VDA QMC

German Association of the Automotive Industry
Quality Management Center

Joint Quality Management in the Supply Chain

Automotive SPICE®

Guidelines

Process assessment using the Automotive SPICE® PAM 4.1

3rd revised edition, March 2026

Online-Download-Document

Automotive SPICE®

Guidelines

Process assessment using the Automotive SPICE® PAM 4.1

3rd revised edition, March 2026

Verband der Automobilindustrie e.V. (VDA)

German Association of the Automotive Industry (VDA)

Online-Download-Document

Non-binding VDA recommendation

The German Association of the Automotive Industry (VDA) recommends that its members apply the following VDA volume when introducing and maintaining quality management systems.

Exclusion of liability

This VDA volume reflects the technical procedures known at the time of its respective edition. The contents therefore represent the knowledge and views of the authors at the time of publication. Although the information has been prepared with the greatest possible care, no claim is made for factual accuracy, completeness, and/or timeliness; in particular, this publication cannot take into account the specific circumstances of individual cases. Use of the information is therefore at the reader's own responsibility. Applying the VDA recommendations does not relieve anyone of responsibility for their own actions; all act on their own responsibility.

Liability for the content and application of this VDA publication and for the persons involved in preparing the VDA recommendations is excluded. Anyone who encounters inaccuracies or the possibility of misinterpretation when reading or applying these VDA recommendations is requested to inform the VDA immediately, so that any deficiencies can be corrected.

Translations

The English document is the original. In the event of interpretive questions in other language versions, the English version shall be referred to as the original. This publication will also appear in other languages. The current version can be obtained from VDA QMC.

Verband der Automobilindustrie e. V. (VDA)
Qualitätsmanagement-Center (QMC)

10117 Berlin, Behrenstrasse 35
Germany
www.vda-qmc.de

Preface

Market demands require permanent innovations with increasing complexity within reliable time frames. It is essential to continually improve the development processes and methods for product creation and ensure that stakeholder quality expectations are met.

The VDA reworked the existing Automotive SPICE® Guidelines version 1.0 based on the Process Assessment Model Automotive SPICE® 4.0 released in conjunction with this Blue-Gold-Book. This was made to take appropriate steps to improve the quality and comparability of assessment results.

Major improvements are made regarding addition of new domains like Hardware Development and Machine Learning. The rating guidelines are provided for all processes in the process assessment model and a new recommended scope for assessments has been determined.

The Blue-Gold-Book “Automotive SPICE® for Cybersecurity” of 2021 remains a separate document.

The “Automotive SPICE® Process Assessment Model” is increasingly used within the global automotive industry for the objective evaluation of processes and the subsequent improvement of processes at a project and organization level. It shall not be misinterpreted as a development methodology. The objective in drawing up this document was to support the interpretation and application of the model for the automotive industry and provide guidance and recommendations to increase the comparability of assessments results.

This document is aimed to support a mature and sustainable development within the automotive industry.

Content

List of Figures

List of Tables

Terms and Glossary

In the following, definitions of terms used in the present volume are provided. When applicable, a citation of the definition provided in the ISO/IEC 330xx process assessment series of standards is given in italic letters.

Please refer to ISO/IEC 33001:2015 *[ISO33001]* for a full glossary of the terms used in the ISO/IEC 330xx series.

Term	Definition
Assessing organization	The organization which performs the assessment.
Assessment log	The formal documentation of the execution of an assessment drawn up by the assessor. The assessment log is the evidence of the assessor's assessment activities and is provided to the certification body.
Assessment scope	*Definition of the boundaries of the assessment, provided as part of the assessment input, encompassing the boundaries of the organizational unit for the assessment, the processes to be included, the quality level for each process to be assessed, and the context within which the processes operate.* *→ [ISO/IEC 33001:2015, 3.2.8]*
Assessment sponsor	*Individual or entity, internal or external to the organizational unit being assessed, who requires the assessment to be performed and provides financial or other resources to carry it out.* *→ [ISO/IEC 33001:2015, 3.2.9]*
Assessment team	*One or more individuals who jointly perform a process assessment.* *→ [ISO/IEC 33001:2015, 3.2.10]*
Assessor	*Individual who participates in the rating of process attributes.* *→ [ISO/IEC 33001:2015, 3.2.11]*
Audit	*A systematic, independent and documented process for obtaining audit evidence [records, statements of fact or other information which are relevant and verifiable] and evaluating it objectively to determine the extent to which the audit criteria [set of policies, procedures or requirements] are fulfilled.* *→ [ISO 19011]*

Term	Definition
Automotive SPICE®	A process assessment and reference model conformant to the requirements of ISO/IEC 33004:2015 *[ISO33004]*. It is primarily addressing the development of embedded software-based systems within the automotive domain. It can be downloaded free of charge on *www.automotivespice.com*.
AUTOSAR	**AUT**omotive **O**pen **S**ystem **AR**chitecture: an initiative by the automotive industry for standardization of software in electronic control units (www.autosar.org).
AUTOSAR domains	Categories used to classify electronic control units by their area of application, e.g., chassis, powertrain, telematics, body.
Capability level	Point on a scale of achievement of the process capability derived from the process attribute ratings for an assessed process.
Certification body	A central body that administrates the certification information of the trained assessors and classifies these assessors by their qualifications and practical experience according to a certification scheme.
Certification scheme	A set of rules and procedures used by a certification body to certify assessors.
COTS	Commercial Off The Shelf
Evidence	Artifact or information reflecting practice performance. Evidence is used during assessments to understand process performance and can comprise documents, oral information, data or information from tools or other sources.
Evidence repository	Repository for storing obtained evidence.
Feedback presentation	A process step at the end of the assessment, when the assessment team provides feedback on the results. It usually covers the main strengths and potential improvements. The set of provisional process capability profiles is also presented if appropriate.
Findings	The evaluations documented by assessors regarding strengths and potential improvements of the assessed organizational unit, based on verbal affirmations from interviews and work products presented (→*Evidence*).
FOSS	Free and Open-Source Software
Indicator	*Sources of objective evidence used to support the assessor's judgment in rating process attributes.*

Term	Definition
	→ *[ISO/IEC 33001:2015, 3.3.1]*
Lead Assessor	*Assessor who has demonstrated the competencies to conduct an assessment and to monitor and verify the conformance of a process assessment.* → *[ISO/IEC 33001:2015, 3.2.12]*
NDA	**N**on-**D**isclosure **A**greement
OEM	"**O**riginal **E**quipment **M**anufacturer." In the automotive industry this term is used to describe the vehicle manufacturers. (See also →*Tier 1…n*).
Organization assessed	The organizational unit which is assessed. This usually refers to projects in one or more departments in the assessed organization.
Practice level	Lowest level of granularity within the Automotive SPICE® process assessment model, determined by the "base practices" and "generic practices" of the processes. Strengths and potential improvements should be traceable to this level and are derived from expectations regarding a state-of-the-art implementation of the practices.
Process assessment model (PAM)	*Model suitable for the purpose of assessing a specified process quality characteristic, based on one or more process reference models.* → *[ISO/IEC 33001:2015, 3.3.9]*
Process capability	A characterization of the ability of a process to meet current or projected business goals.
Process context	*Set of factors, documented in the assessment input, that influence the judgment, comprehension, and comparability of process attribute ratings.* → *[ISO/IEC 33001:2015, 3.2.16]*
Process measurement framework	*Schema for use in characterizing a process quality characteristic of an implemented process.* → *[ISO/IEC 33001:2015, 3.4.6]*
Process (capability) profile	*Set of process attribute ratings for an assessed process.* → *[ISO/IEC 33001:2015, 3.2.18]*
Process quality characteristic	*Measurable aspect of process quality; category of process attributes that are significant to process quality.*

