# Tailoring and Testing of Process Models applied to the Vorgehensmodel

The ProcePT Approach

Hans-Ludwig Hausen
GMD-FIT-CSCW Sankt Augustin, Germany, hausen@gmd.de
Dieter Welzel
IABG Bonn, Gemany, welzel@iabg.de
www.scope.gmd.de

1997-02-20

**Abstract**

Activities to be performed and documents to be produced or modified are the basic elements for the software process description and are grouped according to different engineering aspects, such as development, quality assurance, configuration management or project management. Activities can be refined to processes using refined document structure or generalized to global procedures. Each resulting process can be tested, separately or in context with all of the other processes, whether it is enactable. Tailoring of a generic process model to a project-specific software development procedure is supported by refinement or abstraction. The trace of the tailoring process can be used to prove that the given project-specific software process is in compliance with a required process model. It is shown that a rule-based software process description can be effectively applied to program and test software processes according the German DoD software process standard Vorgehensmodell. Process requirements standards, such as ISO9000, are also tested in the context of model. The tailoring and testing of software processes are supported by the tool ProcePT implemented in PROLOG.

## 1 Introduction

Software quality and software productivity are key issues of the IT-industry. In order to improve the quality of a software product, the formation of the development process is important too. For this a project-specific software development process is specified by adapting activities and documents of a generic process model. This procedure requires that:

- software process models and their processes have to be formally described in order to test them, and
- adaptations to project-specific conditions and constraints have to be possible.

Figure 1 illustrates two possible ways of how to format a project-specific development procedure starting from a given process model. The following steps are used:

- *Formalization of the process model:*
  In order to describe several specific process models a rule-based approach is useful. Activities and documents, described by rules, can be used as basic elements to program software processes.

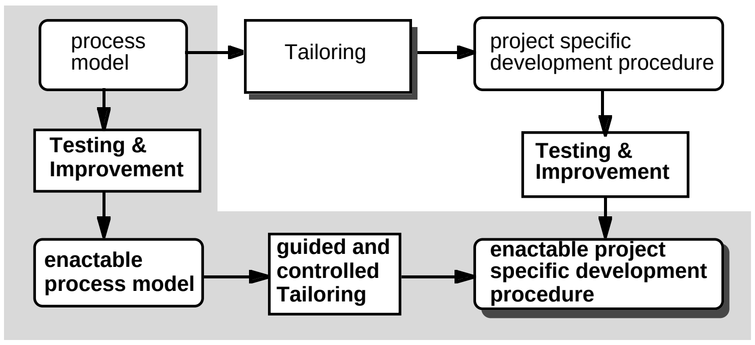
process model
Tailoring
project specific development procedure
Testing & Improvement
Testing & Improvement
enactable process model
guided and controlled Tailoring
enactable project specific development procedure

Figure 1: Tailoring Procedures

- *Testing and Improvement of the process model:*
  A process model should not contain inconsistencies in the way that its processes are not enactable. Analyzing the programmed structure of activities and documents helps to find out some theoretical mistakes in the dynamics which have to be removed in the "improvement" step.
- *Tailoring to a project-specific development procedure:*
  Using the programming and testing facilities an enactable project-specific development procedure can be specified. Normally, experience gained from previous projects as well as conditions and regulations (for example ISO900, SPICE or [ISO25]) would and should be taken into consideration. In this case, deletion conditions in terms of rules have to be reflected. This leads to a guided and controlled tailoring procedure.

As an example the German process model VORGEHENSMODELL (short V-Model, [BWB92] [BD93]) is used because it is now a standard for all software development projects where German authorities are involved. For specification of the V-Model and the testing and tailoring procedure production rules are taken (approach based on [Davi77], [Mins75]). Especially, for the specification and testing task the results of [Haus89a], [Haus89b] are used. Adaptations to other process models can be easily performed by just modifying them. The translation into PROLOG statements helps to validate the rules themselves and to perform the testing and tailoring of software processes. This led to the development of the prototype tool ProcePT (Process Programming & Tailoring) which has been applied for consultancy purposes.

## 2 Formalization of Process Models

Activities to be performed and documents to be produced or modified are the basic elements for a software process description and are described on different refinement levels and on different engineering aspects like software development or quality assurance. The specification of document flows of all activities on the indicated levels and aspects are important for enacting testing [Wel93a] and [Wel93b].

