling fragment, and its constituents, as follows:

Modeling fragment – A method fragment that is concerned with the actual modeling of a domain. Three classes of method fragments are identified: viewing framework, way of working, viewpoint, model relation type. Each method fragment must have a unique underlying way of thinking.

In the context of these lecture notes this is refined to:

A method fragment that is concerned with the actual modeling of a domain. Three classes of method fragments are identified: viewing framework, way of working, viewpoint, model relation type. Each method fragment must have a unique underlying way of thinking.

Way of thinking – Articulates the assumptions on the kinds of problem domains, solutions, engineers, analysts, etc. This notion is also referred to as *die Weltanschauung* [Sol83, WAA85], *underlying perspective* [Mat81] or *philosophy* [Avi95].

In the context of these lecture notes this is refined to:

Articulates the assumptions on the kinds of problem domains, solutions, engineers, analysts, etc. This notion is also referred to as *die Weltanschauung* [Sol83, WAA85], *underlying perspective* [Mat81] or *philosophy* [Avi95].

Viewing framework – A kind of modeling fragment, which identifies a set of related perspectives from which to view a system. A viewing framework may be composed of yet other viewing frameworks.

In the context of these lecture notes this is refined to:

A kind of modeling fragment, which identifies a set of related perspectives from which to view a system. A viewing framework may be composed of yet other viewing frameworks.

Viewing cell – A viewing framework that is not decomposed into a (covering) set of smaller viewing frameworks.

In the context of these lecture notes this is refined to:

A viewing framework that is not decomposed into a (covering) set of smaller viewing frameworks.

Way of working – Structures (parts of) the way in which a system is engineered. It defines the possible tasks, including sub-tasks, and ordering of tasks, to be performed as part of the development process. It furthermore provides guidelines and suggestions (heuristics) on how these tasks should be performed.

In the context of these lecture notes this is refined to:

Structures (parts of) the way in which a system is engineered. It defines the possible tasks, including sub-tasks, and ordering of tasks, to be performed as part of the development process. It furthermore provides guidelines and suggestions (heuristics) on how these tasks should be performed.

Viewpoint – A specification of the conventions for constructing and using views.

This involves: a way of thinking, a way of modeling, a way of communicating, a way of working, a way of supporting and a way of using

In the context of these lecture notes this is refined to:

A specification of the conventions for constructing and using views.

This involves: a way of thinking, a way of modeling, a way of communicating, a way of working, a way of supporting and a way of using

Model relation type – A type of well-defined relations between models. An example would be a mapping algorithm between a conceptual model and a relational database schema, the relation between the concepts of a business process model and the concepts of a supporting work-flow, etc.

In the context of these lecture notes this is refined to:

A type of well-defined relations between models. An example would be a mapping algorithm between a conceptual model and a relational database schema, the relation between the concepts of a business process model and the concepts of a supporting work-flow, etc.

Way of modeling – Identifies the *core concepts* of the language that may be used to denote, analyze, visualize and/or animate system descriptions.

In the context of these lecture notes this is refined to:

Identifies the *core concepts* of the language that may be used to denote, analyze, visualize and/or animate system descriptions.

Way of conceiving – A set of modeling concepts by which viewers are to observe domains. This usually takes the form of a meta model.

In the context of these lecture notes this is refined to:

A set of modeling concepts by which viewers are to observe domains. This usually takes the form of a meta model.

Way of describing – The medium and ‘notations’ used to represent the concepts as identified in a way of conceiving. It describes how the abstract concepts from the way of conceiving are communicated to human beings, for example in terms of a textual or a graphical notation.

Note that it may very well be the case that different modeling techniques are based on the same way of conceiving, yet use different notations.

In the context of these lecture notes this is refined to:

The medium and ‘notations’ used to represent the concepts as identified in a way of conceiving. It describes how the abstract concepts from the way of conceiving are communicated to human beings, for example in terms of a textual or a graphical notation.

Note that it may very well be the case that different modeling techniques are based on the same way of conceiving, yet use different notations.

Note: as a synonym for ‘way of working’ we will also use the term approach and as synonym for ‘way of modeling’ we will also use the term technique.

3.3.2 Models

The second class of method fragments deals with Models. Some may be partial models in the sense that they deal with such things as reference models, reference architectures, patterns, partial solutions, etc. All of these latter these models can be regarded as incomplete in the sense that they all require some form of *applying* to make them usefull in a specific situation. They cannot be used directly, but need tuning, extending, or any other form of *application* to make them fit a situation at hand.

At present, this class of method fragments is not yet worked out in full detail, as also shown in Figure 3.1. Further research is indeed needed to ascertain what concepts are involved in Partial Models, their representation and their application/applicability in specific situations.

3.3.3 Views

A Viewpoint comprises one or more Techniques. A View essentially is a set of Models describing the same domain in terms of the Techniques associated to the Viewpoint used in creating the View. Similarly to Models, Views can be partial as well. If a Model is signified as being a partial model, then any View containing this Model must also be a partial view.

3.3.4 Model relations

The Model Relation Types which define relations between models in general can be used to identify specific relations between Partial Models. This leads to the notion of a Model Relation, which runs

from a partial model to another one, and is an instance of some Model Relation Type. Similarly to Partial Models, the notion of Model Relation will definitely need further refinements in future versions of the WSML.

Chapter 4

Characterization dimensions

In this chapter we provide an initial overview of characterization dimensions based on existing literature. During the actual populating of the WSML, these dimensions will need to be described in more detail, in particular the Characterization Method. Below, we provide a first set of classification dimensions. Note that the set of dimensions used in the WSML should be fairly ‘orthogonal’.

The field of method engineering [RB96, RP96] has already done some work on the classification of (fragments of) methods. This work needs to be taken into account when defining the characterization dimension. It should be noted, however, that our focus is on domain modeling methods and not so much on methods for work systems engineering in general. Even more, our goal of studying modeling methods is to improve our understanding of the act of modeling as such. In other words, we are less interested in the construction of methods, but more in understanding the inner workings of *modeling* methods.

At present we identify four classes of characterization dimensions:

- Why the model is produced.
- What the model is focussing on.
- How the model is produced.

Each of these three classes are discussed in detail below. The current set is based on [TC93, FV99, Pro01, JPB⁺01, HP02, GKV03, Pro04, Lo05, PHV05].

