

Method Engineering: A Formal Description

Ali Sunyaev¹, Matthias Hansen¹ and Helmut Krcmar¹

¹ Technische Universität München, Department of Informatics, Chair for Information Systems, Germany, {sunyaev,hansenm,krcmar}@in.tum.de

Abstract: The development of information systems (IS) requires methods that recommend how to act during the development process. In some cases existing methods cannot cope with the requirements of the project situation at hand. Therefore new methods must be developed. Method Engineering (ME) attends to this application field. In this article, we provide a detailed overview of IS method engineering approaches in order to describe the concept of method engineering. Based on a literature review we derive a formal description of methods that can be used to describe methods in a basic way and transfer them to other fields of application. With the formal description of methods this article facilitates the process of understanding method engineering both for method user and its engineer.

1. Introduction

The development of IS often demands a methodical approach in order to know which steps have to be taken in which order at which time in the development process (e.g. [1], [4]). While in many cases the method and its included steps as a whole are the objects of research, this paper in detail examines the fundamental parts and elements of methods as well as their interplay among each other. The research subject is set in the discipline method engineering. In this context the development of a formal description, which overviews the structure of methods and its corresponding building blocks, shall be emphasised. The formalisation of methods on the one hand helps explaining existing methods and method parts and on the other hand facilitates the transfer of methods to other application fields. Furthermore the formal described methods become more transparent, traceable and understandable for both method users and its engineers.

The paper is organized as follows: Since this research is based on a comprehensive literature review of relevant journals and articles, section 2 overviews the research approach used. Section 3 describes the concept of method engineering and the existing building blocks and elements that can be used to build a method. Section 4 deals with the formalisation of the concept of methods. The application of the formalisation on the basis of an example will be the subject of section 5. The last section concludes the paper and provides an outlook.

2. Research Approach

In order to describe the concept of method engineering we examined description of IS method engineering approaches currently receiving attention in the literature. The literature review was based upon the approach by Webster and Watson [23]. After the identification of relevant journals the examination of appropriate articles was performed. Therefore, a search by keywords was carried out to identify relevant articles. By examining the title and abstract of each article, a total number of 61 articles have been found to be relevant. A further in-depth review resulted in an assortment of 17 articles that were relevant and of importance for the research (table 1).

Table 1. Analyzed articles

Journal	Keywords	Abstract	In-depth review
Accounting, Management & Information Technologies	40	1	0
ACM Computing Surveys	3	1	1
Communications of the ACM	7	5	0
Computers & Security	41	1	1
European Journal of Information Systems	87	4	0
Information Management & Consulting	1	1	0
Information and Software Technology	39	6	1
Information Systems	31	6	2
Information Systems Journal	37	7	0
Information Systems Management	12	1	0
Information Systems Research	5	1	0
Information Technology & People	15	4	0
International Journal of Software Engineering and Knowledge Engineering	10	1	0
Knowledge-Based Systems	10	1	1
MIS Quarterly	14	1	0
Total Journals	352	41	6
Dissertations/ Master's Theses/ Working paper	5	5	3
Conferences/ Workshops	19	19	8
Total	376	65	17

3. Description of the Method Elements

Methods are the cornerstone of goal-oriented and tactical action in many application fields and different kinds of projects. The use of methods is very important

for the development of IS, because they support the developers of IS by providing systematic development approaches [1]. However, the simple existence of development methods is not always a guarantee for successful acting. The environment in which companies are located is changing over time and forces the companies to conform their development work to the increasing complexity of IS. In this context, the need for comprehensive methods that cover a wide range of situations within the development process is emphasised [16].

However, the application of comprehensive IS development methods to specific situations as well as the application of detailed methods to a broad project context is problematic. Rather, methods have to be adapted to the situation at hand. This process is part of the discipline method engineering. Brinkkemper [5] defines method engineering as follows: „Method engineering is the engineering discipline to design, construct and adapt methods, techniques and tools for the development of information systems”.