The V-Model describes the activities for software development and their accompanied activities of quality assurance, configuration management, and (technical) project management in different submodels shown in figure 2. The arrows indicate the interfaces between the submodels where documents are handled. Documents used in V-Model are called products.

Activities are described by product flow schemes (see table 1) which not only names input and output products, but also activities where products come from and where products go to. Products are described at different states. The values of a product state and the submodels which activities may change the state are shown in figure 3. The project management defines a product in state planned and the configuration management relates to it a configuration identification item. A product in the state in use is in process of filling its content. When it is ready for quality assurance its state changes to submitted. After assessing the product it becomes the state accepted if it has successfully passed QA, otherwise it goes back to the state in use, and has to be reworked. If however the project management decides to work further with

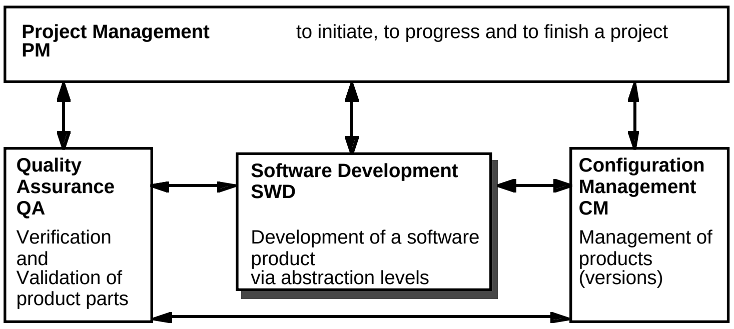
Project Management
PM
to initiate, to progress and to finish a project
Quality Assurance QA
Verification and Validation of product parts
Software Development SWD
Development of a software product via abstraction levels
Configuration Management CM
Management of products (versions)

Figure 2: Submodels of the V-Model

a product already accepted, a new version of the product must be taken with a new version number and it becomes the state in use. When a product comes from or goes to an external authority or has already been produced by a previous development process, it is assumed that the product is in the state present.

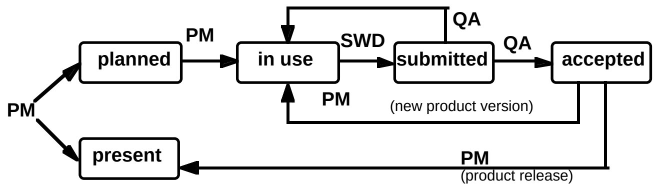

Figure 3: Product States and their Changes

An indication of QA or CM in the product flow scheme (see table 1) indicates that an activity of quality assurance or configuration management has to occur. Strokes by the fields 'in state' are used if no influence of activity exists for the product state, or if an input product is in state planned.

Table 1: *Product Flow Scheme of an Activity*

| | INPUT | | OUTPUT | | | |
|---|---|---|---|---|---|---|
| product | from activity | in state | to activity | in state | QA | CM |
| product1 | act1 | accepted | - | - | | |
| product2.*xxx* | - | - | act2 | in use | | |

$P.P_i$ indicates the subproduct (chapter) $P_i$ of the product $P$. If some $P.P_i$'s are input products and $P$ is the output product of an activity, the activity closes the product $P$ by collecting all its subproducts according to the product structure. The product flow scheme of an activity refers only those products which are 'really' used or produced by this activity.

If an activity $A_1$ produces or modifies a product for activity $A_2$ and this product is needed by activity $A_2$, the product flow have also to consider the changing of the product state or product version. A changing indicates that a QA or CM activity have occurred.

The V-Model is used as a basis for the data processing-technical handling of a project and can therefore be generally applied. In this it does not depend on where it is applied. This general applicability requires, however, that not all the inherent rules are relevant for any project. In order to apply the V-Model for a certain project it must first be decided which activities are required for the realization of the project and which products must be generated within the scope of the project. In order to support the deletion of activities and products (called tailoring) the V-Model contains a set of deletion conditions that are divided into deletion conditions for products and for activities. If a deletion condition is valid the product (activity) should be deleted from the set of products (activities) in order to receive the required project-specific V-Model. A deletion condition could influence other deletion conditions, for example, if a product can be deleted from the set of products the activity that has to produce it can also be deleted. An example is shown in Table 2.

