NORTH-HOLLAND

# Contingent Information Systems Development

Kees van Slooten
*University of Twente, School of Management Studies, 7500 AE Enschede, The Netherlands*

Bram Schoonhoven
*RCC Information Services, 7300 HN Apeldoorn, The Netherlands*

Situated approaches based on project contingencies are becoming more and more an important research topic for information systems development organizations. The Information Services Organization, which was investigated, has recognized that it should tune its systems development approaches to the specific situation. A model has been developed, dealing with the matching between prevailing contingency factors and the preconditions of already existing situated approaches. Furthermore, a generic process model for systems development, including the information systems operations stage, is proposed. This model makes it possible to derive from it specific systems development strategies. A number of basic development strategies, specific for the Information Services Organization, are described. Preconditions, specific for this organization, are added to the standard situated approaches.

## 1. INTRODUCTION

In practice, the linear way of working during information systems development is abandoned, due to specific requirements of the specific situation. Different circumstances, due to different application domains, interest groups, business strategies, cultures and skills, require a different approach, a different collection of methods and tools, and the performance of a different set of development tasks in a different sequence. van Slooten et al. (1994) define contingency factors as follows:

> Contingency factors are circumstances of the project influencing in some way the selection or construction of an approach (or situated method) to systems development.

Situated method engineering has been defined by van Slooten and Brinkkemper (1993) as follows:

> The process of configuring a project scenario for information systems development using existing methods, or fragments thereof, to satisfy the factors that define the project context.

A scenario is an approach to the systems development process or, in other words, a situated method, determined by contingency factors. The configuration procedure proposed by van Slooten and Brinkkemper (1993) consists of the following stages:

- Characterization of the project by determining the important contingency factors and by deriving from these factors, so-called intermediate variables, namely: development strategy, ways of modeling (aspects), levels of detail and situational constraints. van Slooten et al. (1994) have mentioned two additional intermediate variables: the roles of the participants, and the relationship between the project and its environment.
- Composition of a first project approach by the selection of the most appropriate method fragments and the route map. Route maps are development strategy plans consisting of activities and

*Address correspondence to Dr. Kees van Slooten, University of Twente, School of Management Studies, P.O. Box 217, 7500 AE Enschede, The Netherlands.*

0164-1212/96/$15.00
SSDI 0164-1212(95)00186-7

products. The composition is supported by a computer-aided method engineering tool.
- Further refinement of the approach during the course of the project. The complete performance of the project leads to more development expertise, which is exploited by future projects.

The focus of this article is on the determination of the development strategy and under which conditions a particular development strategy is feasible, which means that the use of the term "approach" must be interpreted in this restricted way. We concentrate on the relationship between project characterization and development strategy.

The empirical research, a kind of field study, has been accomplished in cooperation with the staff of a Dutch information services organization in the domain of information systems development: RCC. In the past, RCC has developed computer-based information systems on a mainframe hardware platform; today, also on other platforms. The development of large-scale systems is supported by SDM, which is the adopted Systems Development Method. SDM is a widely used methodology in the Netherlands and follows a typical linear development strategy. RCC has indicated the following trends in information systems development:

- Increasing dynamics in the environment of the customer organization, which means that information must be more flexible and changeable due to changing requirements and circumstances.
- Increasing deconcentration and decentralization of the information service function, which means more diffusion in the organization of the development and use of information systems and more responsibility and decision making for the user.
- More automation also means a shift of knowledge about automation to the user organization with, as a consequence, more criticizing from the side of the user.
- The evolution of technology (hard- and software) is an important incentive for down- and right-sizing as well as approaches like prototyping.

Since RCC also develops systems on other platforms, there is a need for the development of more small-scale and flexible information systems. More decentralization and technological development necessitates the application of other approaches to systems development like prototyping and incremental development. New approaches to systems development cause a different way of working. Although the importance of new approaches to systems development has been recognized within RCC, the practice has not been changed yet sufficiently. However, today more alternative approaches besides the SDM approach are applied, which is caused by the following developments:

- Increasing use of CASE-tools makes it possible, necessary, or desirable to apply another approach.
- Increasing need for support of project teams using a prototyping strategy.
- RCC wishes to acquire more small-scale projects like client-server, PC and PC-LAN applications in the future.