In the majority of cases a method consists of several parts, which divide the development process in manageable and ordered steps. This division of methods in parts supports the clarity and eases the responsibility assignment. The following section will show and explain the basic concepts and elements regarding methods.

3.1 Method Chains and Alliances

In some cases it can be necessary to connect methods and method parts respectively. This can be done with method chains and method alliances [16].

Nilsson defines method chains as “integration of methods between different levels of development work¹”. This approach to combining methods is a kind of vertical integration. For example, there can be a higher level dealing with conceptual modelling and a lower level dealing with object modelling. Latter can in turn be used for database scheme definition. Method alliances are a method integration within the same abstraction level and thus can be called horizontal integration. Method alliances cover several aspects of problem domains or perspectives at a specific level.

Cronholm and Ågerfalk [8] state that in their opinion Nilsson means method fragments (see next section) when he speaks of methods. We subscribe to this view.

¹ Nilsson makes a difference between strategic, process-oriented and system-oriented development, see [16].

3.2 Method Fragments

Since a fragment is a detached, broken off or an incomplete part of something [18], Harmsen, Brinkkemper and Oei [10] call the building blocks of methods method fragments. These can be classified according to the dimensions perspective and granularity layer² [6].

The dimension perspective considers the product and process view on method fragments. Product fragments are goal-oriented and are deliverables, milestone documents, models, diagrams, etc. Process fragments are process-oriented and are stages, tasks and activities to be carried out [20].

The granularity layer is very important for the description of the type and structure of a method fragment. A fragment can reside on one of the layers *method*, *stage*, *model*, *diagram* or *concept* [6]. The *method* layer addresses the complete method for developing the IS. For instance, the Information Engineering method [15] resides on this granularity layer. The *stage* layer addresses a segment of the life cycle of the IS. An example of a method fragment residing on the stage layer is a Technical Design Report. The *model* layer addresses a perspective of the IS. Such a perspective is an aspect system of an abstraction level. Examples of method fragments are the Data Model and the User Interface Model [17]. The *diagram* layer addresses the representation of a view of a Model layer method fragment. For instance, the Object Diagram and the Class Hierarchy [17] both address the data perspective, but in another representation. For instance, the Statechart [9] resides on this granularity layer, as well as the modelling procedure to produce it. The *concept* layer addresses the concepts and associations of the method fragments on the Diagram layer, as well as the manipulations defined on them. Concepts are subsystems of Diagram layer method fragments. Examples are: Entity and Entity is involved in Relationship [7]. Ter Hofstede and Verhoef [22] add that the sensible arrangement of method fragments on the respective granularity layer is very important in order to avoid negative consequences in the form of monetary or organisational costs.

Concluding the concept of method fragments the relations of method fragments will be explained. Relations clarify the interrelation of method fragments and divide the development of methods in logic steps. Relations between fragments of the same layer and relations between fragments of different layers exist [10].

The relevant relation between fragments of the same layer is the predecessor relation. It says that one step can be done not until the previous step is done. This relation is only defined for process fragments, but can be derived for product fragments. The important relation between fragments of different layers is the input/output relation. It says that on the one hand process fragments require product fragments and on the other hand process fragments produce product fragments.

² The third dimension *abstraction level* is not subject of consideration.

3.3 Method Chunks

Ralyté and Rolland [19] choose an approach that builds up on the concept of method fragments. The existing process and product fragments get combined to a method chunk, which guarantees the close coupling between the process part and its correspondent product part. This is supposed to emphasise the consistency and independence of method chunks [2]. Thus a single method consists of several, loosely coupled and already existing method chunks.

3.4 Method Components

Röstlinger and Goldkuhl [21]³ consider methods as exchangeable and reusable components. Karlsson and Wistrand [14] amend that method components are the smallest meaningful part of methods and consist of a process, notation and concept.