Table 2: Example of Deletion Conditions

| object to be deleted | condition |
|---|---|
| Activity PM 1.1 | Software Maintenance Project <u>and</u> Project Manual available |
| Product Project Manual | Software Maintenance Project <u>and</u> Project Manual available |

Products which are available receive the state present.

## 3 Process Testing

The enacting testing consists of the reviewing of all activities whether they can occur, i.e. if the input products of an activity would be produced or modified in the needed state by the referred activity or if the output products of an activity would be used in the expecting state by the referred activity.

A process model specified by the rules proposed should be tested for enactability by the following steps:

T1 *testing product and activity structure:*
Does the product structure only contain products produced or modified by activities which are described in the activity structure? Are there no isolated products and activities?

T2 *testing product flows of activities:*
Is a product, which is produced in activity A1 for activity A2, also needed by activity A2? Is a product, which is needed in activity A2 expected from activity A1 also produced or modified by activity A1?

T3 *testing product state changes:*
Do the product flows of the activities accept the state changes?

T4 *testing interfaces between engineering aspects:*
Are activities of several submodels synchronised? Are products which are handed over identified?

T5 *testing refinements:*
Are each of the refinement levels enactable?

## 4 Tailoring

The adaptation of a process model to project-specific conditions and regulations is called tailoring.

The software process tailoring identifies the activities and products which can be deleted, deletes them and takes some required modification in the product flows in order to get the project-specific process model.

To specify the tailoring several steps are required (see figure 6):

(i) *Determination of valid deletion conditions:*
Based on a set of deletion conditions and a software process model *SPM* the activities and documents provided for tailoring have to identified. Because of the two kinds of deletion conditions (conditions for products and conditions for activities) two tailoring types are distinguished:

(ii) *Product tailoring:*
Products to be deleted should not be considered with the software process model *SPM'*. Products that should be seen as products coming from external sources or produced in a previous project should be considered in *SPM'* as products in the state present. Products

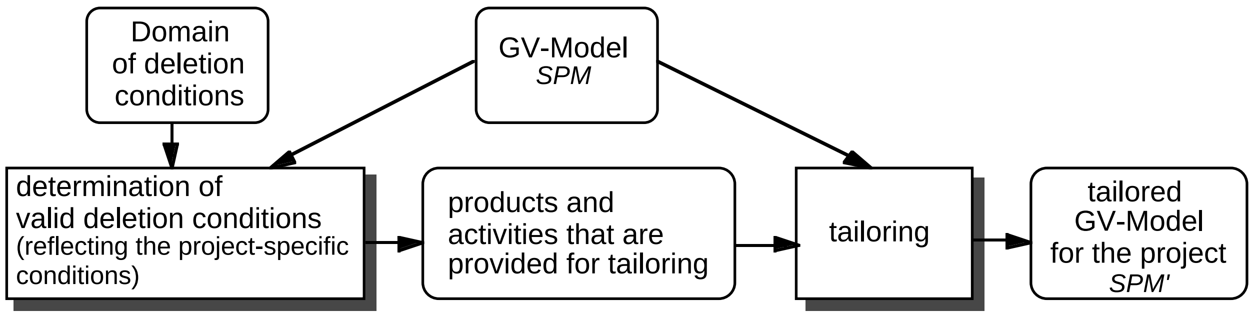
Domain of deletion conditions
GV-Model SPM
determination of valid deletion conditions (reflecting the project-specific conditions)
products and activities that are provided for tailoring
tailoring
tailored GV-Model for the project SPM'

Figure 4: Tailoring Procedure

in state present requires changes in the input-lists of the activities. Deleted products need modifications of references.

(iii) *Activity tailoring:*
Activities in *SPM* where all their output products are renouncable no longer exist in *SPM'*. References which contains the output products have to be deleted. Furthermore, references have to be 'bridged' if a product is only passed by the activity to be deleted

Starting from an enactable process model and using the product and activity tailoring should produce again an enactable tailored process model reflecting the project-specific conditions.

## 5 The ProcePT Tool

The production rules for the specification of the V-Model, for the testing and tailoring have been translated to PROLOG III predicates. Because the tailoring procedure is implemented slightly differently to the specified rules the implemented procedure is described. All deletion conditions are shown sequentially in order to determine whether they are valid.

