Success-driven Selection of Fragments for Situational Methods
The S^3 model

Frank Harmsen[1], Ivo Lubbers[1], Gerard Wijers[2]

[1] Design Methodology Research Group
Department of Computer Science, University of Twente
P.O. Box 217, 7500 AE Enschede, the Netherlands
phone: +31.53.893690, e-mail: {harmsen, lubbers}@cs.utwente.nl

[2] ID Research
Kastanjelaan 4
3833 AN Leusden, the Netherlands
phone: +31.33.951416, e-mail: GWijers@idr.iaf.nl

Abstract

Engineering situational methods requires first of all matching the situation with the available method fragments. In this paper, we describe an approach in which not only the situation, but also the success one wants to achieve with a project is taken into account. In the Situation-Success-Scenario (S^3) model, the project situation is compared with the project success, which is then related to the set of aspects that are required in the situational method, the project scenario. We give lists of aggregated situation factors, performance indicators, and scenario aspects, mainly taken from the literature on these subjects. We describe a procedure that assists in selecting appropriate method fragments by offering heuristics.

1 Introduction

Current IS development methods are situation independent, and claim to be applicable in virtually any application domain. However, due to the ever increasing complexity and diversity of information systems, IS projects do not require general purpose IS development and management methods, but specialised, dedicated approaches [Kumar 92; Swede 93; Vessey 94]. These so-called situational methods should take at least the *situation* in which they are applied, for instance the IS type, the DBMS platform, the experience of the project members, etc., into account, thus obtaining a better fit between approach to be followed and the required tasks and deliverables.

Various field studies (e.g. [Slooten 94]) showed, that this situation-dependency is in particular important during the first stages of IS development. In order to efficiently perform requirements analysis and conceptual design, and to produce functional specifications of high quality, general-purpose methods are less suitable. Many IS

development projects failed, because requirements analysis was not performed in conjunction with an assessment of the project situation or the project goals [Charette 89]

Situational Method Engineering [Harmsen 94a] is the research area having the philosophy, that IS development projects should strive for *controlled flexibility*, being the balance between rigid general-purpose methods and ad-hoc, flexible development. To engineer a project-specific or *situational* method, first a characterisation of the situation in which the method will be applied, often a project, is made. This characterisation is input to the selection process, where parts of existing IS development methods are retrieved. Such parts, called *method fragments*, stored in a *method base*, address both the method process perspective, such as modelling steps and stages, and the method product perspective, such as descriptions of reports, models, and diagrams. The unrelated method fragments are then assembled into a situational method, using a large number of assembly guidelines to ensure internal consistency and completeness. Finally, the situational method is forwarded to the systems developers in the project. As the project may not be definitely clear at the start, a further elaboration of the situational method can be performed during the course of the project. Similarly, drastic changes in the project require to change the situational method by the replacement of inappropriate fragments. Due to the complexity of this process, it has to be supported by a CAME (Computer Aided Method Engineering) tool, a prototype of which is currently under development in our research group.

This paper focuses on the selection of appropriate method fragments, given a project situation and given the success one wants to achieve with a project. In section two, we outline the research approach we followed to obtain a procedure that supports the method engineer in building a project scenario. The cornerstone of this procedure is the S^3 model, which relates the situation, the required success of the project, and the aspects belonging in the situational method. The S^3 model is covered in section three, while the actual selection procedure is described in section four. The paper ends with conclusions and suggestions for further research.

2 Approach

Practical experience and literature (for instance, [Boehm 88; Bennatan 92; Euromethod 94ab]) brought us to the assumption, that the required aspects in a situational method, indicated by the term *project scenario*, depend on both the project situation and the required project success. This assumption has led to the Success-Situation-Scenario (S^3) model., which was developed after extensive review of existing literature concerning situation theory, project management, IT performance measuring, and IS development methods. Moreover, two field studies in a large Dutch bank and a large Dutch software house have been performed under the supervision of our research group.