A process describes rules and recommendations for the IS development and informs the method (component) user what actions to perform and in what order. Notation means semantic, syntactic and symbolic rules for documentation. Concepts are categories included in the process and the notation [8]. They support the description of the problem domain and the method itself [13].

4. Formal Description of the Concept of Method

At this point, the concept of method will be described in a formal way for a better understanding. This will facilitate a concise and logical explanation of the construction of methods and its consisting method elements. In addition the formalization will ease the transfer of the concept of method to other areas and the application there.

The method concepts considered in the formalization are method chains and alliances, method fragments, method chunks and method components and will be aggregated under the collective term “method fragments”. Since the involved elements in the mentioned concepts largely vary only by name⁴ and not by content, the aggregation avoids recurrences when formulating the formal description.

Annotation

METHOD: Method itself

S_{MF}: Set of method fragments

TASKS: Set of tasks

ACTIVITIES: Set of activities

³ Cited in [8].

⁴ Process and product fragments in [10] and [19], process and notation in [21].

P: Set of process fragments	STAGES: Set of stages
R: Set of product fragments	DOCUMENTS: Set of documents
REL: Set of relations between method fragments	MODELS: Set of models
MBL: Set of method blocks	DIAGRAMS: set of diagrams

Additionally, we define the function returning the level of a particular method fragment as

$$level: S_{MF} \rightarrow \{n \mid n \in \mathbb{N}\} \quad (1)$$

Since a method fragment is either a process fragment or a product fragment, for S_{MF} applies: $S_{MF} = P \cup R$ (2)

A process fragment $p \in P$ is process oriented and provides guidelines. For p applies:

$$P = \{p \mid p \in TASKS \oplus p \in ACTIVITIES \oplus p \in STAGES\} \quad (3)$$

A product fragment $r \in R$ is goal oriented and provides the results of process fragments. For r applies:

$$R = \{r \mid r \in DOCUMENTS \oplus r \in MODELS \oplus r \in DIAGRAMS\} \quad (4)$$

For the set of relations REL between method fragments the following applies:

$$REL = \{produces, requires, precedes, I_{verb}, I_{hor}\} \quad (5)$$

Relations between method fragments are meaningful and necessary for the method construction. The following relations exist:

Input/Output relations: either for the case that a process fragment requires a product fragment (rule (6)) or for the case that any process fragment produces a product fragment (rule (7)).

$$requires = \{p \in P, r \in R \mid r \text{ is required by } p\}^5 \quad (6)$$

$$produces = \{p \in P, r \in R \mid r \text{ is produced by } p\} \quad (7)$$

Predecessor relations for consistency-checking [10]: if a process fragment requires an existing product fragment, the product fragment has to be produced by a preceding process fragment.

$$precedes = \{p_1, p_2 \in P \mid \exists r \in R: produces(p_1, r) \wedge requires(p_2, r)\} \quad (8)$$

While relations show the relationships between method fragments, the following concepts are used to link method fragments to logical units. These concepts are the integration between method fragments and the development of method blocks.

The integration of method fragments leads to the development of method chains and method alliances. The vertical integration produces method chains by linking method fragments of different levels of granularity. Connecting method

⁵ To be read: for a process fragment p element of P there exists (at least) one product fragment r element of R , for that applies: p requires r .

fragments with same granularity is called horizontal integration. The following two rules show these concepts in a formal way.

$$\text{connects} = \{(mf_1, mf_2 \in MF) \mid ((mf_1 \text{ and } mf_2 \text{ are connected}))\} \quad (9)$$

$$I_{\text{vert}} = \{(mf_1, mf_2 \in \text{connects}) \mid ((\text{level}(mf_1) = \text{level}(mf_2) + 1) \vee (\text{level}(mf_1) = \text{level}(mf_2) - 1))\} \quad (10)$$

$$I_{\text{hor}} = \{(mf_1, mf_2 \in \text{connects}) \mid \text{level}(mf_1) = \text{level}(mf_2)\} \quad (11)$$

The distinction whether it is a vertical (10) or horizontal (11) integration can be made with the help of the granularity level of the involved method fragments. Rule 10 states that the number of the granularity level of the method fragment must be one number higher or lower than the number of the granularity level of the preceding method fragment.