This means that there is a need for more situation-specific project approaches, i.e., an approach accommodating the project context to support the project team. Furthermore, it is important for RCC to gain more experience with other approaches to the systems development process. A topic of our research is the development of tools enabling the project manager to select a proper approach. A model for the choice of an approach is elaborated in Section 2. A meta process model for the generation of situation-specific development strategies is discussed in Section 3. The basic approaches, described in Section 4, are based on this meta model for the primary process of systems development. These basic approaches are selected by comparing the project-specific contingency factors with a number of preconditions specific for a certain basic approach. Conclusions and further research can be found in the last section. Schoonhoven (1993) contains more detailed information about the field study.

## 2. THE DETERMINATION OF AN APPROACH

### 2.1 The Need for Contingency-Based Approaches

The linear approach, based on the waterfall model, is appropriate for the development of information systems when the specifications are stable and clear. However, new application domains are less structured and specifications become less obvious and are often subject to change. There are many other disadvantages related to the linear approach, e.g., the emphasis on phase products instead of the end product; organizational changes during the development process, which are often not considered. These and other factors have led to alternative development models, e.g., incremental development, evolutionary development, rapid prototyping, the spiral model (Boehm, 1986). Those nonlinear process models are supported by new technological development, e.g., CASE-tools and fourth generation languages. Through the emergence of different process models (approaches) and the situation-specific nature of each model, it is necessary to characterize the project, which enables the selection of the right approach.

The first contingency-based approaches have been developed by Naumann et al. (1980); Davis (1982); Burns and Dennis (1985). Baskerville et al. (1992) proceed in this direction and state that the selection of an approach, based on fixed criteria, is not sufficient. Furthermore, they argue that an approach to information systems development must be developed, just as an information system. In this way, approaches are emergent, which means that they have a short lifetime.

In practice, the relationship between situation and project approach is bidirectional (van Slooten et al., 1994), which means that the situation influences the project approach and vice versa. A contingency-based, situation-specific project characterization occurs preceding systems development (ex ante), but it also may occur and it usually occurs during project performance (on-the-fly). During the initial characterization, one might choose between two alternatives. On one side, one might choose the most appropriate approach from a set of available approaches (the selection alternative); on the other side, one might construct a completely or partially new approach accommodating the specific situation (the construction alternative).

### 2.2 A Process of Approach Determination

An approach will be considered as a coherent set of rules and characteristics indicating the way a systems development project is carried out. The focus of the characteristics is on the goal, the product, and the process of the information systems development project. An approach will be considered as a global strategy and not as a cook-book. Ould (1990) already represented development strategies through process-based models of the systems life-cycle, but this was not based on a generic model. Here, we also choose the process as the basis for the classification of information systems development projects. The availability of a number of possible project approaches is very useful for the practice of project management. Those classes of project approaches may contain, for instance, a specification of phases, the sequence of phases, a way of user participation, and possible pitfalls. Boehm (1989) has formulated this as follows: "What we really need are process model generators," which is exactly the subject of this article and is supported by our process of approach determination including our meta process model. We wish to describe a process for the determination of a project approach with the possibility to learn from experience.

We have already mentioned a few contingency-based approaches for the determination of a project approach. Furthermore, we wish to say a little bit more about two topics: risk analysis and the determination of critical success factors.

—Risk Analysis

Often one risk number is derived from a number of different contingency factors. A well-known example of such a calculation is published by Naumann et al. (1980). They propose four groups of contingency factors, from which a development strategy is determined through the calculation of a level of uncertainty. A low uncertainty number corresponds to an "accept user statement" strategy, a high uncertainty number corresponds to an experimental development strategy and a number in between corresponds to either a linear or an iterative development strategy. However, in practice, different risks might be solved in different ways. Sometimes an organizational solution is more appropriate. Furthermore, Naumann et al. (1980) made the implicit assumption that development strategy choices can be made independently of the source of the uncertainty. Moreover, the subjectivity of such calculations is a real danger. The conclusion is that such a risk analysis is only useful during a project characterization to obtain an initial, global idea about a possible development strategy, but a more specific project characterization is still necessary using specific knowledge from experience.

—Critical Success Factors

Critical success factors are entities, e.g., activities or conditions, which are crucial for achieving the project goal. After the determination of potential critical success factors, measures are taken in order to prevent problems. Critical success factors may play the role of contingency factors during the project characterization. van Slooten et al. (1994) emphasize the success or failure behavior of contingency factors and mention a few examples, e.g., team spirit, involvement of all participants, early and clear decision making.