In literature [Gilb 88; Olle 91; Slooten 93; Euromethod 94a; Offenbeek 94; Slooten 94; Swede 94] and in our field studies, we have found about 250 situation factors, many of which were overlapping. A complete list of these factors, including their respective sources, can be found in [Hoef 95a]. All factors have to be valuated, dependencies between factors have to be taken into account, and consensus has to be found among those who valuate the factors. The number of possible combinations would have complicated the S^3 model in such a way, that practical application would not be possible. By removing *unclear*, *unimportant*, and *unreal* situation factors, a substantial reduction

was achieved. Unclear situation factors are factors that are not clearly defined, unimportant factors are factors that are only mentioned one time while not being acknowledged very important by the author, and unreal situation factors are factors that are, in our perception, performance indicators or scenario aspects.

Subsequently, we have grouped the remaining factors by looking at *overlapping* factors, *sub-factors*, and *factors having the same influence*. Finally, the situation factors were partitioned into three groups.

To identify performance indicators, a similar approach was followed. Part of the performance indicators, the product-related indicators, are software specification quality factors, as found in, for instance [Gilb 88], [ISO 91], and [SERC 92]. Other, process-related, performance indicators can be found in literature on software process improvement (for instance [Humphrey 89; Dion 92]) and software costs [Boehm 88]. Performance indicators related to the information produced by the information system were also found in the software product quality literature, as well as in [Swede 94]. After the two reduction steps, we obtained 23 performance indicators, divided in a number of sub-factors.

The scenario aspects were mainly found by investigating literature on IS methodology [Olle 91] and by delving into manuals of IS development methods, notably Information Engineering, Method/1 9.0, Navigator 2.0, KADS, Merise, SDM, and SSADM.

To efficiently determine the relationships between situation factors and performance indicators, and between scenario aspects and performance indicators, for each type of relationship a matrix was drafted. Subsequently, these matrices were filled during brainstorm sessions. About 550 relationships were identified, which is, again, too many for practical use by a project manager or method engineer. Therefore, we have clustered groups of related performance indicators and situation factors. The 20 clusters we found were then mapped onto the scenario aspects, using the relationships between scenario aspects and performance indicators.

Finally, we determined the selection procedure. The relationships between performance indicators and scenario aspects were ordered by degree of differentiation, which resulted in three groups of scenario aspects: the prerequisites (which turned out to be the project goal), the non-differentiating aspects and the differentiating aspects. This partitioning has determined the focus of each of the first three steps of the selection procedure. The relationships between clusters and scenario aspects resulted in a number of practically applicable heuristics, for each cluster one.

3 The S^3 model

Situation and method have to be tuned to each other by selecting appropriate method fragments. In traditional approaches to method fragment selection (e.g. [Slooten 93]), so-called situation factors or contingencies are matched with properties of method fragments in the method base. In risk analysis and management approaches (see, for instance, [Charette 89]), a number of activities is incorporated into the project plan to reduce anticipated risk. A combination of these two approaches is, to take the *success* the project is aiming at, as well as the situation of the project, into consideration. Our Success-Situation-Scenario model is an elaboration of [Swede 94]. We take the variables situation

and success as the starting point of method fragment selection. The relationships between situation factors and *performance indicators* describing the expected success determine the aspects required in the situational method, the project scenario.

Figure 3.1 visualises the dependencies between the 3 S's: Success, Situation, and Scenario.