Term	Definition
	→ *[ISO/IEC 33001:2015, 3.4.9]*
Process reference model (PRM)	*Model comprising definitions of processes in a domain of application described in terms of process purpose and outcomes, together with an architecture describing the relationships between the processes.* → *[ISO/IEC 33001:2015, 3.3.16]*
SPICE	Name of the starting project, elaborating the draft of ISO/IEC TR 15504. These days the term “SPICE” is used colloquially to refer to ISO/IEC 330xx. In PAM 4.0 the acronym stands for Software-based **S**ystems **P**rocess **I**mprovement and **C**apability d**E**termination.
Tier 1…n	The term “Tier 1…n” is used to refer to suppliers at various levels in the supply chain. Direct suppliers to the OEM are referred to as “Tier 1”, a supplier to a Tier 1 supplier is referred to as a “Tier 2”, etc.
VDA	“**V**erband **d**er **A**utomobilindustrie”, the German Association of the Automotive Industry

Introduction

The intent of this publication is to revise the Blue-Gold Book Automotive SPICE® Guidelines version 2 to improve the quality and reproducibility of assessment results.

The objective is to give necessary clarifications and recommendations for the application of Automotive SPICE® for the purpose of performing assessments and monitoring of resulting process improvements in the development of software-based systems.

To fulfill this mandate, the following activities were performed:

- Improving the Automotive SPICE® Process Assessment and Reference Model regarding structure, inconsistencies, clarifications and additional concepts. This was done with the publication of the 4.1 version of Automotive SPICE® PRM/PAM *[AS41]*.

- Improvement and update of the guidelines on the interpretation of Automotive SPICE® and on Assessment Performance. This is provided by the current publication.

- Setting requirements for the qualification of assessors and updating the existing procedures, training materials and examinations. This will be done in collaboration with the international assessor certification scheme (intacs®) to accompany the release and roll-out of this publication *[intacs]*.

The current publication will replace the existing Blue-Gold Print Automotive SPICE® Guidelines version 2 and can be applied with its official publication in the VDA QMC online shop.

The present publication addresses the mandate by providing two parts:

Part 1: Interpretation and rating guidelines

This part provides rules for the rating performed in an assessment.

Part 2: Guidelines for performing the assessment

By defining the requirements for the assessment process, it is intended to standardize the procedure, so that the companies involved in an assessment can follow a defined assessments approach. This present volume specifies the requirements related to the assessment process, as well as the qualification of assessors carrying out assessments based on Automotive SPICE.

All rules for rating in assessments reflect best practices from assessors having extensive experience in assessments based on Automotive SPICE® in various applications.

Besides the knowledge of the participating members and third-party members involved, the present publication leverages other sources giving valuable input, which has been proven in many years of assessment practice and assessor trainings. For particulars see bibliography.

Document scope

The scope of the current document is to support assessments using Automotive SPICE®. It addresses the process of performing the assessment and in detail the rating performed in an assessment. It is based on the 4.0 version of the Automotive SPICE® Process Assessment and Reference Model.

Automotive SPICE® 4.0 is a full process assessment model (incl. reference model) complying to the requirements of ISO/IEC 33004 *[ISO33002]*. It can be used on its own to perform assessments. The intention of this publication is NOT to replace or extend the Automotive SPICE® PAM or PRM.

The guidance given in Part 1 of this document is intended to support reproducible assessment results but cannot reflect all the variety in practicing engineering, management and supporting processes. Assessment teams need to understand the context of the assessed organization before they judge a rating rule from this document to be applicable. Lists and enumerations that supplement the process related guidance need not be interpreted as checklists for implementation and are not intended to be complete. References to rating rules must not be used as weakness statement to justify a rating of an indicator or a process attribute.

The guidelines are also applicable for other PAM having the same measurement framework (e.g., Automotive SPICE for Cybersecurity).

The aim of the Part 2 of this document is to set guidelines for the application of Automotive SPICE® to assist the assessors while planning, executing and reporting the assessment. Furthermore, it specifically addresses the improvement process to resolve issues found in an assessment.

The target audience is predominantly assessors who are active in the automotive domain but can also be seen as an additional input for assessments in other domains. It also addresses other parties or roles involved or affected by an Automotive SPICE® assessment such as the assessing organization, the assessed organization or the assessment sponsor.

In addition, the document is intended to support the understanding of the assessment process and should be taken as a basis for clarification in case of dissent about the result of an assessment.

Relation to ISO/IEC 330xx series

The ISO/IEC 330xx series of international standards define the requirements and resources needed for process assessment. Several standards in the ISO/IEC 330xx family were intended to replace and extend parts of the former ISO/IEC 15504 series.

ISO/IEC 330xx process assessments are conducted based on three core elements:

- process models that define processes, the entities that are the subject to assessment;
- process measurement frameworks that provide scales for evaluating specified attributes, and
- a specification of the process to be followed in conducting assessments.

The intention of the working group 13 of the VDA QMC was to provide a domain specific set of documents covering these three elements for performing assessments conformant to ISO/IEC 33002 *[ISO33002]*. This has been achieved

- by providing the Automotive SPICE® process reference and assessment model *[AS41],*
- by defining a measurement framework derived from ISO/IEC 33020:2019 *[ISO33020]* for assessment of process capability using the Automotive SPICE® PAM, and
- by submitting a documented assessment process conformant to ISO/IEC 33002 *[ISO33002]* in Chapter 6 of this volume.

Relation to Automotive SPICE®

At the beginning of the development of Automotive SPICE® 4.0, different extensions to the previous version 3.1 were evaluated by the working group 13 to provide an updated process set suitable for assessments in the automotive domain. Contrary to the version 1.0 of the Guidelines for Automotive SPICE® this documentation covers all processes in the process assessment model.

Since the scope of application has been enlarged to cover different engineering domains, the working group 13 decided to define a new recommended scope for performing assessments. This was done to avoid increasing the effort involved in performing an assessment and provide reproducibility of assessment results.

It is a principle of process assessments according to the ISO/IEC 330xx series that the process scope (the selection of processes to be examined in an assessment) might be adapted in agreement with the sponsor and with respect to the purpose of the assessment.

Part 1: Interpretation and Rating Guidelines

1 Application of Interpretation and Rating Guidelines

1.1 Overview

The purpose of part one of the current publication is to support the assessors in interpreting the Automotive SPICE® process reference and assessment models and rating the process attributes for the given target capability level.