4.1 Why the model is produced

Modeling purpose – The purpose for which the models produced by the method are suitable for. Based on [Lo05] we shall use as sub-classifiers:

Designing – A model which aims to support designers in the design process, from initial sketch, via limitations of the design space to the detailed design.

Deciding – A model which aims to support decision makers in their decision making process by offering insight into the design of a domain and the possible impact of design decisions.

Informing – A model which aims to inform stakeholders about the domain being modeled. This is usually done in order to achieve understanding, obtain commitment, etc.

Design chain – The focus with regard to the design from *why it should be* and *what should be* to *how it ought to be* and *how it will be*. In other words, the question which the model aims to answer about a given work system. Specific methods may produce models that are geared towards a specific phase in the design chain. Based on [PHV05] we shall use the following sub-classifiers:

System purpose – A model focussing on *why* a specific work system is needed.

System functionality – A model focussing on *what* functionality a work system should provide to its environment.

System design – A model representing *how* the functionality will be realized.

System quality – A model identifying *how well* a work system should realize its functionality.

System costs – A model defining *what it may cost* for the work system to be constructed and deliver the desired functionality to its environment.

Intended audience – The audience the model is produced for. Inspired by [GKV03] we identify the following sub-classifiers:

Actor in future system – A (human!) actor in a future system.

Sponsor – A sponsor of the future system and/or its development.

Designer – Designers of the system.

Analyst – Analysts of (parts of) the domain in which the system will operate.

Engineer – The engineers of the actual system.

4.2 What the model is focussing on

Semantic force – The semantic force of the model refers to the fact whether it is a *prescriptive* or *descriptive* nature, leading to sub-classifiers:

Prescriptive – For models with a purely prescriptive nature. These are models which limit the design freedom of future designers who are bound to make a design *conforming to* the earlier pre-scriptive model.

Descriptive – For models with a purely descriptive nature.

Mixed – For models with a mix between prescriptive and descriptive nature.

Nature of the information – This refers to the nature of the content. Based on [GKV03] we identify the following sub-classifiers:

Policy – Policy statements pertaining to the system.

Principles – Principles to which the (design of) the system should adhere.

Guidelines – Guidelines (operationalisations of the principles) which should be met by the future system.

Descriptions – Descriptions of what the (future) system (will) look(s) like.

Standards – Standards (to be) used in the creation/design of the system.

Type of information – This refers to the kind of information that may be contained in the model. Inspired by e.g. [TC93, Lo05], typical sub-classifiers would be:

Business – Business models, markets, products, etc.

Organization – Work processes, culture, organizational structures, skills, etc.

Information – Domains of information/knowledge needed for the business activities.

Application – Automation of work.

Infrastructure – Underlying technological infrastructure.

Most of the “architecture frameworks” provide different sub-classifications for these classes of information.

Systemic scope – The scope of the domain that is modeled by the model. We identify three sub-classifiers:

Use-case – The scope of a specific use-case of the work system.

System component – A distinct component of a work system.

System – The entire work system and its direct environment is the domain that is to be modeled.

System of Systems – The scope is a set of work systems, in other words, a system of work systems.

Temporal scope – The temporal scope at which we regard domain. Sub-classifiers are:

Operational – The work system as it is currently (or in the near future) operational.

Tactical – The work system as it is ideally operational after the execution of a development projects.

Strategical – The work system as it is ideally operational after the execution of a number (a program) of development projects.

Implementation abstraction – The level of abstraction from underlying information technology. Based on the distinction from MDA [], we define the following sub-classifiers:

Computing independent – A model which is independent of decisions regarding computerisation of certain activities.

Platform independent model – A model in which choices for computerisation of tasks has indeed been made, but does not yet make a choice for a specific technological platform.

Platform specific – A model which is tied to a specific technological platform. A program in a programming language such as C or Java would be an ultimate example of a platform specific model.

Systemic aggregation – The level of aggregation that is used in a model. Systemic aggregation is usually used as a means to “hide” information about specificities of a work system. It allows modelers (and viewers of the models) to focus on higher level issues. Based on [Lo05] we identify the following sub-classifiers:

Detailed level – All (relevant) details are shown.

Coherence level – The model focuses on the coherence (between different aspects) of the modeled domain.

Overview level – The model provides an overview of the key issues of the modeled domain.

System qualities – Models may focus on different quality properties of systems. In [ISO01, ISO96] the following key quality attributes have been identified:

Efficiency – Is the relationship between the level of performance of the system (or a sub-system) and the amount of resources used, under stated conditions. Efficiency may be related to time behaviour (response time, processing time, throughput rates) and to resource behaviour (amount of resources used and duration of such use).

In the context of these lecture notes this is refined to:

Is the relationship between the level of performance of the system (or a sub-system) and the amount of resources used, under stated conditions. Efficiency may be related to time behaviour (response time, processing time, throughput rates) and to resource behaviour (amount of resources used and duration of such use).

Functionality – Bears on the existence of a set of functions and their specified properties. The functions are those that satisfy stated or implied needs.

Sub-characteristics of functionality are: suitability, accuracy, interoperability, compliance, security and traceability.

In the context of these lecture notes this is refined to:

Bears on the existence of a set of functions and their specified properties. The functions are those that satisfy stated or implied needs.

Sub-characteristics of functionality are: suitability, accuracy, interoperability, compliance, security and traceability.

Reliability – Is the capability of the system (or a sub-system) to maintain its level of performance under stated conditions for a stated period of time. The mean time to failure metric is often used to assess reliability of systems. The reliability can be determined through defining the level of protection against failures and the necessary measures for recovery from failures.

Sub-characteristics of reliability are: maturity, fault tolerance, recoverability, availability and degradability.

In the context of these lecture notes this is refined to:

Is the capability of the system (or a sub-system) to maintain its level of performance under stated conditions for a stated period of time. The mean time to failure metric is often used to assess reliability of systems. The reliability can be determined through defining the level of protection against failures and the necessary measures for recovery from failures.

Sub-characteristics of reliability are: maturity, fault tolerance, recoverability, availability and degradability.

Maintainability – Bears on the effort needed to make specified modifications to the system (or a sub-system). Modification may include corrections, improvements or adaptation to changes in the environment, the requirements and the (higher levels of the) design.

Sub-characteristics of maintainability are: analysability, changeability, stability, testability, manageability, reusability.

In the context of these lecture notes this is refined to:

Bears on the effort needed to make specified modifications to the system (or a sub-system). Modification may include corrections, improvements or adaptation to changes in the environment, the requirements and the (higher levels of the) design.