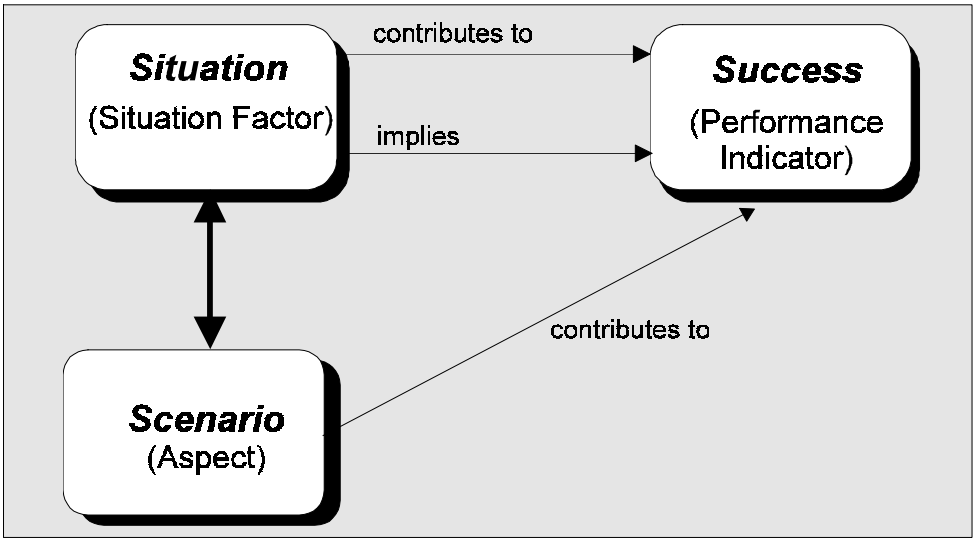

Figure 3.1 The S^3 model

We clarify the S^3 model with the following example. Suppose the situation factor "management commitment" is low. It turned out that this *implies* that success is *not* achieved with respect to the performance indicators "organisation management", "organisational fit", and "system acceptance". This would require additional effort to obatin success, meaning, in this case, the incorporation into the project scenario of the aspects "organisation approach", "custom development", "high degree of user participation", "responsibility for installation with user organisation", "information management", and "phasing". All these aspects *contribute* to the success of the performance indicators mentioned. The relationship between situation and scenario is therefore indirect.

In this section we describe the factors that describe project success, project situation, and project scenario respectively. We have restricted ourselves to scenarios for projects aiming at information systems development; we do not consider, for instance, information planning or business process redesign.

3.1 Performance indicators

For each of the following performance indicators, we give a definition of success. If one or more of the components of such a definition is not achieved, then *no success* with respect to the performance indicator concerned is achieved. We have identified three groups of

performance indicators: those related with the IS development process, those related with the products of IS development, and those related with final result of the project.

Process-related performance indicators:

- *Time*. Success is achieved if the project is performed within the original schedule.
- *Money*. Success is achieved if the original budget is not exceeded.
- *Organisation management*. Success is achieved if tasks are well-defined, mutual relationships and responsibilities between project members are clear, and if communication and decision-taking are structured.
- *Information management*. Success is achieved if documentation is accurate, well-recorded, and complete.
- *Quality control*. Success is achieved if quality requirements for the delivered products are defined, and if a quality system with quality audits and inspections is set up.
- *Customer relationship*. Success is achieved if much attention is paid to contracting, and customer advise, in order to obtain customer-satisfiability.
- *Uncertainty management*. Success is achieved if the project organisation succeeds in identifying and successfully managing risks.
- *Complexity control*. Success is achieved if the information system structure as well as the relationships of the information system with its environment are not complicated.
- *Installation in accordance with specifications*. Success is achieved if all requirements, contained in contract, functional specification etc., are met.

Product-related performance indicators:

- *Functionality*. Success is achieved if the requirements completeness, correctness, connectability, protectability, traceability, and reusability are met.
- *Reliability*, defined by the requirements: availability, durability, degradation possibility, recoverability, and flexibility. Success is achieved if all these requirements are met.
- *Usability*, defined by the requirements: operability, support, clearness, user-friendliness, transparency, trainability, surveyability, transaction speed, and equipment level. Success is achieved if all these requirements are met.
- *Efficiency*, defined by the requirements: resource allocation, response time, and batch transaction speed. Success is achieved if all these requirements are met.
- *Maintainability*, defined by the requirements: parameterizability, controllability, adaptability, testability, and modifiability. Success is achieved if all these requirements are met.
- *Portability*, including the possibility to install the system. Success is achieved if these two requirements are met.