The mentioned contingency approaches are at most partial solutions for the determination of a development strategy as a component of situation-specific information systems development. van

Slooten et al. (1994) state that we have to deal with a number of aspects of contingency factors during project characterization, namely: duration, direction, scope, deepness, origin, and mutual relationships. They also mention some problems with the determination of contingency factors: the unit of the factor, the visibility of the factor, the measurability of the factor and the stability of the factor. Also, Lyytinen (1987) mentions a number of existing problems of contingency approaches to systems development:

- The current contingency approaches are only partial solutions.
- The emphasis of existing contingency approaches is on measurable situation factors and use a strictly cause-effect interpretation scheme, which means that the cultural context is neglected.
- The proposed frameworks have an ad-hoc nature.
- Ambiguity arises if one tries to apply the models to specific development processes.

Figure 1 presents a model for the determination process of approaches to systems development. This is used by the remainder of this article and is actually a partial elaboration of the configuration procedure proposed by van Slooten and Brinkkemper (1993) restricted to the determination of a development strategy.

On one side, an approach might be determined based on dominant factors in the client situation (Figure 1), on the other side, based on preconditions that must be satisfied when applying a related possible standard approach (base of approaches in Figure 1). The standard approaches have already been proved useful in practice under the related conditions. In this way, an organization is able to build its own knowledge base or method base (van Slooten and Brinkkemper, 1993) with possible approaches, that is consulted in similar situations. To make a proper decision about an approach, there must exist a match between the dominant factors of that particular situation and the preconditions of the approach. A possible approach might be tuned to fit the actual situation better or, sometimes, the situation is conditioned with the intention to facilitate the application of a particular approach (van Slooten et al., 1994). In fact, we distinguish two options for the choice of a specific approach. One option is the use of a reasoning mechanism to bridge the gap between the contingency factors and a possible approach. The other option is the check of the preconditions of possible approaches for that particular situation. We think that we have to implement an integration of the two options. Both are necessary for the determination of a specific approach to information systems development.

## 3. THE META PROCESS MODEL

Many problems with information systems development projects arise from mismatches between the process model (e.g., waterfall, spiral model, prototyping) and project contingencies (e.g., budget, technology, customer standards, development expertise, and time). The primary process modeling approach to date, for avoiding these mismatches, is trying to develop one "best" process model that works well for any combination of project contingencies.

We propose to use a process model generator, our meta process model, making it possible to construct situation-specific primary process models. Our meta process model consists of two cycles of the primary process: the development cycle and the operations cycle (Figure 2).

The addition of the operations cycle makes it easier to describe completely the possible approaches, which is illustrated in the remainder of the article. Around the primary process of systems development, four different management processes can be arranged: project management, user manage-

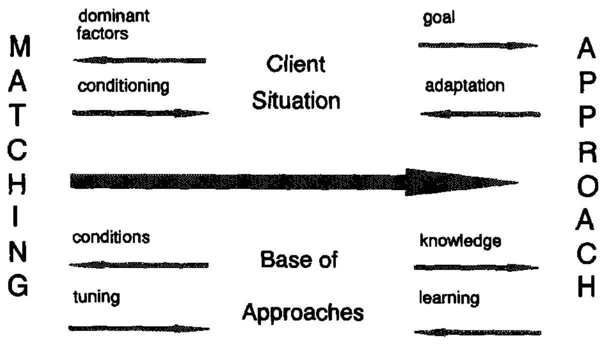

Figure 1. Approach determination.

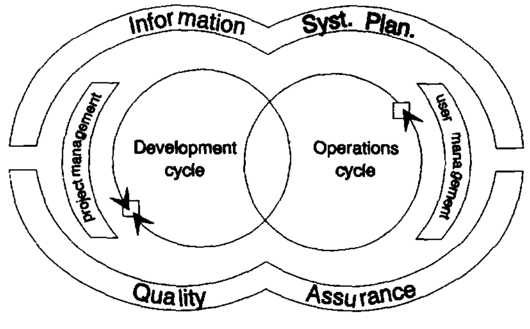

Figure 2. The meta process model.

ment, information systems planning, and quality assurance. The management processes may play an important role within the primary process, depending on the nature of the process.