Sub-characteristics of maintainability are: analysability, changeability, stability, testability, manageability, reusability.

Portability – Is the ability of the system (or one of its components) to be transferred from one environment to another.

Sub-characteristics of portability are: adaptability, installability, conformance, replaceability.

In the context of these lecture notes this is refined to:

Is the ability of the system (or one of its components) to be transferred from one environment to another.

Sub-characteristics of portability are: adaptability, installability, conformance, replaceability.

Usability – Bears on the effort needed for the use of the system (or one of its sub-systems) by the actors in the environment of the system.

Sub-characteristics of usability are: understandability, learnability, operability, explicitness, customisability, attractivity, clarity, helpfulness and user-friendliness.

In the context of these lecture notes this is refined to:

Bears on the effort needed for the use of the system (or one of its sub-systems) by the actors in the environment of the system.

Sub-characteristics of usability are: understandability, learnability, operability, explicitness, customisability, attractivity, clarity, helpfulness and user-friendliness.

System realization – The level of realization of the services provided by a system to its environment. The underlying way of thinking is that a work system is a *supporting system*, which provides functionality to a *using system*, while making use of *infrastructure system(s)*. Sub-classifiers are therefore:

Using system – The way the environment will use the work system.

Supporting system – The way the work system will provide functionality/services to the environment.

Infrastructure system – The infrastructural facilities that are used by the work system in order to provide the functionality/services to the environment.

Actor kinds – What kind of actors *in the system* does the modeling focus on? Sub-classifiers are:

Heterogenous – Heterogenous (typically composed) actors.

Human – The role of human beings in a work system.

Computerised intelligence – The role of software agents/components that aim to show intelligent behavior.

Computerised – The role of “traditional” software components.

Explicitness of represented knowledge – Models produced during system development are essentially forms of *explicit knowledge* pertaining to an existing/future system; its design, the development process by which it was/is to be created, the underlying considerations, etc.

Based on [FV99, Pro01], the following dimensions of explicitness for representations of system development knowledge (pertaining to both target domain and project domain knowledge) can be identified:

Level of formality – The degree of formality indicates the type of language used to represent the knowledge. Such a language could be formal, in other words a language with an underlying well-defined semantics in some mathematical domain, or it could be informal –not mathematically underpinned, typically natural language, graphical illustrations, animations, etc.

Level of quantifiability – Different aspects of the designed artefact, be it (part of) the target or the project domain, may be quantified. Quantification may be expressed in terms of volume, capacity, workload, effort, resource, usage, time, duration, frequency, etc.

Level of executability – The represented knowledge may, where it concerns artefacts with operational behaviour, be explicit enough so as to allow for execution. This execution may take the form of a simulation, a prototype, generated animations, or even fully operational behaviour based on executable specifications.

Level of comprehensibility – The knowledge representation may not be comprehensible to the intended audience. Tuning the required level of comprehensibility of the representation, in particular the representation language used, is crucial for effective communication. The representation language may offer special constructs to increase comprehension, such as stepwise refinements, grouping/clustering of topically related items/statements, etc.

Level of completeness – The knowledge representation may be complete, incomplete, or overcomplete with regards to the knowledge topic (see previous subsection) it intends to cover.

Note: models may not only focus on one of the areas identified in the sub-classifiers, they are also likely to identify links between the different areas, leading to the vertical/horizontal integration of models with a homogeneous focus. This will need some refinement of the classification scheme.

4.3 How the models are produced

Based on [FV99, Pro01], we currently identify:

Cognitive approach – The cognitive approach used during the analysis phase represents the way in which information is processed to make design decisions in the migration project. Two options are distinguished:

Analytical approach – When information is processed analytically, the available information is simplified through abstraction in order to reach a deeper and more invariant understanding. An analytical approach is used to handle the complexity of the information. In an analytical approach the information system is mainly described by use of some degree of formality.

Experimental approach – When using an experimental approach the project actors learn from doing experiments. The purpose is to reduce uncertainties by generating more information. Experiments can, for example, be based on prototypes, mock-ups, benchmark test of migrated components or other kinds of techniques which make the results of migration scenarios visible.

The two cognitive approaches may be combined. An analytical approach to reduce complexity may introduce new sources of uncertainty requiring experimental actions. Conversely, an experimental approach to reduce uncertainty may introduce new sources of complexity requiring analytical actions.

Social approach – The social approach is the way in which project actors work together with the business actors during the analysis phase. Two options are distinguished:

Expert-driven – In an expert-driven approach, project actors (the experts) will produce descriptions on the basis of their own expertise, and interviews and observations of business actors. The descriptions can then be delivered to the business actors for remarks or approvals.

Participatory – In a participatory approach, the project actors produce the descriptions in close co-operation with some or all the business actors, e.g. in workshops with presentations, discussions and design decisions. A participatory approach may allow the acquisition of knowledge, the refinement of requirements and the facilitation of organisational change.

Bibliography

- [Alt99] S. Alter. A general, yet useful theory of information systems. *Communications of the Association for Information Systems*, 1(13), 1999.
<http://cais.isworld.org/articles/1-13/default.asp>
- [Alt02] S. Alter. The work system method for understanding information systems and information system research. *Communications of the Association for Information Systems*, 9(9):90–104, 2002.
<http://cais.isworld.org/articles/default.asp?vol=9&art=6>
- [Avi95] D.E. Avison. *Information Systems Development: Methodologies, Techniques and Tools*. McGraw-Hill, New York, New York, USA, 2nd edition, 1995. ISBN 0077092333
- [BRJ99] G. Booch, J. Rumbaugh, and I. Jacobson. *The Unified Modelling Language User Guide*. Addison Wesley, Reading, Massachusetts, USA, 1999. ISBN 0201571684
- [FV99] M. Franckson and T.F. Verhoef, editors. *Specifying Deliverables*. Information Services Procurement Library. ten Hagen & Stam, Den Haag, The Netherlands, EU, 1999. ISBN 9076304823
- [GKV03] D. Greefhorst, H. Koning, and H. van Vliet. Dimensies in raamwerken. *Informatie*, 45(11):22–27, 2003. In Dutch.
- [Hal01] T.A. Halpin. *Information Modeling and Relational Databases, From Conceptual Analysis to Logical Design*. Morgan Kaufmann, San Mateo, California, USA, 2001. ISBN 1558606726
- [HP02] W.-J. van den Heuvel and H.A. (Erik) Proper. De pragmatiek van Architectuur. *Informatie*, 44(11):12–14, 2002. In Dutch.
- [IEE00] Recommended Practice for Architectural Description of Software Intensive Systems. Technical Report IEEE P1471–2000, The Architecture Working Group of the Software Engineering Committee, Standards Department, IEEE, Piscataway, New Jersey, USA, September 2000. ISBN 0738125180
<http://www.ieee.org>
- [ISO96] ISO. *Kwaliteit van softwareproducten*. ten Hagen & Stam, Den Haag, The Netherlands, EU, 1996. In Dutch. ISBN 9026724306
- [ISO01] *Software engineering – Product quality – Part 1: Quality model*, 2001. ISO/IEC 9126–1:2001.
<http://www.iso.org>
- [JPB⁺01] R.D.T. Janssen, H.A. (Erik) Proper, H. Bosma, D. Verhoef, and S.J.B.A. Hoppenbrouwers. Developing an Architecture Method Library. Technical report, Ordina Institute, Gouda, The Netherlands, EU, January 2001.