Result-related performance indicators:

- *Satisfaction of information need*. Success is achieved if the following performance indicators associated with the information produced by the system have a high value: correctness of information, completeness of information, up-to-dateness of information, accuracy of information, and verifiability of information
- *Business process improvement*. Success is achieved if the business process is improved with respect to effectiveness, efficiency, flexibility, stability, and continuity.
- *System acceptance*. Success is achieved if the user organisation accepts all aspects of the information system.
- *Organisational fit*. Success is achieved if the information system can be installed without eliciting major changes in organisational structure or culture.

3.2 Situation factors

Situation factors are partitioned into three groups: environment, project organisation, and project.

Environment:

- *Management commitment*. This factor depends on the height of the hierarchical organisation level at which the project goal is supported, the availability of influential project sponsors, and the way management is involved in the project.
- *Strategic importance*. This factor also includes the project priority, and the nature of the system's relationships with the market place.
- *Organisational impact*. This factor also includes the consequences of the project goal for the organisation, and the system's impact on the persons involved.
- *Degree of resistance*. Sub-factors of this factor are the interests of the persons involved, the attitude of the user organisation towards the system, and the flexibility and adaptability of the user organisation.
- *Conflict potential*. This factor depends on the existing conflicts within the user organisation, as well as the internal discord concerning system functionality.
- *Scarcity of people and resources*, including available people and resources within the user organisation, the amount of time, and the budget available for the project.
- *Stability* of information and business processes, referring to the number of changes the information or business process has gone through or is expected to go through.
- *Formality* of information and business processes, indicating the extent to which information is subject to strict rules, and the availability and formality of business process descriptions.
- *User knowledge and experience*. Sub-factors of this factor are the extent to which the system is new for the user organisation, the professionality of the end-user, and the capability maturity level of the user organisation.

Project organisation:

- *Capability* of project team, indicating the relevant knowledge and experience of the team members with the problem area and with equipment and tools to be used.
- *Size* of project team, which is determined by the number of team members.
- *Dependency* of other projects, including the sub-factors: number of interfaces with other systems, and the number of sub-contractors.

Project:

- *Clarity of project goal*, including the well-definedness of the project goal, and the quality of the requirements,
- *Stability of project goal*, *Quality of specifications*, *System size*,
- *System complexity*, including the complexity of the problem area, and the complexity of the information and the business processes.
- *Degree of innovation*, which depends on the degree of innovation of the system, the degree of innovation of system development, and the use of state-of-the-art equipment and tools.

3.3 Scenario aspects

A scenario can contain the following aspects:

- *Project goal*, which can be: *knowledge acquisition*, applicable if knowledge of -part of- an organisation should be increased before actually adapting or developing an information system, *adaptation*, implying that an existing automated or partly automated information system is re-engineered, or that functionality of the information system is modified or extended, *custom development*, implying developing a new automated information system, or *packaged system selection and adaptation*, where a software package is selected and adapted according to the needs of the situation.
- *Approach*, which we distinguish into: *organisation approach*, in which the information supply associated with the information system is considered one of the organisation's essential functions, and *product approach*, in which the information system is considered an isolated product with less importance for the organisation.
- *Degree of user participation*, distinguishing between: *no participation*, *consultation*, in which users are consulted by developers before design decisions are made, *representation and consensus*, where users actively take part in the development process, responsibility for chosen solutions is with one user, or with the entire user organisation, and *end-user computing*, where users develop large parts of the information system themselves. To reduce complexity, only a distinction between little user participation (including consultation), and much user participation was made.
- *Installation strategy*, which can be: one shot installation, *installation in parts*, where a preliminary system is installed and subsequently extended, and *evolutionary installation*, meaning that the system will be periodically updated by new versions.
- *Time management* is the set of activities focused on keeping the project on schedule, such as time estimating, time planning, and time monitoring.