Figure 3 contains a possible instantiation of the primary process, which is greatly self explanatory. The terms "object systems analysis and design (OSAD)" and "information systems analysis and design (ISAD)" have been precisely defined by van Slooten and Brinkkemper, 1993. Object systems analysis and design aims to design a new object system by articulating and solving the problems of the old object system. Information systems analysis and design aims to design a computer-based information system through the analysis of that particular part of the object system that has been selected by object systems analysis and design for that purpose.

Each approach to information systems development is an instantiation of the meta process model, e.g., different routes through the cycles with different goal/product combinations to realize, different actors, and different method fragments used by the processes. The different components of the information systems development process are described. Using these components, one can construct a specific process model: the route through the processes.

One can start a project from three possible starting points: information systems planning (e.g., new system), operations (e.g., maintenance) and development (e.g., prototyping, incremental). When a project is initiated, a process route is chosen through which a certain product must be delivered, and a goal must be achieved. In Figure 4, the different components of a complete process are given. Figure 4a shows the development of a subsystem or prototype; the results are demonstrated for future users, which means that the developed product is not used immediately, and the decision must be made for what must be developed during the next development cycle. Figure 4b shows the development of a "throw away after use" system, which means that the system will not come back to the development cycle for maintenance. Figure 4c shows incremental development. After the development of a part of the desired information system, this part will actually be used, and another part will simultaneously be developed. Figure 4d shows the development of a (part of a) system, that after it has been in operation for a while, returns to the development cycle, e.g., maintenance or slowly growing system. Figures 4e and 4f show the reverse engineering principle.

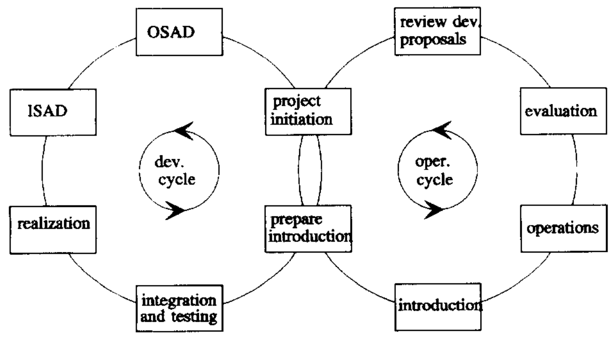

**Figure 3.** A primary process model.

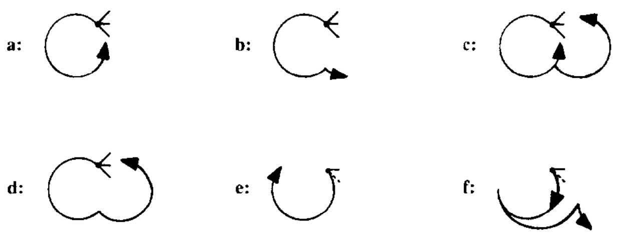

**Figure 4.** Components of the primary process.

The different components of the information systems development process are used to compose a project-specific process model. A model of the information systems development process is only one aspect of a complete approach. Other aspects, in order to investigate, are for instance, user participation, the role of quality assurance, and the selection of method fragments.

## 4. APPROACHES TO SYSTEMS DEVELOPMENT

During the field study at RCC the following basic approaches have been derived: phase-wise development, incremental development, stroke-wise development, evolutionary development, and reverse development. A few supplementary approaches are recognized for the specific situation of RCC, namely: package development and package selection, the construction of components and development for reuse, experimental development, and end-user computing. A supplementary approach cannot be used stand alone but is always supplementary to a basic approach. An approach can be applied, if a number of preconditions, belonging to the approach, have been checked and satisfied. The sets of preconditions have been acquired by interviewing the project managers and experts of RCC. A specific questionnaire has been developed to support this research.

## 4.1 Phase-Wise Development

The main variant of phase-wise development is strictly linear, phase after phase. This main variant is similar to the classical linear development model including some iteration, but the system does not return to the development cycle, except for maintenance. Such an approach does not allow intensive user participation and the application of formal planning and control techniques is usual. A disadvantage of this approach is the probably late detection of a possibly wrong route. Other variants are subsystems tile-wise, subsystems in parallel, and the development of throw away systems. The variants, "subsystems in parallel" and "subsystems tile-wise," are used to shorten the time needed for the development, depending on the availability of human resources. Preconditions for phase-wise development are:

- The specifications of the system are clear and stable. There is a clearly arranged project.
- How to realize the solution of the problem is clear and well known. There is no uncertainty about the success of the project.
- Formal decision making is desired and necessary. The project must be controlled.
- It is a critical system with strategic importance for the customer organization.
- The system has a long lifetime. Low maintenance expenses are expected.
- The user and the developer have enough domain knowledge available.
- The developers have enough experience with appropriate methods, techniques, and tools.
- It is not a small system.
- User acceptation of the system is not a problem.