These recommendations are based on the extensive experience of the assessor community. Most of the assessments in the automotive domain do not address capability levels higher than 3. Therefore, no rating rules are provided for level 4 or 5 due the limited amount of experience in application of these levels.

Chapter 1, "Application of Interpretation and Rating Guidelines" introduces a clearer definition of how-to set-up and consider the assessment purpose and scope input and provides an overall guideline on rating in an assessment.

An integral part of the interpretation and rating guidelines are rules addressing specific key concepts, application environments and the different capability levels.

Chapter 2, "Key Concepts and Overall Guidelines" gives rules related to key concepts introduced or modified with the 4.0 version of Automotive SPICE®. Further, rules for rating in specific application environments are provided.

Chapter 3, "Rating Guidelines on Process Performance (Level 1)" is related to the process specific outcomes, base practices and output information items associated with the capability level 1. In this chapter, specific rating rules are given for each process of Automotive SPICE®.

Chapter 4, "Rating Guidelines on Process Capability Level 2" gives specific rating rules for each process attribute of level 2.

Chapter 5, "Rating Guidelines on Process Capability Level 3" gives specific rating rules for each process attribute of level 3.

Chapter 6 "Understanding Capability Level 4" and ***Chapter 7 "Understanding Capability Level 5"*** have been introduced to give guidance for higher levels and cover the entire capability dimension of Automotive SPICE®.

1.2 Assessment purpose

Automotive SPICE® assessments are performed within a certain variety of use cases for specific purposes. In general, the purpose of process assessment is to understand and assess the processes implemented by an organizational unit.

Specifically, as defined in ISO/IEC 33001 *[ISO33001]*, 3.2.6 the assessment purpose is a

> *"statement provided as part of the assessment input, which defines the reasons for performing the assessment."*

Note: The assessment purpose may not be confused with the process purpose.

Based on this definition, the assessment purpose needs to be documented when identifying the assessment input (see also *8.2.2 Assessment inputs*).

Additionally, it is strongly recommended to include the assessment purpose in the assessment report (see chapter *8.4.2 Formal information about the assessment*).

The assessment purpose may be documented by specifying the objectives of the assessment, such as:

- determining the process profile to set a baseline for further improvements
- providing evidence for Production process and Product Approval (PPA) if required
- determining the process profile to evaluate a supplier

The assessment scope (see next chapter) shall be defined to cover the assessment purpose, accordingly.

1.3 Defining the assessment scope

As defined in ISO/IEC 33001 *[ISO33001]*, 3.2.8 the assessment scope shall provide

"the definition of the boundaries of the assessment, provided as part of the assessment input, encompassing

- *the boundaries of the organizational unit for the assessment,*
- *the processes to be included,*
- *the quality level for each process to be assessed and*
- *the context within which the processes operate (process context)."*

Note: ISO/IEC 33001 [ISO33001] uses the term "quality level." Since Automotive SPICE® applies only capability levels as a specific implementation of a quality level, the term "capability level" is used throughout the process assessment model and within this guideline.

1.3.1 Defining the boundaries of the organizational unit

As defined in ISO/IEC 33002 *[ISO33002]*, the boundaries of the assessed organizational unit according to the definition in ISO/IEC 33001 *[ISO33001]* shall be given in the assessment scope. The definition of the organizational boundaries shall be given in terms of

- the localization of the involved organizational unit(s) and
- the responsibilities of the involved organizational unit(s).

These boundaries shall always be defined with respect to the defined processes (see subchapter 1.3.2) and the defined process context (see subchapter 1.3.4).

In summary, the boundaries shall identify which part of the organization is responsible for the performance of the given processes in the scope and provide information about the location of the development sites. This is a necessary input for the planning of the assessment.

1.3.2 Defining the processes to be included

It is a principle of process assessments according to the ISO/IEC 330xx series that the process scope (the selection of processes to

be examined in an assessment) might be adapted in agreement with the sponsor and in accordance with the purpose of the assessment. Identifying the processes “under scope” is a significant step to tailor the content of the Automotive SPICE® assessment model to cover the assessment purpose in terms of the specific development scope of the project assessed.

The recommended VDA process scope provides a standard selection of processes that are recommended to give a comprehensive overview of an assessed project.

It consists of a set of base processes, including core supporting processes and the project management process. This set of processes needs to be supplemented by at least one set of engineering processes addressing a specific development domain within the project (plug-in). Depending on the disciplines involved in the development, the base scope may be combined with plug-ins and other processes from Automotive SPICE® (flex scope) to enable a needs-based coverage of the assessment purpose.

In specific cases, other process reference models may also be considered.

The recommended VDA Scope is based on the release 4.1 of the Automotive SPICE® process reference and assessment model *[AS41]*.

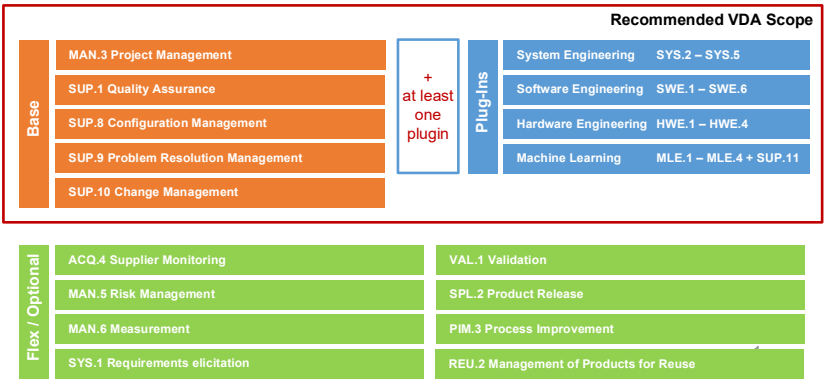


Figure 1-1: Recommended VDA Scope and optional processes

For certain use cases, the recommended VDA Scope may also be further tailored. A typical example could be focusing on some specific aspects of the development to identify process specific improvement opportunities only.

If the processes to be assessed deviate from the recommended VDA Scope, a rationale shall be documented for choosing this specific set of processes with respect to the purpose of the assessment.

Each process in the defined assessment scope shall be assessed and the result documented in the assessment report. To ensure a sufficient basis for rating, each process defined in the scope must be performed by the assessed organization/project at least once.

Exceptionally there might be the need to exclude or add processes after agreement of the assessment scope, such as during the execution of the assessment. Any exclusion of processes from the rating with respect to the agreed scope shall be documented as a modification of the assessment input and shall be approved by the sponsor of the assessment. A classification of a process as “not applicable” without approval from sponsor is prohibited.

A simple non-existence of evidence shall not be a valid justification of an exclusion of a process from rating. Usually, in such cases an N rating is recommended.

In Figure 1-2 an example is given for an exclusion of processes during assessment conduct.