- *Budget control* focuses on financial aspects of project management, such as cost estimating, budgeting, and defining cost reduction procedures.
- *Information management* is the set of activities managing the -intermediate- results of the project, such as defining the contents of documents, allocating responsibilities for information, and recording information. A distinction can be made between documentation management, and software configuration management
- *Quality control*, which contains activities to define quality requirements, quality procedures, and a quality system. We distinguish between quality control by performing inspections, and quality control by performing audits.
- *Project organisation management*, containing all activities aiming at better co-ordination and communication within the project, such as task allocation, keeping personnel records, writing project team meeting minutes, and communication training.
- *Procurement and contract management*, which are all activities associated with tendering, the project contract, and the customer-supplier relationship.
- *Methodological support and training*, containing activities such as defining method usage guidelines, assisting method users, and identifying training requirements.
- *Phasing* is the partitioning of the project in stages, such as business analysis, system design, system construction, and system evolution.
- *Responsibility* for development, partitioned into: *responsibility with development organisation* for the entire development process, user-responsibility for *designing specifications*, user-responsibility for *testing*, and user-responsibility for *installation*.
- *Degree of re-use* of existing design or software components, distinguishing: *some or much re-use*, and *no re-use at all*.
- *Analysis starting point*, which can be the *current* situation or the *required* situation.
- *Validation strategy*, which is the way the delivered products, such as the specifications and the system itself, are evaluated and reviewed by the user organisation. We distinguish between: *immediate acceptance*, without any validation, and *validation*, by applying test cases or prototypes, or by simply the using the system for a while.
- *Tools* to be used, which are partitioned into: *analysis and design tools*, *generators* (code generators and database structure generators), *programming environments*, *Database Management Systems*, *management tools* (process management tools and project management tools), *documentation tools* such as word processors, *configuration/version management tools*, and *automated methods*, which are electronically available method manuals containing hyperlinks to various components.
- *Modelling aspects*. In information systems modelling, several aspects to be modelled play a role. We distinguish: *goals & requirements* of the organisation and the information supply, *data*, *processes*, *behaviour*, *interaction with the environment* of the information system, and *allocation* of processes and data to people and resources.
- *Modelling perspectives*, distinguishing between *organisation* and *information system.*

4 Selection procedure

The selection of project-appropriate method fragments contains 4 steps: determination of the project goal, determination of a preliminary scenario, adaptation of the preliminary scenario, resulting in a project scenario, and selection of the method fragments containing the aspects of the project scenario.

The four steps to be taken are explained in the next sub-sections. Note, that the procedure cannot yield *all* suitable method fragments, as each situation may require additional method fragments, the selection of which is possibly not covered. The procedure therefore only assists in finding necessary method fragments, and does not guarantee completeness. The issue of completeness is partly related to method assembly, which is beyond the scope of this paper (see, for instance [Harmsen 94b] and [Hoef 95b]).

4.1 Determination of the project goal

The overall *goal* of a project is a prerequisite influencing both situation factors and performance indicators [Euromethod 94ab]. For the sake of simplicity, we have identified three main project goals. For each of the goals, we indicate its influence on project success.

- Knowledge acquisition, which does not primarily lead to a working information system, but to better uncertainty management and complexity control.
- Adaptation, which should lead to an information system of a higher quality than the existing -already partly automated- system. This contributes to the performance indicators functionality, reliability, usability, efficiency, maintainability, portability, and business process improvement, as well as to correctness, completeness, up-to-dateness, and verifiability of the produced information.
- Development, which should lead to a new -partly automated- information system. This goal is partitioned into:
    - Packaged systems selection and adaptation, which contributes to the performance indicators time and money, reliability, portability, and to completeness and correctness of information. It is less suitable to obtain good organisational fit, high system acceptance, and high maintainability.
    - Custom development, which contributes to the performance indicators installation in accordance with specifications, functionality, efficiency, usability, and organisational fit. It is less suitable to obtain success regarding time and money.