## 4.2 Incremental Development

If a part of the system has been developed, it enters the operations cycle. At the same time, the development of another part of the system starts. Often, the nucleus of the system, including one or more subsystems, is developed firstly. After introducing that part of the system, the development of another part is started immediately. After realizing a part of the system, which must be integrated with other parts, it can be used by the user organization. The consequences for the user organization are less far-reaching and better manageable because the changes are realized in small steps. A disadvantage may be that more time is needed for the development of the system in comparison with phase-wise development. However, increased quality, faster return-on-investment, and the possibility to select the most urgent functionality may amply compensate this disadvantage. Preconditions for incremental development are:

- The timely delivery of the whole system is uncertain, and the customer accepts a subset of the system on a certain date. After that date, larger subsets of the system are delivered.
- Each time, it requires a lot of work to introduce the realized part of the system. It must be necessary to have a working subset of the system very quickly.
- The specification of the first part of the system must be clear and stable.
- Through the realization of the first part of the system, one may expect to acquire more clarity about the other desired parts of the system concerning time for development, difficulties, and required functionality. Usefulness and priority of the other parts of the system are discussed at the very beginning of the project.
- The quick delivery of a part of the system is required to get commitment and confidence of the

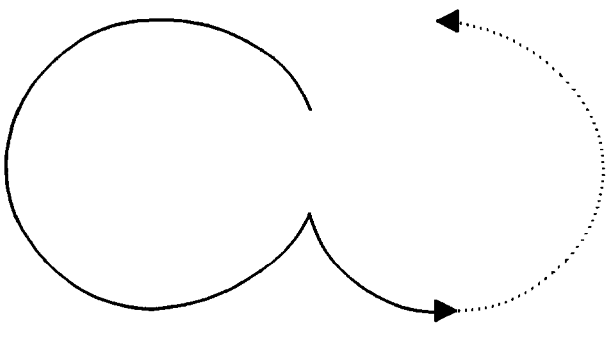
Figure 5. Phase-wise development.

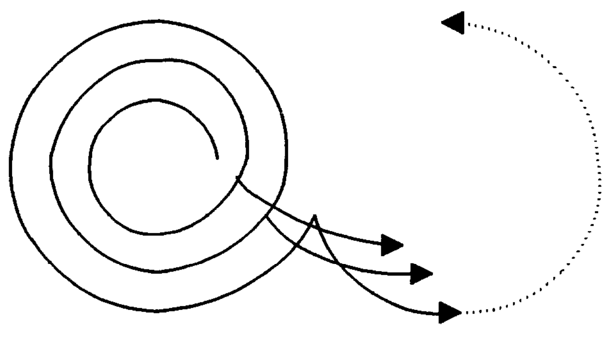
Figure 6. Incremental development.

users. Firstly, one must develop that part of the system from which increased user participation can be expected.

- The system must be large enough to be divided into a number of large parts that can be developed separately.
- If an incremental development strategy has been chosen for software package development, then the first version must offer enough functionality to get sufficient attention of the market.

### 4.3 Stroke-Wise Development

One stroke of the development cycle corresponds to the development of a part of the system. The development of the whole system takes place through a number of subsequent development strokes (one time the development cycle) based on former development strokes. Two variants of stroke-wise development exist: aspect systems development and subsystems development. Aspect systems development is similar to throw-away prototyping and subsystems development to keep-it prototyping. After every development stroke, it is checked whether the developed subsystem prototype satisfies the expectations and requirements of the users. The required changes are implemented during the next development stroke. The whole system is made ready for use during a final development stroke. Stroke-wise development is especially useful to reduce uncertainties, to evaluate intermediate products (subsystems) or to increase the involvement of users. Preconditions for stroke-wise development are:

- The specifications are unclear because the users do not know exactly what they need.
- It is unclear how the problems should be solved.
- The business processes are not stable or not well defined.
- Active user participation is necessary, and the users must have enough time/capacity to evaluate the outcome of each development stroke.
- The users must be able to communicate about information needs and information models.
- The development organization must be able to demonstrate their expertise.
- Tools for a quick realization of prototypes must be available.
- Developers have little experience with the development of similar systems.
- The users are able to criticize the functioning of the prototype.