Initial Scope requested by sponsor

ID	Process Name	Target Capability Level
ACQ.4	Supplier Monitoring	1
SYS.2	System Requirements Analysis	1
SYS.3	System Architectural Design	1
SYS.4	System Integration and Integration Verification	1
SYS.5	System Qualification	1
SUP.1	Quality Assurance	1
SUP.8	Configuration Management	1
SUP.9	Problem Resolution Management	1
SUP.10	Change Request Management	1
MAN.3	Project Management	1

Adapted Scope agreed with sponsor

ID	Process Name	Target Capability Level
~~ACQ.4~~	~~Supplier Monitoring~~	~~1~~
SYS.2	System Requirements Analysis	1
SYS.3	System Architectural Design	1
SYS.4	System Integration and Integration Verification	1
~~SYS.5~~	~~System Qualification~~	~~1~~
SUP.1	Quality Assurance	1
SUP.8	Configuration Management	1
SUP.9	Problem Resolution Management	1
SUP.10	Change Request Management	1
MAN.3	Project Management	1

Report of assessment

ID	Process Name	Achieved Capability Level
SYS.2	System Requirements Analysis	1
SYS.3	System Architectural Design	1
SYS.4	System Integration and Integration Verification	1
SUP.1	Quality Assurance	1
SUP.8	Configuration Management	1
SUP.9	Problem Resolution Management	1
SUP.10	Change Request Management	1
MAN.3	Project Management	1

Figure 1-2: Recommended VDA Scope and optional processes

1.3.3 Defining the target capability level for each process

Since the measurement framework used in Automotive SPICE® is applicable for rating capability, the term "*capability level"* as a refinement for the *"quality level"* is used. There are five capability levels specified in the PAM for the assessment. As mentioned before, the rules given in this publication are only considering capability levels 1 to 3, because this covers most of the Automotive SPICE® assessments in the automotive domain.

Since the planning of the assessment is significantly influenced by the choice of the target capability level, the intended maximum capability level to be assessed shall be defined for each process as part of the assessment scope.

1.3.4 Defining the process context in the assessment scope

In ISO/IEC 33001 *[ISO33001]*, 3.2.16 the process context is defined as

> *"the set of factors, documented in the assessment input, that influence the judgment, comprehension, and comparability of process attribute ratings."*

When defining the process context, the boundaries within which the processes operate in terms of a set of aspects shall be identified.

Exemplary aspects for boundaries to be combined for defining the process context are:

- functional/content related
- time/release related
- requirements/change related

Examples:

- all past releases, a specific release or a selection of specific product releases
- since a specific point in time (to address new processes, organizational changes)
- including or excluding platforms/standard components
- including or excluding open source

- all requirements and changes valid for a specific product release
- all requirements and changes excluding particular product releases
- all requirements and changes related to a defined architectural element
- all requirements and changes to be implemented between two defined project milestones
- all changes and affected stakeholder requirements in a (delta) project developing additional functionalities based on an existing system or software
- complete system delivered by a supplier
- a complete software platform delivered by an internal or external organization

Note: The definition of the process context needs to be aligned with definition of the boundaries of the organizational unit (see subchapter 1.3.1).

Each process attribute rating shall strictly remain within the boundaries of the process context in the assessment scope.

1.3.5 Defining instances when setting up the assessment scope

Depending on different constraints, the same process might be applied in different process instances within the same project, like for parts that are developed using model-based approaches in comparison to parts that are manually coded. Therefore, different process attribute ratings might be derived for different instances of the rated process. The corresponding rating methods are provided in the measurement framework of ISO/IEC 33020 *[ISO33020]*, subchapter 5.4.

There are different use cases where a separation of a process into instances may be reasonable. Building instances may reflect the need of a higher granularity of the assessment findings due to the execution of the process with different approaches or in separate organizations or locations.

Setting up instances does not change the given scope and process context of an assessment. If instances are defined, they all shall be rated according to the given scope, and the rules must be applied on each process performance attribute rating of each single instance.

To provide a more detailed understanding of the term “process instance,” the following exemplary use cases are given:

- A project used standard process version 2 until March 2023, and standard process version 3 since then. If the assessor can clearly see that the usage of these two standard process versions actually do not overlap, a reasonable instantiation may be:
  - a rating of process instance “SWE.3 until March 2023”
  - a separate rating of Process instance “SWE.3 after March 2023”
- Parties responsible for different hierarchical levels in the architecture of a mechatronic product development project use different requirements engineering approaches, such as:
  - a rating of process instance “SYS.2/Mechatronic level”
  - a separate rating of process instance “SYS.2/ECU level”
  - a rating of process instance “SWE.2/Application SW level”
  - a separate rating of process instance “SWE.2/Basis SW level”
- Different reuse strategies used for different parts of the overall SW, like
  - a rating of process instance “SWE.x/Platform code”
  - a rating of process instance “SWE.x/Project specific developed code”
- Different SW development paradigms are used for different parts of the overall SW, for example
  - a rating of process instance “SWE.3/Model-based development”
  - a rating of process instance “SWE.3/Manual coding”
- Different sub-projects use different project management approaches, including
  - a rating of process instance “MAN.3/SW level”
  - a separate rating of process instance “MAN.3/Overall project”
- Different organizational units develop different parts of the software. These organizational units might even be in different geographical locations and regions, such as

- a rating of the process instances “SWE.x/Standard SW components in the reusable platform – Asia”
- a rating of the process instances “SWE.x/Standard SW components in the reusable platform – Europe”
- a rating of the process instances “SWE.x/All customer-/application-specific SW components – Germany”

Reasons for assessing different process instances separately can be meaningful, like

- to have company-internal benchmarking
- for a more accurate understanding of the various characteristics in the organization to better launch precise process improvement initiatives.

The ratings of the process attributes for each process instance shall be documented in the assessment report.

In case several instances are defined, a process is rated independently for each instance, thus resulting in separate ratings of the process. This requires an aggregation of the results to a single process attribute rating considering the impact of the instance on the overall rating. The recommendations how to perform the aggregation can be found in subchapter 1.4.4.

1.4 General rating practice

1.4.1 Rating of process attributes

1.4.1.1 Rating according to ISO 33002

According to the ISO/IEC 33002:2015 *[ISO33002]*, the rating of process attributes is mandatory.

To achieve a higher degree of reproducibility and consistency in rating of process attributes the ISO/IEC 33020 [ISO33020] provides the following optional approach:

> *Process outcomes and process attribute outcomes may be characterized as an intermediate step to providing a process attribute rating. (ISO/IEC 33020:2019 [ISO33020], clause 5.4)*

At the end of each process attribute there are tables with the relationship of process indicators (practices and information items) to process attribute outcomes. The characterization of process attribute outcomes can be derived using these relationships.

Although practice rating is not mandatory, it can improve reproducibility of the outcome characterization. It is common practice to establish traceability between the objective evidence and the rating of process attributes.

This common practice is supported by many rating rules. On one hand they refer to significant weaknesses (downrate) that may have a negative impact on process attribute rating, and on the other they should avoid misinterpretations of the indicators (not to be downrated) and not impact process attribute rating negatively. In subchapter 1.5 the semantics and application of rating rules are described

A process attribute rating that is conform to ISO 33002 and ISO 33020 can be performed using two approaches:

a) rating of the process attribute based on objective evidence using the rating scale

b) rating of practices based on objective evidence to characterize achievement of process attribute outcomes resulting in process attribute rating

In case of direct rating of process attributes, a violation of rating rules for downrating has to be identified as a weakness if applicable. In this case, the rating rules for "… shall not be downrated" (e.g., **TAC.RL.1)** will be considered anyway.