4.2 Determination of the preliminary scenario

The preliminary scenario assumes an ideal situation without situation factor values negatively affecting project success. The preliminary scenario contains non-differentiating scenario aspects, i.e. aspects that do not contribute to any performance indicator in particular, extended with the “standard” situation of the other aspects. The following aspects are part of the preliminary scenario: Product approach (only for custom development), user participation, one shot installation strategy (not for knowledge

acquisition), phasing, responsibility with development organisation (not for knowledge acquisition), methodological support and project member training, linear development, analysis of required situation, analytical way of design (not for knowledge acquisition, in that case: experimental way of design), no re-use at all of existing design or application components, no validation of the system.

The following management aspects will not be considered: time management, budget control, information management, quality control, procurement and contract management.

4.3 Adaptation of the preliminary scenario

By valuating the situation factors and applying heuristics, the preliminary scenario will be transformed into a project scenario, which prescribes the aspects that should be incorporated into the situational method. To give an impression of these heuristics, we have incorporated some of them into this paper:

- If project success is important with respect to the performance indicator *project organisation*, and if the following situation factor values are applicable: *low management commitment*, *large development team*, *unclear project goal*, *high conflict potential*, then the following aspects are suitable for incorporation into the scenario: *quality audits and inspections*, *contract management*, *procurement*.
- If project success is important with respect to the performance indicator *good customer relationship*, and if the following situation factor values are applicable: *unclear project goal*, *high conflict potential*, *unstable project goal*, *high organisational impact*, then the following aspects are suitable for incorporation into the scenario: *quality audits*, *contract management*, *procurement*.
- If project success is important with respect to the performance indicator *system reliability* and if the situation factor value *high organisational impact* is applicable, then the following aspects are suitable for incorporation into the scenario: *quality control by inspections, re-use*, *validation*, *attention to information system modelling perspective*, *attention to the performance and allocation modelling aspects*, *application of generators*.

4.4 Selection of method fragments

The project scenario determines to a large extent the choice of the actual method fragments from the method base, that is, the techniques, tools, descriptions of deliverables and models, activities, etc., that together form a situational method. Method fragments are characterised by a number of properties, including name, origin, experience required, and purpose, which is directly related to the scenario aspects. Therefore, selection of method fragments boils down to matching required aspects with available purposes. As was mentioned before, often additional method fragments have to be selected, because the S^3 model cannot cope with all situations. Different choices among method fragments having the same purpose are possible, which can be resolved by considering the mutual conceptual fit between method fragments. Method assembly rules as well as IS development ontologies and frameworks can be used to obtain this fit.

5 Conclusions and further research

The S^3 model is intended to assist in selecting appropriate method fragments for a particular project. The method engineer may have to select additional method fragments due to situation factor values that cannot be covered by heuristics. The procedure associated with the model is one of the few contingency approaches giving actual guidelines *which* aspects to choose in a particular situation, aiming at a particular performance. However, the balance between the practical, relatively simply handling of the model and triviality is hard to find. For some applications, the model may be too fine-grained, whereas for other, it may be too simple. The valuation of the situation factors is related to this problem. Our starting-point has been, to formulate binary-valued situation factors, which may also be, in some cases, too simplistic. Another issue is the dynamic character of both situation factors and performance indicators, as some of them are likely to change during the project. At the moment, we only take into account the a priori situation and expected success.

To validate and iteratively adapt the S^3 model, we have designed a questionnaire, which will be sent to a significantly large number of experienced project managers. The questions in the questionnaire are based on the 20 heuristics. More further research includes the full integration of the procedure in our overall approach to Situational Method Engineering, and the automated support of the procedure in our CAME tool prototype.

Acknowledgements

The authors wish to express their thanks to the following persons, who were of great help in conducting their research. First of all, the substantial contribution Raoul van de Hoef provided by identifying the situation factors is acknowledged. Sjaak Brinkkemper put in, as one of the initiators of this research, a lot of his creative ideas. Richard Swart and Bram Schoonhoven performed the field studies at two large Dutch companies.

References

[Bennatan 92] Bennatan, E.M., Software Project Management: A Practitioner's Approach, McGraw-Hill, London, 1992.