### 4.4 Evolutionary Development

During evolutionary development, a complete system is developed, after which it is used. Based on experiences with the use of the system, further development is undertaken, which means that the system evolves. Every version or release of the system is complete, which means that the user is provided with sufficient functionality for the time being. Only one delivery is planned for the current project, which differs from incremental development. A final number of releases is not determined beforehand. Each version of the system may be developed using a different development approach. One can use, for instance, for the first versions, a prototyping approach to clarify the specifications, but later versions might be developed using a phase-wise approach because the basic specifications of the system have already been clarified by using a prototyping approach for the first versions of the system. The preconditions for the evolutionary approach are:

- There must be enough clarity about specifications to develop the first version, or prototyping is used to clarify these.

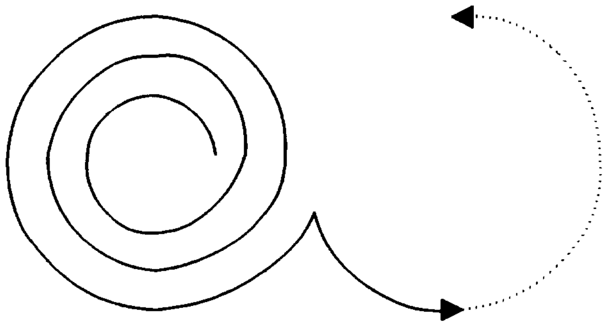

**Figure 7.** Stroke-wise development.

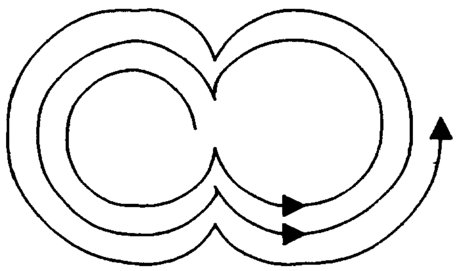

**Figure 8.** Evolutionary development.

- There must be a reason in order to believe that from the use of the system it will become necessary to realize important modifications in the next version. This may be the case if the system has an important impact on its environment, e.g., it changes the work procedures of the user, the interaction between systems development, and organization development.
- Similar to incremental development, it must be possible to enforce the introduction of the system several times.
- Low maintenance and operations expenses must not be a requirement for the project.
- Through the use of the system, new functionality must become emergent.
- This approach is also useful for package development. Assume that a package must be developed with complete functionality. Experience with the package in the market may lead to new functional specifications and critical remarks about the already realized functionality. Thereafter, the supplier of the package decides about the implementation of new or improved functionality in a new release.

### 4.5 Reverse Development

RCC has not gained much experience with reverse development or reverse engineering. The expectation is that reverse development will become more important in the future. More often, it is the case that a project does not start from scratch, but one starts from already existing systems.

The concept "reverse engineering" comes from hardware design, where the reconstruction of the design through analyzing the product is a common way of working. The same approach can be applied during information systems development by reconstructing higher level specifications from lower level specifications (binary code is the lowest level). This approach may simplify maintenance, the realization of modifications or replacement of components.

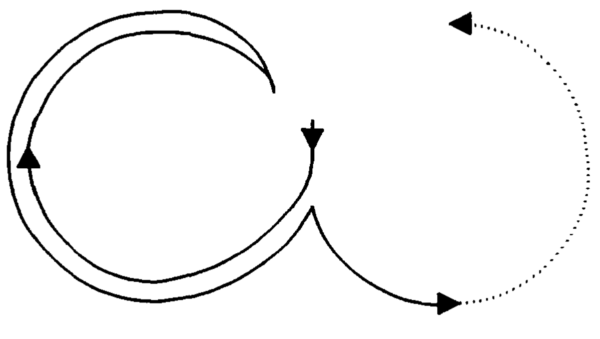
**Figure 9.** Reverse development.

Re-engineering may be followed by a forward engineering step to change the functionality of the system, or not. Preconditions for reverse development are:

- Reverse engineering may be applied for well-functioning systems with a low technical quality, e.g., ill-structured systems.
- Reverse engineering is also a feasible approach, if we need documentation for maintenance purposes or for the construction of a new system and adequate documentation is not available.
- Computer Aided Reverse Engineering tools must be available.