The development of method blocks is one more way to link method fragments to logical units. In this case it is the connection of a process fragment with its respective product fragment. The following applies:

$$MBL = \{(p \in P, r \in R) \mid r \text{ is produced by } p\} = \text{produces} \quad (12)$$

In this context the following rule states the fact that a process fragment and its respective product fragment are always on a higher granularity layer than the method block that is constructed of them:

$$\forall mbl \in MBL, \forall p \in P, \forall r \in R \mid (p, r) = mbl \Rightarrow \text{level}(mbl) = \text{level}(p) - 1 \wedge \text{level}(mbl) = \text{level}(r) - 1 \quad (13)$$

After the explanation of the method elements now the formal description of the global method concept will follow.

$$\text{METHOD} = \{mf_1, mf_2, mf_3, \dots, mf_n, \text{rel}_{1,2}, \text{rel}_{2,3}, \text{rel}_{3,4}, \dots, \text{rel}_{m-1,m}\}$$

$$\text{with } mf_n \in S_{MF}, \text{rel}_{m,n} \in REL$$

The rule states that a method consists of several method fragments, which are connected among each other by relations.

5. Application of the Formal Description

To show the application of the formal description an appropriate method will be drawn on. Risk analysis methods [11] like CRAMM⁶ or ISRAM⁷ do suit for this. The application of the formal description will be performed on the basis of ISRAM because of its broad range of application and its ease of use. ISRAM is a

⁶ CCTA's (Central Computer and Telecommunications Agency of the UK Government) Risk Analysis and Management Methodology, see Baskerville (1993).

⁷ Information security risk analysis method, see Karabacak and Sogukpinar (2005)

quantitative, paper- and survey-based risk analysis method that allows effective participation of managers and staff into the process. Structured in seven steps, ISRAM provides a guideline for risk assessment that considers the probability of occurrence as well as the consequences of occurrence of security breaches. In this paper, only the sub-process for the probability of occurrence of security breaches is taken into account (see fig. 1). In the following due to the lack of space only the first three steps of ISRAM will be described to show the application of the formalisation. The remaining four steps of ISRAM are developed in the same way as shown below.

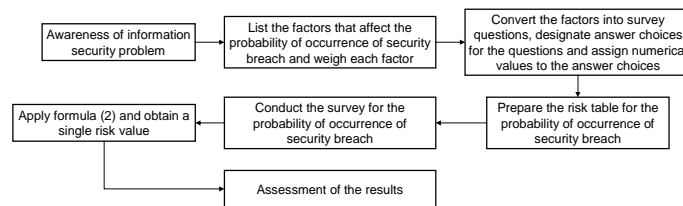


Fig. 1. Basic flow diagram of ISRAM – Sub-process for the probability of occurrence of security breach (according to [12])⁸

(1) Awareness of the problem

Process fragment (PCF) “Realize information security problem(s)”

Within this phase people realize information security problems and decide to perform a risk analysis.

Knowing that a process fragment always produces a product fragment because of the output relation, according to rule (7) $produces = \{(p \in P, r \in R) \mid r \text{ is produced by } p\}$ the following applies:

$produces(PCF \text{ ”Realize information security problem(s)”}, PDF \text{ ”Information security awareness”})$

Product fragment (PDF) “Information security awareness”

Since a product fragment is the result of a preceding process fragment, the “Information security awareness” fragment deals with the situation of the awareness of potential security breaches.

For consistency reasons the process and product fragments get connected to logical units, called method blocks. These blocks can be chosen and reused in any other project where their application makes sense. Applying rule (12) $MBL = \{(p \in P, r \in R) \mid r \text{ is produced by } p\} = produces:$

$MBL \text{ ”Information security awareness”} = (PCF \text{ ”Realize information security problem(s)”}, PDF \text{ ”Information security awareness”})$

⁸ For details to the mentioned formula (2), see Karabacak et al. (2005).