Formally, the assessment indicators (base and generic practices, and IIC) must be used to support the assessor's judgment in rating process attributes.

The assessment sponsor decides which approach shall be selected and documented in the assessment plan.

1.4.1.2 Consideration of information items

As described in the Automotive SPICE® assessment model, information items (II) including their characteristics (IIC) serve as a second type of assessment indicators. They are provided as guidance for "what to look for" in the documentation available in the assessed organization.

The extent of implementation of an information item (in line with its defined characteristics) in a work product serves as objective evidence supporting the assessment of a particular process. Information item characteristics should be considered as indicators when considering whether, given by the context, a documented information is contributing to the intended purpose of the process.

Please refer to the process assessment model for further understanding of information items and their relation to work products produced by the organization assessed.

1.4.2 Independent rating of processes

A process assessment model provides a two-dimensional view of a process quality characteristic. Each process within the scope (process dimension) shall be rated individually on the scale provided within the capability dimension. The rating will be based on the

objective evidence presented in the assessment independent of the assessment purpose.

This means that only process weaknesses subject to rating shall be the source of a potential downrating. This implies that only BPs explicitly referring to another process (e.g., the consistency/traceability BPs) can be downrated due to a weakness in that other process, because these are the only “connection points” between processes.

[GEN.RL.1] A rating of PA 1.1 of P or N for a process X shall not be used to downrate PA 1.1 of the process Y.

1.4.3 Sampling of work products for rating

The selection of the work products must be carried out carefully to ensure that work product samples are representative, comprehensive, and provide evidence of the implemented process.

1.4.3.1 Selection of work product samples

The following aspects apply for the selection of work products:

- coverage of the most important functions relevant for the assessment scope
- coverage of new functionality, adapted functionality, reused and platform content according to the assessment scope
- coverage of the whole spectrum of ASIL levels and security relevant content within the assessment scope
- coverage of manual coding (all programming languages used) and model-based development (all modelling tools used), where applicable

Metrics (e.g., number of requirements, cyclomatic complexity, lines of code, number of change requests) can support the selection of work product samples. It can be useful to select units with different complexity to sample the corresponding detailed designs.

For the engineering processes the following approach is recommended: The assessor chooses stakeholder requirements based on above-mentioned aspects. The work products selected for evaluating the indicators of the processes should mark a clear path

through the engineering lifecycle. The same approach should be applied when evaluating supporting processes like change management or problem management.

Although the assessed organization may propose certain work products, it remains the assessor's decision to which extent they are considered for the process attribute.

In all cases the number of work product samples selected shall be representative to cover the given assessment purpose and scope.

1.4.3.2 Plausibility checks of work product samples

All documents used as candidate for objective evidence must be checked for consistency, in terms of plausibility of the last change time stamp and appropriateness of the change history. The latter can be easily checked by inspecting the history of the work product in the respective tool which is used for configuration or document management. If a document was initially generated shortly before the assessment, it should not be considered for the rating of the process attribute in question – unless there is a plausible reason for the late documentation.

The history of the work product should show an appropriate lifecycle and several versions which correlates with the update cycle of the respective work product.

For instance, if a schedule should be updated on a weekly basis then at least one version per week could be expected (or some evidence that an update was unnecessary). Technical documents tend to have more versions than plans. However, if the architecture is based on a platform, there may not be that many versions. It is up to the assessors to check whether the number of versions reflects appropriately the lifecycle and status of the project and fulfills the purpose of the process attribute which is assessed.

1.4.3.3 Content-related examination

The content-related examination of the work products should always cover the whole scope of the assessment.

This means the whole scope shall be represented based on the criteria for selecting the work products samples.

Given the time constraints it is typically not possible to cover all aspects of the project. Nevertheless, the samples shall also be checked regarding the right content. For the content of information items, the information item characteristics can be used as guidelines.

The system requirements for example are not only to be checked for traceability to the stakeholder requirements but also if the system requirements reflect the intention of the stakeholder requirements. Another example would be to check the unit tests against the detailed design. The engineer should explain the detailed design. The unit tests are then checked against the detailed design. Inconsistencies found between the test cases and the explanation of the detailed design shall be considered when rating the process attributes.

Automotive SPICE® must not be mistaken for a checklist. The assessor has the duty to check appropriate instantiation of documentation to cover the different process attributes. Appropriateness is based on, for instance, the scope, size and complexity of the project team (e.g., distributed development), the size and complexity of the product, the timeline, and other influencing factors as defined in the process context.

1.4.4 Aggregation of process attribute ratings

Using the rating method R2 from ISO/IEC 33020 *[ISO33020]* for the rating of each process attribute is recommended.

This means,

1) to rate each process attribute for each process within the scope of the assessment for each process instance and
2) aggregating the process attribute ratings of the process instances.

An aggregation of the process attribute ratings of all process instances is mandatory. This means, in the assessment report there will be one additional set of process attribute ratings for the aggregation.

The aggregation is done according to the following schema ("one dimensional aggregation using arithmetic mean" according to ISO/IEC 33020 *[ISO33020]*):

1) In accordance with ISO/IEC 33020 *[ISO33020]* NPLF, rating values can be expressed as interval values as

   N → 0; P → 1; L → 2; F → 3,

   with rounding the result to the nearest integer (by rounding up or down) and converting the result back to the corresponding ordinal rating. Rounding rules are:

   - Rounding down to the nearest integer, if the average value is less than the midpoint between consecutive integers
   - Rounding up, if the average value is at or above the midpoint between consecutive integers

2) The aggregation can be done

   a. by calculating an arithmetic mean, or
   b. by assigning these internal values a percentage weighting first and then converting them back to the ordinal NPLF rating scale. Weightings and their rationale must be explained in the assessment report, and may depend on, e.g., the
      - size of personnel of organizational unit/sub-project,
      - strategic significance of the product (e.g., commodity vs. new innovative products),
      - its contribution to the revenue in %, and the
      - criticality of product parts (e.g., a risk class according to ISO 26262 *[ISO26262])*.

	Process instance **A**	Process instance **B**	Process instance **C**	Aggregated rating
2a. Arithmetic mean without any weighting of process instances	**L (2)**	**L (2)**	**F (3)**	(2+2+3) / 3 **→ L (2.33)**
	P (1)	**L (2)**	**F (3)**	(1+2+3) / 3 **→ L (2)**
	N (0)	**P (1)**	**F (3)**	(0+1+3) / 3 **→ P (1.33)**

2b. Arithmetic mean with weighting	**L (2)** 70%	**L (2)** 15%	**F (3)** 15%	(2*0.7+2*0.15+3*0.15) → **L (2.15)**
	P (1) 70%	**L (2)** 20%	**F (3)** 10%	(1*0.7+2*0.2+3*0.1) → **P (1.4)**
	N (0) 30%	**P (1)** 20%	**F (3)** 50%	(0*0.3+1*0.2+3*0.5) → **L (1.7)**

Each row represents a process as defined in the assessment scope.

