Methodology Framework for Information Systems Development
Vaclav Repa
Department of Information Technologies, Prague University of Economics
Prague, Czech Republic

ABSTRACT
The paper describes so-called “Methodology Framework for Information Systems Development”. It has been developed at the Department of Information Technologies of the University of Economics in Prague, Czech Republic, and its development still continues. Moreover, the development of this facility is intended as the never ending process – the process of international collaboration and experience exchange in the field of the information systems development methodologies.

Keywords: information systems development, ISD methodologies, software development, conceptual modeling, project management.

INTRODUCTION
This paper describes the framework for describing and analyzing information systems development methodologies. It has been developed as a result of multiyear research in the area of information systems development methodologies at the Prague University of Economics.

The reasons for this work were:
1. The need for the unified view on the problem of the development of an information system allowing the comparison of different methodologies, methods, and tools in this field.
2. The need for review of basic principles and rules of an information system development methodology which is “time and space independent” (i.e. independent on different schools, fashions, temporary trends, etc.).
3. The need for the tool for combining different methodologies into the purpose-specific methods and tools.

The first chapter describes the state of the art in the field of information systems development methodologies, problem of their classification as well as their general principles and actual general trends.

In the second chapter the contents of the framework is described in more detail with special respect to its heart – the conceptual model. Particular elements of the model are explained in the context of their relationships to other elements.

The third chapter is aimed on the project of putting the model into the “real life” based on the principles of the communication of open-source communities.

Conclusions contain brief summary, overview of related work, and outline the necessary future work with the Methodology Framework.

STATE OF THE ART
Information systems development methodology is quite complex and ambiguous concept. According to Iivari, Hirschheim, and Klein [4] there are hundreds of information systems development methodologies which differ significantly from each other. Moreover, they speak about continuing growth of the “methodology jungle”. There are also a lot of classifications in this area. We could distinguish between “traditional” structured (see [20] for example) methodologies and “progressive” object-oriented (see [13] for example) ones. We should also take into the account the degree of “formality” in the methodology – this criterion is lately very popular again, so it is discussed in more detail below in this chapter. Anyway, it is necessary to remember that distinguishing between “formal” and “informal” methodologies is just one from many criteria for the comparison of different methodologies and never should be used alone, no matter what is the purpose of the comparison.

Other possible, and today very popular, classification distinguishes between “rigorous” and “agile” methodologies. This classification very closely matches together with the criterion of formality, especially in the context of project management. At last but not at the least, very important attribute of ISD methodology is its dependency on particular aspects – technical environment, purpose of the information system, given business the information system is to be created for, dispensable development environment, and many other aspects. In this sense we are speaking about more or less independent (or dependent) methodologies.

Outlined ways of the information systems development methodologies classification well illustrate the complexity of the problem. There is no wonder that there exist a lot of attempts for universal classification, more or less successful. Iivari, Hirschheim, and Klein propose in their Framework for Information Systems Development Methodologies (ISDM) classification [4] the combination of two basic levels which allow demonstration of all substantial relationships and attributes of analyzed methodologies:

- On the (higher) level of ISD approach there are important following attributes:
  - Goal
  - Guiding Principles and Beliefs
  - Fundamental concepts
  - Principles of the ISD Process
- On the (lower) level of ISD methodology there are important following attributes:
  - Detailed concepts
  - Notations
  - Techniques
  - Tools
  - Heuristics

Each methodology then lies in the intersection of particular combination of particular approach attributes (goal, guiding principles and beliefs, fundamental concepts and principles of the ISD process) and particular combination of particular methodology attributes (detailed concepts, notations, techniques, tools, and heuristics).

It is obvious that the above described approach to the classification, even with respect to its general quality, is not able to capture the whole complexity. ISD methodology heuristics is evidently dependent on guiding principles and beliefs of the approach as well as the tools must follow the fundamental concepts together with principles of the ISD process etc. Using the language of conceptual modeling the problem of ISD...
methodologies classification is the problem of many factors mutually related one to many other, partially.

**General Principles**

In spite of the complexity and diversification of existing methodologies there are some basic principles which are common for all methodologies and independent on any specific paradigm, domain, or other aspects. These principles are mutually connected and subject to each other, they work as a whole and not independently.