[Boehm 88] Boehm, B., and P. Papaccio, Understanding and Controlling Software Costs. In: IEEE Transactions on Software Engineering, Vol. 14, No. 10, pp. 1462-1477, October 1988.

[Charette 89] Charette, R.N., Software Engineering Risk Analysis and Management, McGraw-Hill, New York, 1989.

[Dion 92] Dion, R., Elements of a Process-Improvement Program. In: IEEE Software, pp. 83-85, July 1992.

[Euromethod 94a] Euromethod, Concepts Manual 3 - Strategy Model, EM-3.3-SM, 1994.

[Euromethod 94b] Euromethod, Delivery Planning Guide, EM-2.3-DPG, 1994.

[Gilb 88] Gilb, T., and S. Finzi, Principles of Software Engineering Management, Addison-Wesley, Reading MA, 1988.

[Harmsen 94a] Harmsen, F., S. Brinkkemper, H. Oei, Situational Method Engineering for Information System Projects. In: Olle, T.W., A.A. Verrijn Stuart (Eds.), Proceedings of the IFIP WG8.1 Working Conference CRIS'94, Maastricht, pp. 169-194, September 1994.

[Harmsen 94b] Harmsen, F., and S. Brinkkemper, Formalisation and Representation of a Product-driven Approach to Method Assembly for Situational Systems Development. Memoranda Informatica 94-75, University of Twente, December 1994.

[Hoef 95a] Hoef, R. van de, F. Harmsen, G. Wijers, Situation, Scenario, and Success, Memoranda Informatica 95-12, University of Twente, February 1995 (*in Dutch*).

[Hoef 95b] Hoef, R. van de, and F. Harmsen, Quality requirements for situational methods. Accepted for the NGCT'95 workshop, Jyväskylä, Finland, June 1995.

[Humphrey 89] Humphrey, W.S., Managing the Software Process, Addison-Wesley, Reading MA, 1989.

[ISO 91] ISO/IEC 9126, Information Technology - Software Product Evaluation - Quality characteristics and guidelines for their use, International Standards Organization, 1991.

[Kumar 92] Kumar, K. and R.J. Welke, Methodology Engineering: A Proposal for Situation-Specific Methodology Construction. In: W.W. Cotterman, J.A. Senn (Eds.), Challenges and Strategies for Research in Systems Development, Wiley, 1992.

[Offenbeek 94] Offenbeek, M.A.G. van, P.L. Koopman, J.H.T.H. Andriessen, Flexibility in the approach to systems development - A choice model with a social-organisational perspective. In: Informatie, July 1994 (*in Dutch*).

[SERC 92] Software Engineering Research Centre, Specification of Software Quality: A practical guide, Kluwer Bedrijfswetenschappen, Deventer, the Netherlands, 1992 (*in Dutch*).

[Slooten 93] Slooten, K. van, and S. Brinkkemper , A Method Engineering Approach to Information Systems Development. In: N. Prakash, C. Rolland, B. Pernici (Eds.), Proceedings of the IFIP WG8.1 Conference on Information Systems Development Process, Como, 1993.

[Slooten 94] Slooten, K. van, and B. Schoonhoven, Towards Contingent Information Systems Development Approaches. In: Zupancic, J., and S. Wrycza (Eds.), Proceedings of the Fourth International Conference on Information Systems Development, Bled, Slovenia, pp.242-253, September 1994.

[Swede 93] Swede, V. van, and J.C. van Vliet, A flexible framework for contingent information systems modelling. In: Information and Software Technology, vol. 35, no. 9, pp. 530-548, 1993.

[Swede 94] Swede, V. van, and H. van Vliet, Consistent Development: Results of a first empirical study on the relation between project scenario and success. In: G. Wijers, S. Brinkkemper, T. Wasserman (Eds.), Proceedings of the 6th International Conference on Advanced Information Systems Engineering (CAiSE'94), pp. 80-93, LNCS 811, Springer Verlag, 1994.

[Vessey 94] Vessey, I., and R.L. Glass, Applications-Based Methodologies. In: Information Systems Management, pp. 53-57, Fall 1994.