1.5 Semantics and application of rating rules

1.5.1 Objective

Rating rules are intended to reduce variance in rating decisions across assessors because of different individual interpretation of assessment indicators and rating dependencies. This is seen as one of the key factors to improve the quality, reproducibility, and comparability of assessment results.

1.5.2 Rule semantics

A rating rule (RL) in this guideline is defined as a directive on how to rate indicators. These can be

- conditional (i.e., dependent on a specific context or domain such as software, firmware, hardware, mechanical engineering, machine learning etc.). A condition can refer to:
  - some context-specific scenario
  - the rating of an individual indicator
  - the rating of particular indicators
  - the ratings across particular indicators;
- or unconditional (i.e., irrespective of any specific context or domain such as software, firmware, hardware, mechanical engineering, machine learning, etc.).

In case the assessor needs to deviate from an RL, a compelling justification must be documented in the assessment report.

Examples of unconditional rules:

> **[CL2.RL.xx]** SUP.1.BP3 shall not be rated higher than the ratings across GP 2.2.4 of all processes.

Examples of conditional rules:

> **[SUP.10.RL.xx]**
> If the analysis omits to address potential side effects, the indicator BP2 shall not be rated F.

[AGE.RL.xx] If the software architecture is modified incrementally and impact analyses evidence that changes were discussed, then SWE.2.BP1 shall not be downrated.

1.5.3 Rating rule formulation patterns

	Wording	Explanation
1	If <condition>, then X shall not be downrated.	Condition can refer to: • some context-specific scenario • the rating of an individual indicator B • the rating of particular indicators • the ratings across particular indicators 'X' can refer to a single or a set of indicators.
2	If <condition>, then X shall be downrated.	Condition can refer to: • some context-specific scenario • the rating of an individual indicator B • the rating of particular indicators • the ratings across particular indicators 'X' can refer to a single or a set of indicators. The indicator(s) shall be downrated for at least one step of the rating scale. It is the decision of the assessor, if a further downrating is necessary to reflect further identified weaknesses.
3	X shall not be rated higher than <condition>.	Condition can refer to: • the rating of an individual indicator B • the rating of particular indicators • the ratings across particular indicators 'X' can refer to a single or a set of indicators.

1.5.4 Further instructions for the application of rating rules

1.5.4.1 No “rating rule algebra”

There might be cases in which for rating of a process attribute, or assessment indicator, different rules apply in parallel. However, the application of *n* different rules, each requiring a downrating, must not automatically lead to a downrating of this indicator exactly *n* times according to the NPLF scale. It remains the responsibility of the lead assessor to decide on the final rating value considering the actual context, gathered objective evidence, and identified process risk.

There may be rules that define for two indicators A and B in the same process “*If A is downrated, then B shall be downrated*.” Still, how many NPLF steps B actually needs to be downrated also depends on the actual context, gathered objective evidence, and identified process risk.

1.5.4.2 Assessment report and record

Normally, a rating rule cannot replace comprehensive weakness statements in the assessment report: any downrating, or when detecting weaknesses within the percentage range of the "Fully” value, requires providing a comprehensive explanation of the associated process risk, substantiated by traceable objective evidence. Omitting or neglecting comprehensive and comprehensible weakness statements in favor of referring only to rating rules is not sufficient and therefore renders the entire assessment report invalid. A rating rule may only support a given comprehensive and comprehensible weakness statement and a given rating. Here, the lead assessor is responsible.

2 Key Concepts and Overall Guidelines

2.1 No production or construction processes

This PRM/PAM does not define a process or assessment indicators for production processes. To avoid redundancies and potential inconsistencies with other international standards having production in scope such as IATF 16949 or VDA 6.3, PRM and PAM counterparts of production processes are not included at all.

Correspondingly, there is no process for prototype and sample construction/workshops either.

For these reasons, “process interfaces” to the production domain are required. In the hardware processes this is achieved by means of

- output information items characteristics for HWE.2:
  - 03-54 Hardware Production Data including
    - 14-54 Hardware bill of material
    - 04-55 Hardware layout
  - 17-57 Special Characteristics, and
- the “Ensure use of compliant samples” BP, including comprehensive notes, for HWE.3 and HWE.4.

2.2 No procurement process

No procurement is introduced in this PRM/PAM for the following reasons:

- Hardware development is requirements-driven, too. Therefore, what matters is compliance to the requirements for the respective environment, irrespective of the source from which HW or mechanical parts are obtained. Verification (HWE.3, HWE.4, SYS.4, SYS.5) will demonstrate that the physical product or sample is compliant with the design and with the requirements, respectively.
- There are other standards related to procurement such as IATF 16949.

A “process interface” to procurement can be considered existent by means of BP “Develop hardware detailed design” in HWE.2, together with Note 3.

2.3 Specific terms used in base practices

Processes in Automotive SPICE® are passed several times within the project lifecycle. This iterative work concept is considered in the description of the processes (except: ACQ.2; Automotive SPICE® for Cybersecurity).

Consequently, there is no hierarchical or temporal dependency for base practices and processes. The base practices do not imply a certain sequence, hierarchical order or pattern. They are primarily listed in a logical order.

Therefore, also continued re-evaluation of work products and work packages is necessary in certain processes (e.g., MAN.5, MAN.3).

2.3.1 Technical scope of the HWE processes

The technical scope of the HWE processes is electrical or electronic hardware engineering. This excludes

- system level engineering, i.e., neither the mechatronic nor the ECU level (see also the definition of the term “hardware” in the glossary),
- procurement (see subchapter 2.2),
- mechanical or hardware sample manufacturing (see subchapter 2.1), and
- production processes (see clause, subchapter 2.1).

However, process interfaces are included to

- procurement in terms of receiving physical design-compliant hardware parts,
- production and prototype/sample workshops in terms of providing information such as production data and requirements, and receiving compliant samples, respectively.

2.3.2 The scope of “system” in SYS.x

The scope of the SYS processes can be interpreted in a generic way, meaning they are not tied to a particular system boundary. This also signifies that the Automotive SPICE® PRM/PAM does not represent a product hierarchy. Rather, via different process instances the SYS.x processes may represent a “system of systems” or different levels of a single product, such as a mechatronic system or drive (motor plus ECU, or an ECU).

The system boundary for:

1. A **mechatronic system supplier** or **drive (i.e., motor plus ECU) supplier** would be the mechatronic product supplier. Both the mechatronic system boundary and the ECU system boundary would be reflected by separate process instances of the SYS processes in a decomposed manner. To the ECU system boundary within the mechatronic system the considerations in Figure 2-1 apply.