- [Ken84] F. Kensing. Towards Evaluation of Methods for Property Determination: A Framework and a Critique of the Yourdon–DeMarco Approach. In T.M.A. Bemelmans, editor, *Beyond Productivity: Information Systems Development for Organizational Effectiveness*, Amsterdam, The Netherlands, EU, pages 325–338. North–Holland, Amsterdam, The Netherlands, EU, 1984.
- [Lo05] M.M. Lankhorst and others. *Enterprise Architecture at Work: Modelling, Communication and Analysis*. Springer, Berlin, Germany, EU, 2005. ISBN 3540243712
- [Mat81] L. Mathiassen. *Systemudvikling og Systemudviklings–Metode*. PhD thesis, Aarhus University, Aarhus, Denmark, EU, 1981. In Danish.
- [McC89] C.L. McClure. *CASE is Software Automation*. Prentice–Hall, Englewood Cliffs, New Jersey, USA, 1989. ISBN 0131193309
- [PBHJ00] H.A. (Erik) Proper, H. Bosma, S.J.B.A. Hoppenbrouwers, and R.D.T. Janssen. Towards an Information Systems Engineering Body of Knowledge. In D.B.B. Rijsenbrij, editor, *Proceedings of the Second National Architecture Congress*, November 2000.
- [PHV05] H.A. (Erik) Proper, S.J.B.A. Hoppenbrouwers, and G.E. Veldhuijzen van Zanten. Communication of Enterprise Architectures. In M.M. Lankhorst, editor, *Enterprise Architecture at Work: Modelling, Communication and Analysis*, Berlin, Germany, EU, pages 67–82. Springer, Berlin, Germany, EU, 2005. ISBN 3540243712
- [Pro01] H.A. (Erik) Proper, editor. *ISP for Large–scale Migrations*. Information Services Procurement Library. ten Hagen & Stam, Den Haag, The Netherlands, EU, 2001. ISBN 9076304882
- [Pro04] H.A. (Erik) Proper. *Architecture–driven Information Systems Engineering*. DaVinci Series. Nijmegen Institute for Information and Computing Sciences, University of Nijmegen, Nijmegen, The Netherlands, EU, 2004.
- [Pro05] H.A. (Erik) Proper. *An Overview of the DaVinci Series*. DaVinci Series. Nijmegen Institute for Information and Computing Sciences, University of Nijmegen, Nijmegen, The Netherlands, EU, 2005.
- [RB96] M. Rossi and S. Brinkkemper. Complexity Metrics for System Development Methods and Techniques. *Information Systems*, 21(2):209–227, 1996.
- [RP96] C. Rolland and N. Prakash. A Proposal for Context–Specific Method Engineering. In S. Brinkkemper, K. Lytinen, and R.J. Welke, editors, *Proceedings of the IFIP TC8 WG8.1/8.2 Working Conference on Method Engineering*, pages 191–208, London, United Kingdom, EU, August 1996. Chapman & Hall.
- [Sol83] H.G. Sol. A Feature Analysis of Information Systems Design Methodologies: Methodological Considerations. In T.W. Olle, H.G. Sol, and C.J. Tully, editors, *Information Systems Design Methodologies: A Feature Analysis*, Amsterdam, The Netherlands, EU, pages 1–7. North–Holland/IFIP WG8.1, Amsterdam, The Netherlands, EU, 1983. ISBN 0444867058
- [Sol88] H.G. Sol. Information Systems Development: A Problem Solving Approach. In *Proceedings of 1988 INTEC Symposium Systems Analysis and Design: A Research Strategy*, 1988.
- [SWS89] P.S. Seligmann, G.M. Wijers, and H.G. Sol. Analyzing the Structure of I.S. Methodologies, an alternative approach. In R. Maes, editor, *Proceedings of the First Dutch Conference on Information Systems*, 1989.
- [TC93] D. Tapscott and A. Caston. *Paradigm Shift – The New Promise of Information Technology*. McGraw–Hill, New York, New York, USA, 1993. ASIN 0070628572

- [WAA85] A.T. Wood–Harper, L. Antill, and D.E. Avison. *Information Systems Definition: The Multiview Approach*. Blackwell, Oxford, United Kingdom, EU, 1985. ISBN 0632012168
- [WH90] G.M. Wijers and H. Heijes. Automated Support of the Modelling Process: A view based on experiments with expert information engineers. In B. Steinholz, A. Sølvsberg, and L. Bergman, editors, *Proceedings of the Second Nordic Conference CAiSE'90 on Advanced Information Systems Engineering, Stockholm, Sweden, EU*, volume 436 of *Lecture Notes in Computer Science*, pages 88–108, Berlin, Germany, EU, 1990. Springer. ISBN 3540526250
- [Zac87] J.A. Zachman. A framework for information systems architecture. *IBM Systems Journal*, 26(3), 1987.

Dictionary

WSML – Abbreviation for **work system modeling** library. A library of methodical **knowledge** pertaining to the **modeling** of **work systems**.

In the context of these lecture notes this is refined to:

Abbreviation for **work system modeling** library. A library of methodical **knowledge** pertaining to the **modeling** of **work systems**.

Active system – A special kind of **system** that is conceived of as begin able to change parts of the **universe**.

In the context of these lecture notes this is refined to:

A special kind of **system** that is conceived of as begin able to change parts of the **universe**.