A method step (MS) consists of several method blocks: $MS \in 2^{MB}$. Hence for the method step (MS) “Awareness of information security problem” of the ISRAM we can state:

MBL “Information security awareness \in MS “Awareness of information security problem”

(2) Listing and weighing the factors

PCF “Categorise factors”

Within this fragment separate analyses are made for two risk parameters to determine the factors, which affect these parameters. After determining and listing all the factors, different weight values are assigned to the factors [12].

produces(PCF “Categorise factors”, PDF “Weighted security breach factors”)

PDF “Weighted security breach factors”

As a result the corresponding product fragment shows all the factors that should be included in the survey developed in the next step.

Both the process and product fragment together form the MBL “Security breach factors” which in turn provides the MS “List the factors that affect the probability of occurrence of security breach and weigh each factor”.

MBL “Security breach factors” = (PCF “Categorise factors”, PDF “Weighted security breach factors”)

MBL “Security breach factors” \in MS “List the factors that affect the probability of occurrence of security breach and weigh each factor”

(3) Converting factors into questions, designating answer choices and assigning numerical values to answer choices

Before the PCF “Develop survey” can be executed, existing relations must be considered. In this case there is an input relation between the PDF “Weighted security breach factors” from the last step and the PCF “Develop survey” in this step. According to rule (6) the input relation can be expressed like $requires = \{(p \in P, r \in R) \mid r \text{ is required by } p\}$. Since the process fragment and its corresponding product fragment is encapsulated in the MBL “Security breach factors”, we need first to decompose the method block into its corresponding method fragments. The following applies:

MBL “Security breach factors” = (PCF “Categorise factors”, PDF “Weighted security breach factors”)

requires(PCF “Develop survey”, PDF “Weighted security breach factors”)

To ensure the consistency of the method fragments the predecessor relation between the involved process fragments has to be checked. Applying rule (8) $precedes = \{(p_1, p_2 \in P) \mid \exists r \in R: produces(p_1, r) \wedge requires(p_2, r)\}$:

requires(PCF "Develop survey", PDF "Weighted security breach factors") produces(PCF „Categorise factors“, PDF „Weighted security breach factors“) → precedes(PCF „Categorise factors“, PCF „Develop survey“)

PCF "Develop survey"

The factors analysed in step 2 are converted into appropriate survey questions and answer choices are designated. Subsequent answer choices are converted into numerical values according to a given schema [12].

produces(PCF "Develop survey", PDF "Survey with questions and corresponding numerical values")

PDF "Survey with questions and corresponding numerical values"

The finished process fragment provides the complete survey with questions and numerical values. The corresponding MBL is called "Survey development" and facilitates the development of the MS "Convert the factors into survey questions, designate answer choices for the questions and assign numerical values to the answer choices".

MBL "Survey development" = (PCF "Develop survey", PDF "Survey with questions and corresponding numerical values")

MBL "Survey development" ∈ MS "Convert the factors into survey questions, designate answer choices for the questions and assign numerical values to the answer choices"

The graphical overview of the three described method steps is shown in Figure 2.