There are three main general principles:

- Principle of Modeling
- Principle of Abstraction
- Principle of Three Architectures

**Principle of modeling**

The principle of modeling was first formulated from the data analysis point of view: contents and structure of database objects reflect the contents and structure of real world objects. Correctness of the data model is measured via its similarity to the real world. The need for emphasizing this "similarity" was the main reason for development of the Entity Relationship Diagram - (ERD) [2]. ERD describes the essential characteristics of the real world: objects and their mutual relationships. It is constructed to be able to describe exactly objects and their relationships in the same way as we see them in the real world. At the same time this model describes the essential requirements for the database - it must contain the information about the same objects and their relationships. The form in which a particular database describes these facts always depends on technological and implementation characteristics of the environment in which the database is realized. However, the essence of the model remains the same. Because of the need to describe the same database in its various shapes (essential, technological, implementation) the principle of different architectures has been formulated. This principle, generalized to the scope of the whole system (not only its database) is discussed below. The modeling principle proves to be general too - some parts of system processes have to be regarded as the model of the real world also. However, the main problem of the so-called structured approach to IS development is that it is not able to recognize which system processes form the model of the real world and which do not. Such recognition requires separation of the modeling operations from the other ones and organizing them into the special algorithms according to real world objects and their relationships. This point of view is not reachable under the "structured paradigm" without accepting the natural unity of the modeling system processes and the data in a database. Acceptance of the natural unity of the modeling processes and the data entities, formulated as the main OO principle, enables us to solve Yourdon's problems with control processes [20] - the essential controlling algorithms follow from the entity life histories.

As shown above the Modeling Principle seems to be general and independent of existing paradigms. Each new paradigm can only specify its place in IS development but cannot eliminate or limit it.

**Principle of abstraction**

This principle expresses the actual need for fight against the natural complexity of the problem of information system development. An abstraction is a general arm of this fight. It is a tool allowing the concentration on just some aspects of a problem via abstracting other aspects, which are not relevant or necessary from given context. Special meaning have "hierarchical abstractions" allowing repeating of abstraction levels.

Hierarchical abstractions are the means for decomposing the elements of designed information system to the level of detail. Higher level concepts consist of the lower level ones. On each level of detail the elements of developed IS and their relationships are described. The elements on each higher (i.e. non-elementary) level of detail are abstract concepts. Only the lowest (i.e. most detailed, elementary) level contains definite elements. There is the "tree structure" of dependencies between the concepts of the higher and lower levels, so that each element has only one parent element on the higher level (with the exception of the highest element - root of the tree) and can have several child elements on the lower level (with the exception of the lowest elements - leaves of the tree). Hierarchical abstractions are of two basic types:

- Aggregation. Subordinated elements are parts of the superior concept.
- Generalization. Subordinated elements are particular types of the superior concept.

The aggregation type of abstraction is typically used for decomposing functions into sub-functions as well as processes into sub-processes (using Top-Down procedure) while the generalization type of abstraction is typically used for decomposing entities of the conceptual data model into sub-entities as well as classes into sub-classes. Incompatibility of these two basic approaches to the concept decomposition inside the structured methods has often played the role of a source of vital problems of the "structured paradigm".

**Principle of "Three Architectures"**

The principle of "Three Architectures" was mentioned above in the context of the Principle of Modeling. These two principles have very much to do with one another. Separation of the implementation and technology-dependent aspects of developed information system from the conceptual ones is the vital condition for putting the Modeling Principle into practice. Without such separation, the developer would not be able to see (and to discuss it with the user) the model of the real world as the functional and database structure of developed IS. Three levels of the model of the IS seem to be essential:

- Conceptual model represents the clear model of the real world, which is not "contaminated" by the non-essential aspects given by assumed technology and implementation environment of the system
- Technological model is based on the conceptual model enriched by the aspects given by assumed technology. Technological aspects often significantly change the original (conceptual) shape of the system. For example, 3GL technology using sequential files leads to the data structures considerably distant from the conceptual entities and their relationships. On the other hand, relational database technology preserves a maximum of the original shape of the conceptual data model
- Implementation model depends on the used technology taken into consideration in the technological model respecting also implementation details given by the used particular environment. Thus, the implementation model is even more distant from the real world than the technological one.