Activity participation – A **system link** between a **system activity** and one of its **actor**.

In the context of these lecture notes this is refined to:

A **system link** between a **system activity** and one of its **actor**.

Actor – A **system element** that is conceived of as having some involvement in a **system activity**. This involvement is a special kind of **system link**, referred to as an **activity participation**.

In the context of these lecture notes this is refined to:

A **system element** that is conceived of as having some involvement in a **system activity**. This involvement is a special kind of **system link**, referred to as an **activity participation**.

Approach – Is a synonym for: **way of working**.

In the context of these lecture notes this is refined to:

Is a synonym for: **way of working**.

Architectural description – A [**system-description-ls**] documenting/describing an **architecture**.

In the context of these lecture notes this is refined to:

A [**system-description-ls**] documenting/describing an **architecture**.

Architecture principle – A **design principle** which is used to direct the design/architecting of an **architecture**.

In the context of these lecture notes this is refined to:

A **design principle** which is used to direct the design/architecting of an **architecture**.

Architecture – A **model** of which the [**system-description-ls**], the so-called **architectural description**, is used during **system engineering** to:

- express the fundamental organization of the **system domain** in terms of **components**, their relationships to each other and to the **environment** and
- the principles guiding its evolution and design,

and which's explicit intend is to be used as a means:

- of communication & negotiation among stakeholders,
- to evaluate and compare design alternatives,
- to plan, manage, and execute further system development,

- to verify the compliance of a system implementation's.

In the context of these lecture notes this is refined to:

A **model** of which the [system-description-Is], the so-called **architectural description**, is used during **system engineering** to:

- express the fundamental organization of the **system domain** in terms of **components**, their relationships to each other and to the **environment** and
- the principles guiding its evolution and design,

and which's explicit intend is to be used as a means:

- of communication & negotiation among stakeholders,
- to evaluate and compare design alternatives,
- to plan, manage, and execute further system development,
- to verify the compliance of a system implementation's.

Characterization dimension – A set of (possibly ordered) values by which an artefact may be characterized.

In the context of these lecture notes this is refined to:

A set of (possibly ordered) values by which an artefact may be characterized.

Component system – A component-system S' of a **system** S , is a **sub-system**, where the set of **model concepts** in S' is a proper subset of the set of entities in S .

In the context of these lecture notes this is refined to:

A component-system S' of a **system** S , is a **sub-system**, where the set of **model concepts** in S' is a proper subset of the set of entities in S .

Component – An abbreviation of: **component system**

In the context of these lecture notes this is refined to:

An abbreviation of: **component system**

Conception – That what results, in the mind of a **viewer**, when they interpret a **perception** of a **domain**.

In the context of these lecture notes this is refined to:

That what results, in the mind of a **viewer**, when they interpret a **perception** of a **domain**.

Concept – Any **element** from a **conception** that is not a **links**.

In the context of these lecture notes this is refined to:

Any **element** from a **conception** that is not a **links**.

Data – Any representation in some language. Data is therefore simply a collection of symbols that may, or may not, have some meaning to some **actor**.

In the context of these lecture notes this is refined to:

Any representation in some language. Data is therefore simply a collection of symbols that may, or may not, have some meaning to some **actor**.

Description – The result of a **viewer** denoting a **conception**, using some language to express themselves.

In the context of these lecture notes this is refined to:

The result of a **viewer** denoting a **conception**, using some language to express themselves.

Design principle – A system description language, which is intended to aid in the transition from requirements to design.

Each principle is a predicate over the products of the design process, where a system is no longer treated as a black-box. It thus provides direction to the ensuing design process.

In the context of these lecture notes this is refined to:

A system description language, which is intended to aid in the transition from requirements to design.

Each principle is a predicate over the products of the design process, where a system is no longer treated as a black-box. It thus provides direction to the ensuing design process.

Domain – Any ‘part’ or ‘aspect’ of the **universe** a **viewer** may have an **interest** in.

In the context of these lecture notes this is refined to:

Any ‘part’ or ‘aspect’ of the **universe** a **viewer** may have an **interest** in.

Dynamic system – A special kind of **system** that is conceived of as undergoing change in the cause of time.

In the context of these lecture notes this is refined to:

A special kind of **system** that is conceived of as undergoing change in the cause of time.

Element – The elementary parts of a **viewer’s conception**.

In the context of these lecture notes this is refined to:

The elementary parts of a **viewer’s conception**.

Environment – The environment of a **domain** is that part of a **viewer’s conception** of a **universe**, which has a direct **link** to the **domain**.

In the context of these lecture notes this is refined to:

The environment of a **domain** is that part of a **viewer’s conception** of a **universe**, which has a direct **link** to the **domain**.

Human actors – An **actor** which is a single human being, or essentially a set of human-beings, such as a team.

In the context of these lecture notes this is refined to:

An **actor** which is a single human being, or essentially a set of human-beings, such as a team.

Information system – A **sub-system** of an **organizational system**, comprising the conception of how the communication and information-oriented aspects of an **organization** are composed and how these operate, thus leading to a description of the (explicit and/or implicit) communication-oriented and information-providing actions and arrangements existing within the **organizational system**.

In the context of these lecture notes this is refined to:

A **sub-system** of an **organizational system**, comprising the conception of how the communication and information-oriented aspects of an **organization** are composed and how these operate, thus leading to a description of the (explicit and/or implicit) communication-oriented and information-providing actions and arrangements existing within the **organizational system**.

Information – The **knowledge** increment brought about when a **human actor** receives a message. In other words, it is the difference between the **conceptions** held by a **human actor** *after* interpreting a received **message** and the **conceptions** held beforehand.

In the context of these lecture notes this is refined to:

The **knowledge** increment brought about when a **human actor** receives a message. In other words, it is the difference between the **conceptions** held by a **human actor** *after* interpreting a received **message** and the **conceptions** held beforehand.

Interest – The specific reason(s) why a **viewer** observes a **domain**.

In the case of a **system**, this this is usually a confluence of the **systemic properties** of **interest** to the **system viewer** and the aspects of the **system** that are considered relevant (by the **system viewer** to these **systemic properties**).

In the context of these lecture notes this is refined to:

The specific reason(s) why a **viewer** observes a **domain**.

In the case of a **system**, this this is usually a confluence of the **systemic properties** of **interest** to the **system viewer** and the aspects of the **system** that are considered relevant (by the **system viewer** to these **systemic properties**).