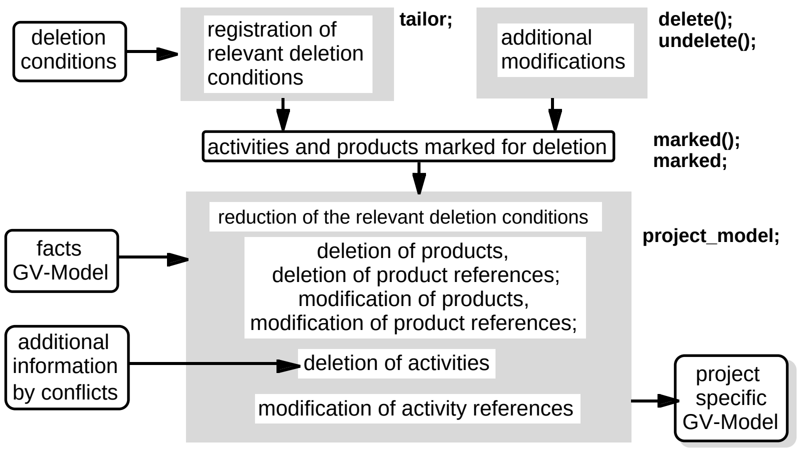
deletion conditions
registration of relevant deletion conditions
tailor;
additional modifications
delete();
undelete();
activities and products marked for deletion
marked();
marked;
reduction of the relevant deletion conditions
project_model;
facts GV-Model
deletion of products, deletion of product references; modification of products, modification of product references;
additional information by conflicts
deletion of activities
modification of activity references
project specific GV-Model

Figure 5: Tailoring Procedure implemented in ProcePT

For each deletion condition the consequences to other activities or products are indicated. The decision whether a deletion constraint is valid could be postponed. After determining all relevant deletion conditions (PROLOG predicate *tailor;*), the same modifications to the set of relevant deletion conditions can be made directly by applying *delete();* for deletion or *undelete();* for withdrawal. All activities and products marked for tailoring including their reasons for deletion can be shown by variations of predicate *marked*.

The adaptation is called by the predicate *project_model;* that performs:

- Product tailoring
  (i) *Deletion of products:*
  Products to be deleted are removed from the set of facts.

(ii) *Deletion of products in product flow schemes:*
The identifiers of deleted products are removed in the product flow scheme of all activities.

(iii) *Modification of products:*
Products that already exist and should be deleted are not to be removed from the set of facts but must change their state to `present`.

(iv) Modification of products references
Products in state `present` requires changes in the input-references of the activities. Deleted products need modifications of products identifiers in each activity.

- Activity tailoring

(i) *Deletion of activities:*
Activities marked for deletion with empty output-lists can be removed from the set of facts. When marked activities have no empty output-lists, conflicts may arise. Products of output-references can be expected by other activities. Through an interactive dialogue with the process designer all conflicts should be solved.

(ii) *Modification of activity references:*
Deleted activities need modifications to references of other activities.

The result of the tailoring is the enactable project-specific V-Model. This description is part of the project handbook.

## 6 Conclusion

A rule-based specification of software process models has been shown. The German process model (V-Model) has been taken as an example. The specification of the V-Model indicates a lot of mistakes in the informal description of the V-Model. After removing these mistakes an enactable basic V-Model was built on which further research has been made. With the rules proposed for tailoring of software process models a tailored enactable process model can be derived. The procedure has been described by the following steps:

- Starting from an enactable software process model
- Collecting all deletion conditions relevant for the software development project
- Testing their consequences
- Modifying deletion conditions if inconsistencies are indicated by the test
- Removing activities and products identified for deletion from the selected software process model
- Modifying remaining activities and products
- Editing the resulting project-specific software process model

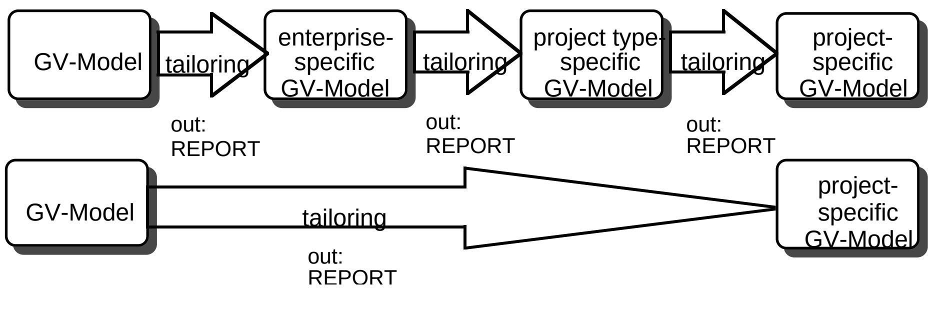
GV-Model
tailoring
enterprise-specific GV-Model
tailoring
project type-specific GV-Model
tailoring
project-specific GV-Model
out: REPORT
out: REPORT
out: REPORT
GV-Model
tailoring
project-specific GV-Model
out: REPORT