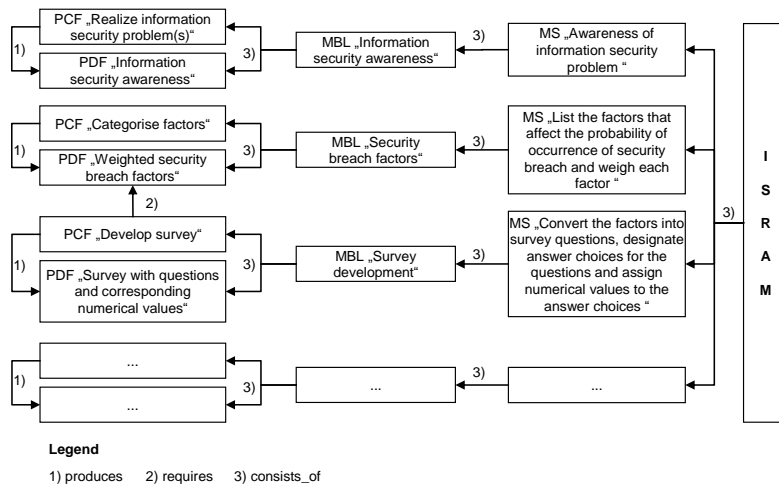


Fig. 2. Structure of the first three steps of ISRAM

To finish the description of the formal application the rule for the global method

concept $METHOD := \{mf_1, mf_2, mf_3, \dots, mf_n, rel_{1,2}, rel_{2,3}, rel_{3,4}, \dots, rel_{m-1,m}\}$ is applied. For the set of process fragments P and the set of product fragments R the following applies:

$P = \{P_1, P_2, P_3\}$ with $P_1 = \{\text{"Realize information security problem(s)"}\}$, $P_2 = \{\text{"Categorise factors"}\}$, $P_3 = \{\text{"Develop survey"}\}$

$R = \{R_1, R_2, R_3\}$ with $R_1 = \{\text{"Information security awareness"}\}$, $R_2 = \{\text{"Weighted security breach factors"}\}$, $R_3 = \{\text{"Survey with questions and corresponding numerical values"}\}$

With the sets of process and product fragments given, the following does apply for the global method concept METHOD of the first three steps of ISRAM:

$METHOD_{ISRAM} = \{P_1, R_1, P_2, R_2, P_3, R_3, produces = \{(P1, R1), (P2, R2), (P3, R3)\}, requires = \{(P3, R2)\}\}$.

6. Summary and Outlook

This article introduced a formal description of method engineering. Using a structured research approach we first examined the existing literature on IS method engineering approaches. Based on the literature review we discussed the construction and description of the concept of method engineering. We then presented a detailed approach for formalising the concept of method engineering, unveiling the method fragments and its relations that underlie formalisation design. As a next step we provided an illustration of an example for formal description of methods in order to show its possible application in research and practice.

The introduced formalisation of the concept of method engineering can be used to describe each part or element of almost any IS method. Therefore, it can be applied to get information about the structure and the underlying assembling technique. This knowledge can help to develop new methods and to reconstruct existing methods in order to adapt them to the project situation at hand. Furthermore due to the formal description of methods and in comparison to descriptive texts as used today, the method engineering becomes more traceable. This modular approach allows a transparent and understandable design of the processes and method development itself. In particular, formal description of methods enables the automated (computer-assisted) method engineering.

References

1. Avison, D.E., Fitzgerald, G. (1995) Information Systems Development: Methodologies, Techniques and Tools. McGraw-Hill, New York.
2. Ayed, M.B., Ralyté, J. and Rolland, C. (2004) Constructing the Lyee method with a method engineering approach. In: Knowledge-Based Systems, Vol. 17 (2004), pp. 239-248.
3. Baskerville, R. (1993) Information Systems Security Design Methods: Implications for Information Systems Development. In: ACM Computing Surveys, Vol. 25 (1993) No. 4, pp. 375-414.