Knowledge – A relatively stable, and usually mostly consistent, set of **conceptions** possessed by a single (possibly composed) **actor**.

In more popular terms: “an actor’s picture of the world”.

In the context of these lecture notes this is refined to:

A relatively stable, and usually mostly consistent, set of **conceptions** possessed by a single (possibly composed) **actor**.

In more popular terms: “an actor’s picture of the world”.

Link – Any **element** from a **conception** that relates two **concepts**.

In the context of these lecture notes this is refined to:

Any **element** from a **conception** that relates two **concepts**.

Message – **Data** that is transmitted from one **actor** (the sender) to another **actor** (the receiver).

A message may actually be ‘routed’ via several **actors** before reaching its actual receiver. For example, when **human actor** exchange messages, they usually need to make use of some other **actor** playing the role of a medium (for example, vibrations in the air, or an e-mail system).

In the context of these lecture notes this is refined to:

Data that is transmitted from one **actor** (the sender) to another **actor** (the receiver).

A message may actually be ‘routed’ via several **actors** before reaching its actual receiver. For example, when **human actor** exchange messages, they usually need to make use of some other **actor** playing the role of a medium (for example, vibrations in the air, or an e-mail system).

Meta model – A **conception** of a **viewer’s viewpoint** on the world.

In the context of these lecture notes this is refined to:

A **conception** of a **viewer’s viewpoint** on the world.

Method fragment – An identifiable part of methodical **knowledge**.

In the context of these lecture notes this is refined to:

An identifiable part of methodical **knowledge**.

Method – An integrated combination of a: **way of thinking, way of controlling, way of working, way of modeling** (a **technique**), and a **way of supporting** aimed at obtaining results.

In the context of these lecture notes this is refined to:

An integrated combination of a: **way of thinking, way of controlling, way of working, way of modeling** (a **technique**), and a **way of supporting** aimed at obtaining results.

Model concept – A **concept** from a **conception** which is a **model**.

In the context of these lecture notes this is refined to:

A **concept** from a **conception** which is a **model**.

Model relation type – A type of well-defined relations between **models**. An example would be a mapping algorithm between a conceptual **model** and a relational database schema, the relation between the **concepts** of a business process **model** and the concepts of a supporting work-flow, etc.

In the context of these lecture notes this is refined to:

A type of well-defined relations between **models**. An example would be a mapping algorithm between a conceptual **model** and a relational database schema, the relation between the **concepts** of a business process **model** and the concepts of a supporting work-flow, etc.

Modeling fragment – A **method fragment** that is concerned with the actual **modeling** of a **domain**. Three classes of **method fragments** are identified: **viewing framework, way of working, viewpoint, model relation type**. Each **method fragment** must have a unique underlying **way of thinking**.

In the context of these lecture notes this is refined to:

A **method fragment** that is concerned with the actual **modeling** of a **domain**. Three classes of **method fragments** are identified: **viewing framework**, **way of working**, **viewpoint**, **model relation type**. Each **method fragment** must have a unique underlying **way of thinking**.

Modeling – The act of purposely abstracting a **model** from (what is conceived to be) a part of the **universe**.

In the context of these lecture notes this is refined to:

The act of purposely abstracting a **model** from (what is conceived to be) a part of the **universe**.

Model – A purposely abstracted **domain** (possibly in conjunction with its **environment**) of some 'part' or 'aspect' of the **universe** a **viewer** may have an **interest** in.

For practical reasons, a **model** will typically be consistent and unambiguous with regards to some underlying semantic domain, such as logic.

In the context of these lecture notes this is refined to:

A purposely abstracted **domain** (possibly in conjunction with its **environment**) of some 'part' or 'aspect' of the **universe** a **viewer** may have an **interest** in.

For practical reasons, a **model** will typically be consistent and unambiguous with regards to some underlying semantic domain, such as logic.

Open active system – A **system** that is an **open system** as well as an **active system**.

In the context of these lecture notes this is refined to:

A **system** that is an **open system** as well as an **active system**.

Open system – A special kind of **dynamic system** that is conceived as reacting to external triggers, i.e. there may be changes inside the system due to external causes originating from the system's **environment**.

In the context of these lecture notes this is refined to:

A special kind of **dynamic system** that is conceived as reacting to external triggers, i.e. there may be changes inside the system due to external causes originating from the system's **environment**.

Organizational system – A special kind of **system**, being normally active and open, and comprising the conception of how an **organization** is composed and how it operates (i.e. performing specific actions in pursuit of organizational goals, guided by organizational rules and informed by internal and external communication), where its **systemic property** are that it responds to (certain kinds of) changes caused by the system **environment** and, itself, causes (certain kinds of) changes in the system environment.

In the context of these lecture notes this is refined to:

A special kind of **system**, being normally active and open, and comprising the conception of how an **organization** is composed and how it operates (i.e. performing specific actions in pursuit of organizational goals, guided by organizational rules and informed by internal and external communication), where its **systemic property** are that it responds to (certain kinds of) changes caused by the system **environment** and, itself, causes (certain kinds of) changes in the system environment.

Organization – A group of **actors** with a purpose, who:

- interact with each other,
- form a network of roles,
- make use of (the services of) other actors.

An **organization** in itself is an **actor** as well, and may as such participate in yet another **organizations**.

In the context of these lecture notes this is refined to:

A group of **actors** with a purpose, who:

- interact with each other,
- form a network of roles,
- make use of (the services of) other actors.

An **organization** in itself is an **actor** as well, and may as such participate in yet another **organizations**.

Perception – That what results, in the mind of a **viewer**, when they observe a **domain** with their senses, and forms a specific pattern of visual, auditory or other sensations in their minds.

In the context of these lecture notes this is refined to:

That what results, in the mind of a **viewer**, when they observe a **domain** with their senses, and forms a specific pattern of visual, auditory or other sensations in their minds.

Perspective – A set of related [**interests-Is**] in terms of which **viewers** may observe a **domain**.

In the context of these lecture notes this is refined to:

A set of related [**interests-Is**] in terms of which **viewers** may observe a **domain**.

Quality attribute – A specific class of **quality properties**.

In the context of these lecture notes this is refined to:

A specific class of **quality properties**.

Quality property – A **systemic property**, used to describe and asses the **quality** of a **system**.

In the context of these lecture notes this is refined to:

A **systemic property**, used to describe and asses the **quality** of a **system**.