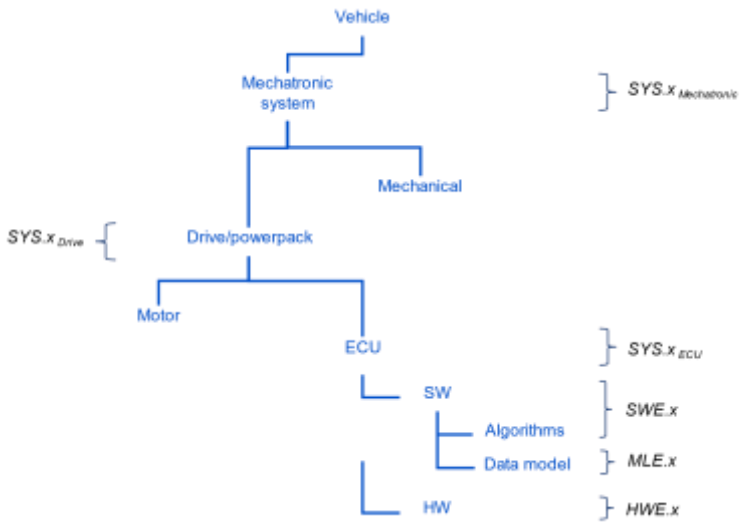


Figure 2-1: Possible use of process instances to represent a mechatronic product composition

2. A **control device supplier** would be the ECU supplier. This system boundary can also be reflected by the SYS processes because it typically comprises hardware, software, housing, connectors, etc. In consistency with the scope of this document, the HWE processes should then be used to reflect development of the fully assembled PCB. In this respect, the definitions of “hardware part” and “hardware component” in this document apply. See also Figure 2-1.
3. A **semiconductor supplier** would be, e.g., a microcontroller or a system-on-chip supplier. This system boundary should be reflected in the Automotive SPICE® SYS processes because besides hardware it typically comprises a mechanical housing, firmware, etc. The HWE processes should then be used to reflect the hardware-related parts of this system. Note that in this context, the definition of “hardware part” and “hardware component” can represent ISO/IEC 26262’s notions of “hardware subpart” and “hardware elementary subpart.” See also Figure 2-2.

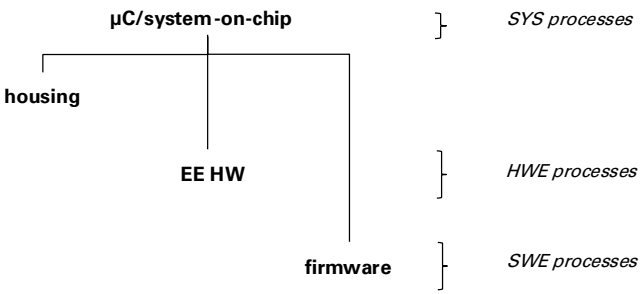


Figure 2-2: Possible use of process instances to represent a microcontroller or system-on-chip

4. A “**software system.**”

A further scenario is a coherent software comprising different pieces of software, each running on a different node and/or target. Sometimes, only the overall software behavior is in focus; hence, the nodes and targets being considered transparent. Some people refer to this as a “software system.”

A seemingly obvious approach could be not to address such a “software system” via SYS.x in favor of SWE.x, as it is about software. Indeed:

- Overall software black-box behavior could be addressed via SWE.1.
- Logical and technical software interfaces between the different pieces of software in such a “software system” could be addressed via SWE.2.BP1; the technical interfaces behind memory-mapped IOs or microcontroller registers such as cables, connectors, or bus connections in between would not need to be considered in SWE.2.
- Logical interactions could be addressed as SWE.2.BP1.
- The “interior” of each piece of software could be addressed via SWE.3 and SWE.4.

However, this view causes problems when it comes to SWE.5. The software requirements will (as demanded by SWE.1.BP1) include non-functional expectations like response times or processing time limits. This indeed serves as meaningful input for software integration verification. However, such timing requirements cannot be achieved and verified without considering the nodes and targets as these also will consume time budgets. Depending on the technical realization, these time budgets could even differ. Similar issues arise when discussing SWE.6.

Further example:

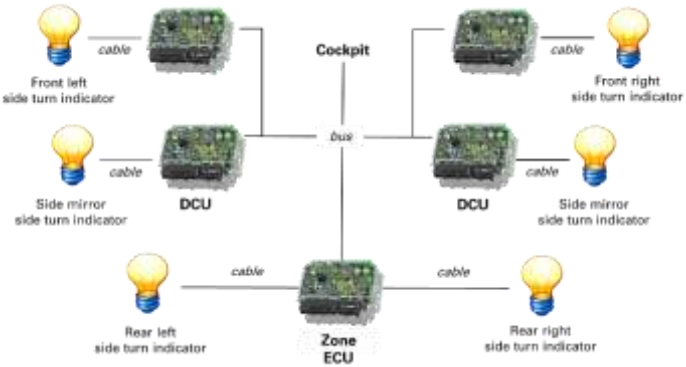


Figure 2-3: Example architecture

Consider the software-controlled synchronous blinking of hazard warning lights (Figure 2-3). This cannot be viewed as a sole “software system” while neglecting the hardware targets: during bus-off the different pieces of software cannot communicate. Therefore, they must have established a common pulse scheme beforehand as otherwise during bus-off the synchronous flashing cannot be consistently maintained. The hardware, however, is subject to tolerances, and tolerances can change over time (e.g., because of thermal influence), potentially resulting in asynchronous flashing.

Therefore, the notion of “software system” still requires using the SYS.x level as network nodes and hardware targets are, in fact, relevant. The SWE.x processes alone do not appear appropriate to address such scenarios.

2.3.3 Requirements process oriented concepts

2.3.3.1 Characteristics of requirements in SYS.2, SWE.1, HWE.1

The original motivation of having an extra Verification Criteria BP in Automotive SPICE® was that, according to the requirements engineering state-of-the-art, a requirement shall be documented in a

verifiable way, otherwise it does not represent a requirement. The former extra Verification Criteria BP was supposed to emphasize that. However, this has introduced PAM misunderstandings:

1. Consider ratings such as:

- BP1 “Specify Reqs” = F
- BP4 “Ver Criteria” = N or P

It is difficult to argue how requirements as a whole (BP1), which have *not* been formulated in a verifiable way (BP4 = N/P), can be rated as F. Further, non-verifiable requirements even put in question how the entire process purpose can be regarded as being fulfilled.

2. The distinct verification criteria BP appears to suggest that isolated information documented separately from requirements would be necessary. However, verification criteria are inherent in a requirements statement:

Example 1:

> #1 “The ECU shall be able *to process at least 5 speed update bus messages within 1 [s] with a tolerance of +0.2[s]”*

Example 2:

> #1a “The ECU shall be able to *process vehicle speed update bus messages*”
>
> #1b “When receiving vehicle speed update bus messages, the ECU shall be able *to process the bus message within 1 [s] with a tolerance of +0.2[s]”*
>
> (The texts in italics are an example of information needed to make the requirement verifiable)

3. The Automotive SPICE® Guidelines v1.0, clause 2.1.3, suggested that:

"*There **may** be 'explicit additional verification criteria' on top of what a requirement already says, … such as 'Identification of a verification method or verification step (e.g., software test, system test) is necessary, ... special test methods, environments, …"*

The word "may" makes it clear that this is optional for requirements processes, namely, not mandatory. Absence of such information therefore cannot be used for downrating.