Figure 6: Tailoring in one or more steps

Starting from an accepted specification of the V-Model several profiles of software development process models can be generated. If constraints and regulations of an enterprise are taken into consideration an enterprise-specific V-Model can be defined. Form this, using the tailoring facility again, a project-specific V-Model might be specified.

Furthermore, V-Models for project types can be tailored from the V-Model considering constraints and regulations (special standards) for projects of the same type. Examples are

- small/medium-sized/large administrative IT projects
- small/medium-sized/large technical/scientific IT projects
- projects for selection, procurement and adaptation of off-the-shelf-products

## References

[—] References and further Readings.

[AS93] Ashworth, Caroline; Slater, Laurence; *An Introduction to SSADM Version 4* (International Software Engineering Series); ISBN 0-07-707725-3; McGraw-Hill International, London, 1993.

[BWB92] BWB General Directive 250; *Software Development Standard for the German Federal Armed Forces, V-Model, Software Lifecycle Process Model*, Bundesamt für Wehrtechnik und Beschaffung, FE VI 2, Box 7360, Koblenz, Germany; August 1992.

[BD93] Bröhl, A.-P.; Dröschel, W. (eds.); *Das V-Modell: der Standard für die Softwareentwicklung mit Praxisleitfäden*, ISBN 3-486-22207-04, Oldenbourg Verlag, München, Wien,1993.

[Davi77] Davis, R.; Buchanan, B.; Shortliffe, E. *Production Rules as a Representation for a Knowledge-Based Consultation Program.* Artificial Intelligence, 8,15-45, 1977.

[ISO25] ISO/IEC Guide 25, *General Requirements for the Competence of Calibration and Testing Laboratories*, International Organization for Standardization, International Electrotechnical Commission, 1990.

[Mins75] Minsky, M. *A Framework for Representing Knowledge.* In Winston, P., editor, The Psychology of Computer Vision, pages 211-277, McGraw Hill, New York, 1975.

[Haus89] Hausen, H.L. *Rule-Based Handling of Software Quality and Productivity Models* in: C. Ghezzi J.A. McDermind (Eds.) ESEC'89 2nd European Software Engineering Conference 1989, University of Warwick, Coventry, UK, September 1989, LNCS 387, Springer Verlag, Berlin, p. 376-394.

[Haus92] Hausen, Hans-Ludwig: A Specification of an Assessment and Certification Advisor, in: Annual Conference of the ACM, March 1992, Kansas City, MO, 20 pp

[Haus89a] H.L. Hausen: Knowledge Based Invocation of Software Methods and Tools. in: N. G. Bourbakis (ed.) Proceedings of the IEEE Workshop on Tools for Artificial Intelligence Fairfax, Virginia, October 23-25, 1989

[Haus89b] H.L. Hausen: Rule-Based Handling of Software Quality and Productivity Models. in: C. Ghezzi J.A. McDermind (Eds.) ESEC'89 2nd European Software Engineering Conference 1989 , University of Warwick, Coventry, UK, September 1989, LNCS 387, Springer Verlag, Berlin, p. 376-394

[Rei85] Reisig, Wolfgang; *Petri Nets An Introduction*, Springer-Verlag Berlin Heidelberg New York Tokyo, 1985.

[Wel93a] Welzel, Dieter; *Specifying and Evaluating Causal Dependencies within Software Engineering Processes*, in: Kafka, P.; Wolf, J., editors, Safety and Reliability Assessment - an Integral Approach - (Proceedings of ESREL'93, Munich, May 10th-12th, 1993), Elsevier Science Publishers, Amsterdam, 1993.

[Wel93b] Welzel, Dieter; *A rule-based process representation technique for software process evaluation*, Information and Software Technology, Vol 35, No 10, October 1993.