Quality – Is the totality of **systemic properties** of a **system** that relate to its ability to satisfy stated and/or implied needs.

In the context of these lecture notes this is refined to:

Is the totality of **systemic properties** of a **system** that relate to its ability to satisfy stated and/or implied needs.

Sub-system – A sub-system S' of a **system** S , is a **system** where the set of **elements** in S' is a subset of the **elements** in S .

In the context of these lecture notes this is refined to:

A sub-system S' of a **system** S , is a **system** where the set of **elements** in S' is a subset of the **elements** in S .

System activity – A **system concept** that is conceived of as changing parts of the **universe**.

In the context of these lecture notes this is refined to:

A **system concept** that is conceived of as changing parts of the **universe**.

System concept – Any element from a **system** that is a **concept**.

In the context of these lecture notes this is refined to:

Any element from a **system** that is a **concept**.

System description – The **description** of a **system**.

In the context of these lecture notes this is refined to:

The **description** of a **system**.

System domain – A **domain** that is conceived to be a **system**, by some **viewer**, by the distinction from its **environment**, by its coherence, and because of its **systemic property**.

In the context of these lecture notes this is refined to:

A **domain** that is conceived to be a **system**, by some **viewer**, by the distinction from its **environment**, by its coherence, and because of its **systemic property**.

System element – Any element from a **system**.

In the context of these lecture notes this is refined to:
Any element from a **system**.

System engineering – A process involving aimed at producing a changed system, involving the execution of four sub-processes: definition, design, construction and installation of the **system**. Processes that may be executed sequentially, incrementally, interleaved, or in parallel.

In the context of these lecture notes this is refined to:
A process involving aimed at producing a changed system, involving the execution of four sub-processes: definition, design, construction and installation of the **system**. Processes that may be executed sequentially, incrementally, interleaved, or in parallel.

System link – Any element from a **system** that is a **link**.

In the context of these lecture notes this is refined to:
Any element from a **system** that is a **link**.

System viewer – A **viewer** of a **system domain**.

In the context of these lecture notes this is refined to:
A **viewer** of a **system domain**.

Systemic property – A meaningful relationship that exists between the **domain** of elements considered as a whole, the **system domain** and its **environment**.

In the context of these lecture notes this is refined to:
A meaningful relationship that exists between the **domain** of elements considered as a whole, the **system domain** and its **environment**.

System – A special **model** of a **system domain**, whereby all the things contained in that model are transitively coherent, i.e. all of them are directly or indirectly related to each other and form a coherent whole.

A system is conceived as having assigned to it, as a whole, a specific characterization (a non-empty set of **systemic properties**) which, in general, cannot be attributed exclusively to any of its components.

In the context of these lecture notes this is refined to:
A special **model** of a **system domain**, whereby all the things contained in that model are transitively coherent, i.e. all of them are directly or indirectly related to each other and form a coherent whole.

A system is conceived as having assigned to it, as a whole, a specific characterization (a non-empty set of **systemic properties**) which, in general, cannot be attributed exclusively to any of its components.

Technique – Is a synonym for: **way of modeling**.

In the context of these lecture notes this is refined to:
Is a synonym for: **way of modeling**.

Universe – The 'world' under consideration.

In the context of these lecture notes this is refined to:
The 'world' under consideration.

Viewer – An **actor** perceiving and conceiving (part of) a **domain**.

In the context of these lecture notes this is refined to:
An **actor** perceiving and conceiving (part of) a **domain**.

Viewing cell – A **viewing framework** that is not decomposed into a (covering) set of smaller **viewing frameworks**.

In the context of these lecture notes this is refined to:
A **viewing framework** that is not decomposed into a (covering) set of smaller **viewing frameworks**.

Viewing framework – A kind of **modeling fragment**, which identifies a set of related **perspectives** from which to **view** a **system**. A **viewing framework** may be composed of yet other **viewing frameworks**.

In the context of these lecture notes this is refined to:

A kind of **modeling fragment**, which identifies a set of related **perspectives** from which to **view** a **system**. A **viewing framework** may be composed of yet other **viewing frameworks**.

Viewpoint – A specification of the conventions for constructing and using **views**.

This involves: a **way of thinking**, a **way of modeling**, a **way of communicating**, a **way of working**, a **way of supporting** and a **way of using**

In the context of these lecture notes this is refined to:

A specification of the conventions for constructing and using **views**.

This involves: a **way of thinking**, a **way of modeling**, a **way of communicating**, a **way of working**, a **way of supporting** and a **way of using**

Way of conceiving – A set of **modeling concepts** by which **viewers** are to observe **domains**. This usually takes the form of a **meta model**.

In the context of these lecture notes this is refined to:

A set of **modeling concepts** by which **viewers** are to observe **domains**. This usually takes the form of a **meta model**.

Way of controlling – The managerial aspects of system engineering. It includes such aspects as human resource management, quality and progress control, and evaluation of plans, i.e. overall project management and governance (see [Ken84, Sol88]).

In the context of these lecture notes this is refined to:

The managerial aspects of system engineering. It includes such aspects as human resource management, quality and progress control, and evaluation of plans, i.e. overall project management and governance (see [Ken84, Sol88]).

Way of describing – The medium and ‘notations’ used to represent the **concepts** as identified in a **way of conceiving**. It describes how the abstract concepts from the **way of conceiving** are communicated to human beings, for example in terms of a textual or a graphical notation.

Note that it may very well be the case that different **modeling techniques** are based on the same **way of conceiving**, yet use different notations.

In the context of these lecture notes this is refined to:

The medium and ‘notations’ used to represent the **concepts** as identified in a **way of conceiving**. It describes how the abstract concepts from the **way of conceiving** are communicated to human beings, for example in terms of a textual or a graphical notation.

Note that it may very well be the case that different **modeling techniques** are based on the same **way of conceiving**, yet use different notations.

Way of modeling – Identifies the *core concepts* of the language that may be used to denote, analyze, visualize and/or animate **system descriptions**.

In the context of these lecture notes this is refined to:

Identifies the *core concepts* of the language that may be used to denote, analyze, visualize and/or animate **system descriptions**.

Way of supporting – The support to system development that is offered by (possibly automated) tools. In general, a way of supporting is supplied in the form of some computerized tool (see for instance [McC89]).

In the context of these lecture notes this is refined to:

The support to system development that is offered by (possibly automated) tools. In general, a way of supporting is supplied in the form of some computerized tool (see for instance [McC89]).