4. Braun, C., Wortmann, F., Hafner, M. and Winter, R. (2005) Method Construction – A Core Approach to Organizational Engineering. In: ACM Symposium on Applied Computing SAC'05, March 13-17, 2005, Santa Fe, New Mexico, USA.
5. Brinkkemper, S. (1996) Method engineering: engineering of information systems development methods and tools. In: Information and Software Technology, Vol. 38 (1996) No. 4, pp. 275-280.
6. Brinkkemper, S., Saeki, M. and Harmsen, F. (1999) Meta-Modelling Based Assembly Techniques For Situational Method Engineering. In: Information Systems, Vol. 24 (1999) No. 3, pp. 209-228.
7. Chen, P. Pin-Shan (1976) The Entity-Relationship Model – Toward a Unified View of Data. In: ACM Transactions on Database Systems, Vol. 1 (1976) No. 1, pp. 9-36.
8. Cronholm, S. and Ågerfalk, P.J. (1999) On the Concept of Method in Information Systems Development. In: <http://citeseer.ist.psu.edu/283885.html>.
9. Harel, D. (1987) Statecharts: A Visual Formalism for Complex Systems. In: Science of Computer Programming, Vol. 8 (1987), pp. 231-274.
10. Harmsen, F., Brinkkemper, S. and Oei, H. (1994) Situational Method Engineering for Information System Project Approaches. In: A.A. Verrijn-Stuart, T.W. Olle (Eds.), *Methods and Associated Tools for the Information Systems Life Cycle*. Elsevier Science B.V., Amsterdam 1994, pp. 169-194.
11. Huber, M., Sunyaev, A., Krcmar, H. (2008) Security Analysis of the Health Care Telematics Infrastructure in Germany. In: Proceedings ICEIS 2008, 10th International Conference on Enterprise Information Systems, Barcelona, Spain, 12.-16.06.2008. To appear.
12. Karabacak, B. and Sogukpinar, I. (2005) ISRAM: information security risk analysis method. In: Computers & Security Vol. 24 (2005), pp. 147-159.
13. Karlsson, F., Ågerfalk, P.J. and Hjalmarsson, A. (2001) Method Configuration with Development Tracks and Generic Project Types. In: 6th CaiSE/IFIP8.1 International Workshop on Evaluation of Modeling Methods in System Analysis and Design (EMMSAD'01), Interlaken, Switzerland, 4-5 June 2001.
14. Karlsson, F. and Wistrand, K. (2004) MC Sandbox – Tool Support for Method Configuration. In: http://www.nuigalway.ie/acc/documents/fredrik_karlsson_NUIG-seminar-21052004.pdf.
15. Martin, J. (1990) Information Engineering, Book II – Planning and Analysis. Prentice Hall, Englewood Cliffs 1990.
16. Nilsson, A.G. (1999) The Business Developer's Toolbox: Chains and Alliances between Established Methods. In: A.G. Nilsson, C. Tolis and C. Nellborn (Eds.), *Perspectives on Business Modelling, Understanding and Changing Organisations*. Springer-Verlag, Berlin, Heidelberg 1999, pp. 217-241.
17. Object Management Group OMG (2008) The Object Management Group. In: <http://www.omg.org/>.
18. Oxford Online Dictionary (2008) Fragment. In: http://www.askoxford.com/concise_oed/fragment?view=uk.
19. Ralyté, J. and Rolland, C. (2001) An Approach for Method Reengineering. In: H.S. Kunii, S. Jajodia, A. Solvberg (Eds.), *Conceptual Modeling. Proceedings of the 20th International Conference on Conceptual Modeling, Yokohama, Japan, November 27-30, 2001*. Springer-Verlag, Berlin, Heidelberg 2001, pp. 471-484.
20. Rolland, C. (1997) A Primer for Method Engineering. In: <http://citeseer.ist.psu.edu/rolland97primer.html>.
21. Röstlinger A & Goldkuhl G. (1994). In Swedish: Generisk flexibilitet – På väg mot en komponentbaserad metodsyn. Presenterat på VITS Höstseminarium 1994. Institutionen för datavetenskap, Linköpings universitet.
22. Ter Hofstede, A.H.M., Verhoef, T.F. (1997) On The Feasibility Of Situational Method Engineering. In: Information Systems Vol. 22 (1997) No. 6/7, pp. 401-422.
23. Webster, J. and Watson, R.T. (2002) Analyzing the past to prepare for the future: writing a Literature Review. In: MIS Quarterly, Vol. 26 (2002) No. 2, pp. xiii-xxiii.