Such a model of the three different views on the same thing has some general characteristics:

- each view has specific logic and requires specific methods of examining and specific language for description, which match this logic
- to retain the consistency between particular views it is necessary to have some means (i.e. methods and techniques) for the transition of the contents of one view into the next view

The following figure illustrates essential relationships between the three architectures:
So each of these three levels of IS development represents the specific goal, the specific type of developer's activity, and specific techniques and tools to use. The transition of the design from one to the next level also requires specific activities, techniques and tools. Methods, techniques, activities and tools used on three levels of IS development differ from the functional, data and object points of view. Following tables set out an example of these differences on the conceptual and technology levels:

Table 1. Functional point of view

<table>
<thead>
<tr>
<th>Level</th>
<th>Activity</th>
<th>Tools</th>
<th>Techniques</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONCEPT</td>
<td>Function</td>
<td>Data Flow Diagram, Structure</td>
<td>Event Partitioning</td>
</tr>
<tr>
<td>- UAL</td>
<td>Analysis</td>
<td>Diagram, Structure Diagram,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>State Transition Diagram</td>
<td></td>
</tr>
<tr>
<td>LOGICAL</td>
<td>Program</td>
<td>Structure Chart</td>
<td>Modular Programming,</td>
</tr>
<tr>
<td></td>
<td>Modules</td>
<td></td>
<td>Composite Design,</td>
</tr>
<tr>
<td></td>
<td>Design</td>
<td></td>
<td>Information Hiding etc.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2. Data point of view

<table>
<thead>
<tr>
<th>Level</th>
<th>Activity</th>
<th>Tools</th>
<th>Techniques</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONCEPT</td>
<td>Data</td>
<td>Entity Relationship Diagram</td>
<td>Normalization,</td>
</tr>
<tr>
<td>- UAL</td>
<td>Analysis</td>
<td>(Chen)</td>
<td>Integration (i.e.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Canonical Procedure)</td>
</tr>
<tr>
<td>LOGICAL</td>
<td>Logical</td>
<td>Entity Relationship Diagram</td>
<td>Transformation of data</td>
</tr>
<tr>
<td></td>
<td>Database</td>
<td>(Martin)</td>
<td>model into the logical</td>
</tr>
<tr>
<td></td>
<td>Design</td>
<td></td>
<td>data structures</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3. Object point of view

<table>
<thead>
<tr>
<th>Level</th>
<th>Activity</th>
<th>Tools</th>
<th>Techniques</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONCEPT</td>
<td>Object</td>
<td>Class Diagram and other UML</td>
<td>Normalization,</td>
</tr>
<tr>
<td>- UAL</td>
<td>Analysis</td>
<td>analytic tools</td>
<td>Integration (i.e.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Canonical Procedure)</td>
</tr>
<tr>
<td>LOGICAL</td>
<td>Object</td>
<td>Class Diagram, Component</td>
<td>Transformation of classes</td>
</tr>
<tr>
<td></td>
<td>Design</td>
<td>Diagram, Deployment Diagram</td>
<td>into the logical data</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>structures, Information</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Hiding etc.</td>
</tr>
</tbody>
</table>

Tables also show some important characteristics of the object-oriented approach:

- OO approach combines activities and tools of both - data as well as function-oriented approaches

- OO approach drives for the unified language for both - conceptual as well as logical levels of modeling using the common tool - Class Diagram
- OO approach covers all aspects of the data point of view and some (not the all) aspects of the functional point of view. Especially the Event Partitioning Technique for the conceptual modeling of functions is not acceptable here.

Actual problems and trends

As the field of ISD methodologies is permanently developing not only “static attributes” of the methodology play the role in the process of their recognition and analysis. Some seemingly temporary or situational factors are in fact important general aspects (or are pointing on them, at least) - general problems and trends.

In the present turbulent world, there are fast and permanent changes in the economic environment. In the area of information systems and information and communication technologies (IS/ICT) these changes are still more distinctive. Hardware, software and IS/ICT management processes are changing very fast. New programming languages and integrated development environments, application frameworks, middleware technologies, new forms of IS/ICT operation (outsourcing, ASP) and new IS/ICT management attitudes (COBIT, ITIL) are emerged. IS/ICT developers have more possibilities, but on the other side, developed systems are more complicated and require integration of many technologies and nowadays extremely fast development. IS/ICT developers and operation management face still more complicated problems. Their bad or late solution may have very distinctive impact on competitive company situation.

Many IS development methodologies try to solve the IS development complexity for very long time. In spite of this fact there is still a big part of all IS projects unsuccessful. According to the research report of Standish Group from 1994 only 16% of all application development projects satisfy the criteria of successfulness (project finished in time, according to the budget and with all specified functions). In 2000 there were 28% projects successful. These numbers show the trend of project successfulness escalation, but there are 72% projects, which completely failed or overran the budget, did not fulfill the term or add less functionality. Among 10 key success factors one of the most important ones is the formal methodology. It is also important to choose the right methodology for every project. But existing methodologies are not adequately and uniformly described, in order to categorize them. Methodologies are mostly concentrated on a certain phase of information system development (e.g. object oriented design), on a certain domain, on a certain type of a project etc. There are no criteria for selecting an appropriate methodology and processes, for its customization for concrete company and project conditions.