So far, we described the basic approaches to systems development. The supplementary approaches can be described in the same way, but we wish to stop here; otherwise, we get too lengthy a description. Of course, it is possible to construct hybrid forms of these approaches if the project situation makes it necessary. The idea is that each organization be able to build its own base of approaches using the models presented in this article and learning from experience.

## 5. CONCLUSIONS AND FURTHER RESEARCH

The project characterization, an important part of the overall situated method engineering process, has been experienced as an iterative process that may continue during actual project performance. Based on the project characterization, it may sometimes be possible to *select* an appropriate approach from already existing standard approaches, or it may be necessary to *construct* a new approach for the specific situation. The need to construct new approaches will decrease because, after an initial expansion stage, most situations will be covered. A learning process must be implemented, which makes it possible to keep the base of approaches up-to-date, to deal with lessons learned from practice, and to start the construction of new approaches if necessary.

The generic process model has been experienced as useful in practice for deriving situation-specific process models, including an appropriate development strategy for systems development. The development strategy is an important variable between the project contingencies and the actual approach with a strong influence on the route map. Adding preconditions and indicating feasible situations to standard process models facilitates the selection of a specific process model based on the project characterization. The process models and preconditions are developed by the organization, which means that within the organization arises an organizational memory that can be consulted for similar situations. This process of consultation will be supported in the near future by computer-aided method engineering tools (Harmsen et al., 1994).

The following topics are important for further research in the field (van Slooten, 1995):

- Eliciting prevalent project contingency factors. Evaluating existing methods and techniques for eliciting contingencies, and developing new validated methods and techniques. Better methods for eliciting contingencies will facilitate the project characterization.
- Relating project contingency factors to project approaches. Such investigations may yield new heuristic rules, which can be formalized and become a part of a computer-aided method engineering tool.
- Building a prototype of a base of situated approaches for various business situations. The availability of such a base can facilitate introducing situated approaches into organizations.
- A longitudinal study of one or more organizations implementing situated approaches. Such an implementation means an evolutionary change process providing more knowledge about the problems of this change process, how to solve those problems, and how to redesign development organizations.

## ACKNOWLEDGMENT

We are very grateful to the staff of RCC. They spent a lot of time facilitating the research and criticizing our reports. Also, the critique of Sjaak Brinkkemper and the comments of the reviewers were very helpful.

## REFERENCES

Baskerville, R. J., and Travis, D., Truex (1992), Systems Without Methods: The Impact of New Technologies on Information Development Projects, *IFIP 1992 Conference*, North-Holland.

Boehm, B. W., A Spiral Model of Software Development and Enhancement, *ACM SIFSOFT Software Engineering Notes*, Vol. 11, No. 4 (1986).

Boehm, B. W., What We Really Need Is Process Model Generators, *Proceedings ACM*, p. 397 (1989).

Burns, R. N., and Dennis, A. R., Selecting The Appropriate Application, *Database* (Fall 1985).

Davis, G. B., Strategies for Information Requirements Determination, *IBM Systems Journal*, Vol. 21, No. 1 (1982).

Harmsen, F. S., and Brinkkemper, H., Oei (1994). Situational Method Engineering for Information System Project Approaches, in *Proceedings IFIP WG 8.1*, Maastricht, 1994.

Lyytinen, K., Different Perspectives on Information Systems: Problems and Solutions, *ACM Computing Surveys*, 19 (1987).

Naumann, J. D. et al., Determining Information Requirements: A Contingency Method for Selection of a Requirements Assurance Strategy, *Journal for Systems and Software*, 1 (1980).

Ould, M. A., Strategies for Software Engineering: The Management of Risk and Quality, *Wiley Series in Software Engineering Practice* (1990).

Schoonhoven, B., Internal Technical Report on Situational Approaches (Dutch), University of Twente, 1993.

van Slooten, C., Situated method engineering, *Managing Information & Communications in a Changing Global Environment*, Idea Group Publishing, Harrisburg (1995).

van Slooten, C., and Brinkkemper, S., A Method Engineering Approach to Information Systems Development, *Information Systems Development Process*, IFIP WG 8.1, North-Holland (1993).

van Slooten, C., Brinkkemper, S., and Hoving, P., Contingency-based situational systems development in large organizations, *Managing Social and Economic Change with Information Technology*, Idea Group Publishing, Harrisburg, 1994.