Way of thinking – Articulates the assumptions on the kinds of problem domains, solutions, engineers, analysts, etc. This notion is also referred to as *die Weltanschauung* [Sol83, WAA85], *underlying perspective* [Mat81] or *philosophy* [Avi95].

In the context of these lecture notes this is refined to:

Articulates the assumptions on the kinds of problem domains, solutions, engineers, analysts, etc. This notion is also referred to as *die Weltanschauung* [Sol83, WAA85], *underlying perspective* [Mat81] or *philosophy* [Avi95].

Way of working – Structures (parts of) the way in which a system is engineered. It defines the possible tasks, including sub-tasks, and ordering of tasks, to be performed as part of the development process. It furthermore provides guidelines and suggestions (heuristics) on how these tasks should be performed.

In the context of these lecture notes this is refined to:

Structures (parts of) the way in which a system is engineered. It defines the possible tasks, including sub-tasks, and ordering of tasks, to be performed as part of the development process. It furthermore provides guidelines and suggestions (heuristics) on how these tasks should be performed.

Work system – An **open active system** in which **actors** perform processes using **information**, technologies, and other resources to produce products and/or services for internal or external actors.

In the context of these lecture notes this is refined to:

An **open active system** in which **actors** perform processes using **information**, technologies, and other resources to produce products and/or services for internal or external actors.

Author Index

A

Alter, S., 7
Antill, L., 13, 30
Avison, D.E., 13, 30

B

Booch, G., 11
Bosma, H., 8, 17
Brinkkemper, S., 8, 17

C

Caston, A., 17, 18

F

Franckson, M., 17, 20, 21

G

Greefhorst, D., 17, 18

H

Halpin, T.A., 9, 11
Heijes, H., 12
Heuvel, W.-J. van den, 17
Hoppenbrouwers, S.J.B.A., 8, 17

I

ISO, 19

J

Jacobson, I., 11
Janssen, R.D.T., 8, 17

K

Kensing, F., 30

Koning, H., 17, 18

L

Lankhorst, M.M., 17–19

M

Mathiassen, L., 13, 30
McClure, C.L., 30

P

Prakash, N., 8, 17
Proper, H.A. (Erik), 8, 17, 20, 21

R

Rolland, C., 8, 17
Rossi, M., 8, 17
Rumbaugh, J., 11

S

Seligmann, P.S., 12
Sol, H.G., 12, 13, 30

T

Tapscott, D., 17, 18

V

Veldhuijzen van Zanten, G.E., 17
Verhoef, D., 8, 17
Verhoef, T.F., 17, 20, 21
Vliet, H. van, 17, 18

W

Wijers, G.M., 12
Wood–Harper, A.T., 13, 30

Z

Zachman, J.A., 11

Subject Index

The following conventions are used in this index:

- A page where a concept is defined: *41*.
- A page where a concept is discussed or mentioned: *41*.
- The page in the dictionary where a concept is defined: **41**.

A

active system, *27, 29*
activity participation, *27, 27*
actor, *7, 27, 27–30*
approach, *13, 27*
architectural description, *27, 27*
architecture, *27, 27*
architecture principle, *7, 27*

C

characterization dimension, *11, 17, 27*
characterization property, *11*
component, *20, 27, 27*
component system, *27, 27*
computerized information system, *7*
concept, *27, 28, 29*
conception, *27, 27, 28*
construction process, *29*

D

data, *27, 28*
definition process, *29*
deployment process, *29*
description, *13, 27, 29, 30*
design principle, *27, 28*

design process, *29*

domain, *13, 27, 28, 28–30*

dynamic system, *28, 29*

E

efficiency, *19*
element, *27, 28, 28, 29*
environment, *7, 20, 27, 28, 29*

F

functionality, *19*

H

human actor, *28, 28*

I

information, *7, 28, 30*
information system, *7, 7, 28*
interest, *28, 28, 29*

K

knowledge, *28, 28*

L

link, *27, 28, 28, 29*

M

maintainability, *19*
message, *28, 28*
meta model, *13, 28, 30*
method, *7, 8, 28*
method fragment, *8, 8–13, 28, 28, 30*
model, *7, 13, 27, 28, 29, 29, 30*

model concept, 27, 28
model relation type, 12, 13, 13, 28, 28
modeling, 7, 8, 10, 13, 27, 28, 29
modeling fragment, 12, 13, 13, 28, 30

O

open active system, 7, 29, 30
open system, 29, 29
organization, 7, 28, 29, 29
organizational system, 7, 7, 28, 29

P

perception, 27, 29
perspective, 13, 29, 30
portability, 20

Q

quality, 29, 29
quality attribute, 19, 29
quality property, 19, 29, 29

R

reliability, 19

S

sub-system, 7, 19, 20, 27, 28, 29
system, 19, 20, 27–29, 30
system activity, 27, 29
system concept, 29, 29
system description, 27, 29
system domain, 27, 29, 29, 30
system element, 27, 29
system engineering, 8, 27, 29
system link, 27, 29
system viewer, 28, 29
systemic property, 7, 28, 29, 29, 30

T

technique, 7, 8, 13, 28, 30

U

universe, 27–29, 30
usability, 20

V

viewer, 13, 27–29, 30, 30
viewing cell, 12, 13, 13, 30, 30
viewing framework, 12, 13, 13, 28, 30, 30
viewpoint, 13, 13, 28, 30

W

way of conceiving, 13, 13, 30, 30
way of controlling, 12, 28, 30
way of describing, 13, 13, 30, 30
way of modeling, 12, 13, 13, 28, 30, 30
way of supporting, 12, 28, 30
way of thinking, 12, 13, 13, 28, 30
way of working, 12, 13, 13, 27, 28, 30, 30
work system, 7, 7, 8, 27, 29, 30
WSML, 7–11, 14, 17, 27

The DAVINCI Lecture Notes Series:

The DAVINCI series of lecture notes is concerned with *The Art & Craft of Information Systems Engineering*. On the one hand, this series of lecture notes takes a fundamental view (*craft*) on the field information systems engineering. At the same time, it does so with an open eye to practical experiences (the *art*) gained from information system engineering in industry.

Main contributors:



P. (Patrick) van Bommel



S.J.B.A. (Stijn) Hoppenbrouwers



G.F.M. (Ger) Paulussen



H.A. (Erik) Proper



Th.P. (Theo) van der Weide