In the field of analysis paradigms the gap between the analysis and design phase of an IS development project still exists as well as the basic misunderstanding between so-called “structured” and “object oriented” methodologies. The roots of this misunderstanding lie deeply in historical circumstances of the object orientation evolution. Especially the fight against the “traditional” style of thinking, which proceeded mainly in the field of programming, led to the mist of the important difference between the analysis, design, and implementation (see [12] for more information). This causes the existence of the “cultural barricade” between the SA and the OOA and their mutual inability to take advantage of the opposite approach although both approaches are much more complementing one the other than competing. There are some attempts for using the convergence of both approaches as it is for example in [11] based on the work of M. Jackson ([5], [6]).

Present methodologies also insufficiently stress “business” aspects of IS development and also nearly related methodologies
Existing methodologies do not sufficiently support the integration of individual systems and efficient integration of the company management with IS/ICT management. Also the “service orientation” of information management (e.g. management of company IS/ICT on the basis of information services) is not explicitly accepted in existing methodologies.

All the above discussed problems are also the subject of solution by the below presented project MeFIS.

THE FRAMEWORK

The Methodology Framework for Information Systems Development (MeFIS) is the tool for managing the information about general information systems development aspects (i.e. Methodologies). It also serves as a basic template for the definition of the information systems development aspects as well as for the specification of methodical differences for specific type of developed systems (Business Intelligence or ERP for instance).

The central point of the MeFIS is the “Methodology Framework for Information Systems Development – Conceptual Model” - MeFISCoM (see the figure below).

Figure 2 MeFIS – Conceptual Model

MeFISCoM is the conceptual model describing the basic concepts of the information systems development methodology and their basic mutual relations. As the information system – the product of the methodology use - is always the model of some real-world system, this conceptual model of the ISD methodology has strong features of a meta-model (it models modeling) and its convergence with other traditional meta-models (describing languages, methods, and techniques) is obvious.

MeFISCoM describes basic elements of the MeFIS together with their basic dependencies. It is modeled with the Class Diagram from the UML [17], what is the usual tool for meta-modeling (it is also the basis of the Meta Object Facility (MOF) [7]) and de-facto standard for the conceptual modeling. Bright classes in the model specify the contents of the general part of the methodology. Dark classes define the contents of the specific complements of the methodology which are necessary for describing the specificities of the given specific kind of the system. Such specific complements are defined here via the concept Domain. Each Domain specifies particular phases of the general methodology process and their steps which are relevant to the given domain (i.e. kind of the developed system) as well as the specific relevant Critical Success Factors, Principles, and Dimensions of the system. In addition it also defines the specific contents of each Dimension in the given Domain. Each Domain complements the general Methodology Framework with its specific Domain Practices and Task Patterns.

As it follows from the model each methodology in general have to specify the set of general phases. Regardless of the basic sequence of phases defined by the methodology (methodology is...
an ordered set of phases, according to the model), each phase have to define its predecessor and successor. This construction expresses the basic difference between the “main stream” of phases, defined by the logical ordering of main logical groups of activities (i.e. phases), and particular possibilities and/or necessities of mutual ordering of chosen phases given by specific type of methodology or problem, which is also one of important interests of a methodology. These two main viewpoints on the phases ordering are clearly explained in [15]. Following figure 3 illustrates the basic approach of the MMDIS methodology [10], [18] to this problem. It is obvious that in special situations some of phases can be removed from the methodology and/or its specific use case (Global A&D in the case of small or clearly recognized system, for example), some of them can be planwise repeated, there can be some changes in their ordering, etc., while the basic logic of their ordering remains valid.

### Figure 3 Basic Phases Ordering According to the MMDIS

A Phase is a main gate of a Methodology to its other – detailed aspects. As the model shows, each Phase in general consists of ordered Tasks and has connected Key Documents and Products. Key Document is a special kind of Product which serves as a tool for the project management. It clearly defines responsibilities of particular management roles in the particular Phase or in the part of phase. The ordering of Tasks in the Phase, similarly to the ordering of Phases, is given by the fact that a Phase is an ordered set of Tasks as well as by the logical ordering of Tasks expressed in the predecessors – successors manner. Unlike in the case of Phases, each Task can have many predecessors and many successors. On the level of tasks (and all possible deeper levels) there is necessary to see the structure of their relationships as a net, not a sequence. Tasks are many-to-many related to Techniques and Tools, while there can be also parallel and non exclusive association between any Technique and Tool. Particular Tool can be used as the special tool for some technique (ER Diagram in the Normalization Technique, for example) and as the general tool in the particular task (Data Analysis, for example) at the same time.

The set of general concepts Dimension, Principle and Critical Success Factor (CSF) mediates the “bridge” between the Methodology and the Domain. Each of these concepts can be general or specific for the particular Domain. There are modeled general CSF’s, Principles, and Dimension aspects for each Phase as well as specific ones in the context of given Domain (so-called domain-specific) in the same time. In the context of the given Domain there are also modeled specific Practices and Task Patterns. Those aspects of the MeFIS are described in more detail in [2].

### THE PROJECT

The current instance of the Methodology Framework contains the general IS Development methodology based on the MMDIS (Multidimensional Methodology for Information Systems Development) [18] – the official general methodology which serves as the basis for the research in the field of IS Development at our faculty. The methodology covers the general part (i.e. the “bright” objects) of the framework. At the same time it is filled with the standard domain-specific aspects (i.e. the “dark” objects), particularly with the specificities of the ERP systems implementation, the Business Intelligence systems development, and specific application development methodologies like RAD, Agile Methodologies and others. First experience indicates the strong tendency to the specialization of the model (to the domain-group specific sub-classes and associations) as well as to the generalization (in the sense of increasing the modeling area – from IS development towards the business modeling in general sense) at the same time. It is obvious that such model is principally dynamic and that it is impossible to develop it sufficiently in the close-shop mode as well. It is necessary to see it as a process instead of the one-shot action as well as an “open” project instead of the private one. This recognition led us to the MeFIS Project.

As it is stated above the Methodology Framework for Information Systems Development (MeFIS) is the tool for managing the information about general information systems development aspects (i.e. Methodologies). During our multiyear research activities we identified the need for having the tool in both senses – the “ideal” as well as the “technological” sense. In the “ideal” sense we need the model specifying basic elements and relationships in the world of information systems development activities. The conceptual model MeFISCoM is such model. This “ideal” model allows us to share our ideas and research results inside the research team as well as to share them with potential collaborators throughout the world. The model itself is the subject of change (including the evolution) which expresses the recognition in the area. The UML [17] and the methodology of conceptual modeling with the UML, including also principles of meta-modeling and Meta-Object Facility (MOF [7]), is then the language of the research team.

In the “technological” sense we need the tool for collecting and sharing information about methodologies, their basic elements, and their mutual relationships in the structure given by the MeFISCoM. In fact we need the information system supporting by the information concerning methodologies, their substantial characteristics, and relationships. So that the conceptual model MeFISCoM is also the adumbration of the database of such information system in the sense of the meaning of conceptual models in the context of information systems development as it is mentioned above (see Principle of Modeling and Principle of Three Architectures in the first chapter).

The main idea of the project is to put this model into the “real life” on the basis of the open-source projects principles. It requires some “material basis” which could be the subject of the community collaboration (analogy of the source code at OS Projects). As such material basis the instances of the MeFISCoM should serve. We suppose the community will use and develop the current instance and possible other instances, and on the basis of experience of them also the (meta) model itself.
CONCLUSIONS AND FUTURE WORK

The Methodology Framework for Information Systems Development – MeFIS, presented in this paper, is not only the framework for describing and analyzing information systems development methodologies, as it is stated in the introduction. It is also the means for the collaboration of the professional community in this field. The previous chapter describes the project in which we plan to create the platform for communication a collaboration of the community. The collaboration will be provided in several layers – from simple information gathering and discussing over the collaboration on fulfilling the database with data about methodologies and their aspects up to the closest form - the collaboration on the improvement of the contents of the conceptual model MeFISCOm.

In the near future we suppose following development of the MeFIS:

- Completing the fulfillment of the current instance of the framework with the general methodology process (based on MMDIS) and with basic domain aspects:
  - Basic domain practices
  - Basic task patterns
- Eventual minor changes of the framework based on the discussion of the professional community in order to remove its current insufficiencies
- Start of the project based on the reference information model of ISD methodologies. The schema of the information model database will be derived from MeFISCOm.

During the existence and future evolution of the project we suppose at first intensive use and complementing of the system database. As the consequence we suppose continuous growing of the system database as well as the growing of the interested professional community.

The professional community will collaborate on the development of the system database, which will result in the continuous improvement of the conceptual model MeFISCOm. During the improvement of the MeFISCOm we suppose:

- Repairing insufficiencies and backlogs.
- Development of various alternatives of the MeFISCOm reflecting various approaches to the ISD.
- Development of various instances of the MeFISCOm reflecting various aspects of the ISD.

REFERENCES