Furthermore, the rules for SWE.6.BP1 and SYS.5.BP1 in the VDA Automotive SPICE® Guidelines v1.0 expected the same information as quoted above in italics. Consequently, downrating this would mean "double punishment" for both the requirements and the testing process, which is not considered compliant with ISO/IEC 33004's notion of disjoint processes in a PRM.

Furthermore, "preconditions," "verification methods," and "verification environment" are testing or verification concerns, respectively, but not requirements concepts ("Separation of Concerns" principle) which are now correctly, and exclusively, addressed in SYS.4, SYS.5, SWE.4, SWE.5, SWE.6, HWE.3, and HWE.4.

4. Verifiability is only one out of many state-of-the-art requirements characteristics. Others are according to ISO/IEC IEEE 24765 *[ISO24765]*, ISO IEEE 29148 *[ISO29148]*, ISO 26262, INCOSE Guide for Writing Requirements *[INCOSE]*, IREB CPRE *[IREB CPRE]*, e.g.,

   - design-free/implementation-free
   - unambiguous/comprehensible
   - consistent, not contradicting any other requirement
   - complete
   - no redundancy across requirements
   - atomic/singular

   To resolve all these misinterpretations, the new BP1 in SYS.2, SWE.1, and HWE.1 was introduced, which integrates the notion of verification criteria.

Note that the decision of requiring characteristics for requirements at capability level 1 is not in conflict, or semantically overlapping, with GP 2.2.1. As pointed out in [Metz2016], GP 2.2.1 of SYS.2 may address different quality criteria such as structural requirements (e.g., by means of templates) or checklists.

2.3.3.2 "Functional requirement" and "non-functional requirement"

There are no clear internationally agreed definition of the terms "functional requirement" and "non-functional requirement" (see discussion of references below). However, Automotive SPICE® still uses the two terms "functional requirement" and "non-functional requirement" in requirements-oriented processes to

- make practitioners not forget about the importance of equally reflecting on 'non-functional' characteristics
- enable assessors to downrate the absence of such information in requirements.

ISO/IEC IEEE 29148 *[ISO29148]* defines in clause 5.2.8.3:

- *"Functional/Performance… describe the system or system element functions or tasks to be performed by the system …"*
- *"Quality (Non-Functional) Requirements – Include a number of the 'ilities' in requirements to include, for example, transportability, survivability, flexibility, portability, reusability, reliability, maintainability and security."*

The **IREB CPRE** *[IREB CPRE]* says that:

- *"Non-functional requirements" is an umbrella term and, thus, represents "quality requirements" or "constraints".*
- *Quality requirements are said to be, for instance, performance, reliability, usability, portability.*

In **ISO/IEC IEEE 24765** *[ISO24765]* the following can be found:

- There are two definitions for "functional requirement:"

1. *"A statement that identifies what a product or process must accomplish to produce required behavior and/or results."*
2. *"A requirement that specifies a function that a system or system component must be able to perform."*

- The definition of 'non-functional requirement' is

  "a <software> requirement that describes not what the <software> will do but how the <software> will do it."

- Non-functional requirements are further claimed to be synonymous to "design constraints."

The systems engineering **INCOSE Guide for Writing Requirements** *[INCOSE]* informs:

- *"Types of requirement – Requirements that address capability and function may be expressed in a different manner to constraints and requirements specifying other system properties (often confusingly called 'non-functional' requirements – a term that will not be used again in this guide). The guide is intended to cover the whole range of requirement types."*

2.3.3.3 "Functional" and "non-functional" do not serve as requirement types

Base practice 2 of SYS.2, SWE.1, and HWE.2 require the structuring of requirements:

BP2: Structure system/software/hardware requirements. Structure and prioritize the system requirements.

NOTE 3: Examples for structuring criteria can be grouping (e.g., by functionality) or product variants identification.

NOTE 4: Prioritization can be done according to project or stakeholder needs (via, e.g., definition of release scopes). Refer to SPL.2.

In this context, the notions “functional” and “non-functional” are no relevant classification or categorization criteria for requirements.

Reasons:

- A particular requirement may, and on most cases will, contain both functional and non-functional information, and would therefore fall into both categories. See subchapter 2.3.3.1 for examples.
- Differentiating would not have any implication on how requirements are further processed, i.e. there is no difference in needs for traceability, verification/validation, etc.

2.3.4 Consistency and traceability

In the Automotive SPICE® consistency and traceability are addressed by a BP in the engineering processes and the change request management process. Furthermore, consistency is addressed in the project management process.

2.3.4.1 Purpose of consistency and traceability

Traceability alone (e.g., the existence of links) does not necessarily mean that the information is consistent with each other.

The information item 13-51 “Consistency Evidence” is explained as:

- Demonstrating bidirectional traceability between artifacts or information in artifacts, throughout all phases of the lifecycle, by, e.g.:
  - tool links
  - hyperlinks
  - editorial references
  - naming conventions (i.e., “linked-by-name”)
- Evidence that the content of the referenced or mapped information coheres semantically along the traceability chain, e.g., by:
  - performing pair working or group work
  - maintaining revision histories in documents indicating that corrections were made in regard to other documents’ content

- providing change commenting (via, e.g., meta-information) of database or repository entries indicating that corrections were made with respect to other documents' content

Experience has shown that it appeared unclear how to ensure consistency without being able to trace the two respective pieces of information (in whatever form). Therefore, these two BPs have been reintegrated into one, which does not invalidate the additional advantages of traceability in subchapter 2.3.4.2.

The following figure shows the relationships mentioned in the BPs respectively for traceability and consistency:

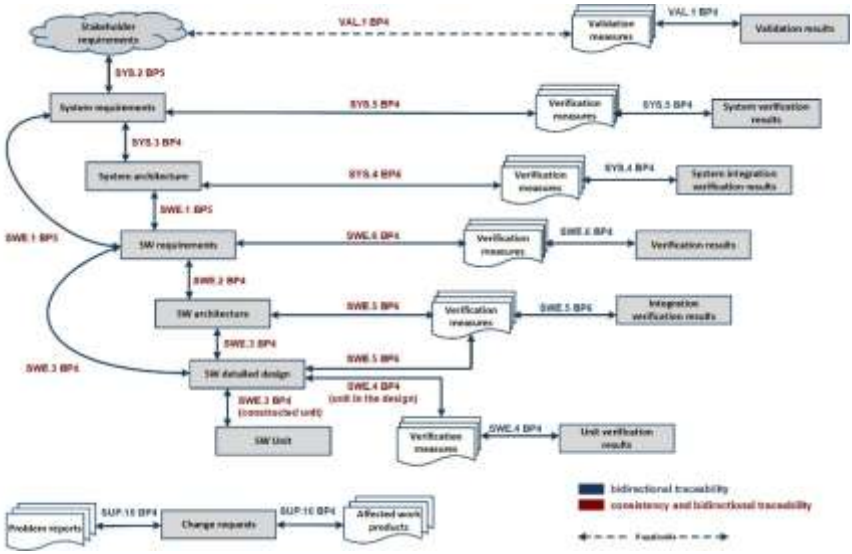


Figure 2-4: Consistency and traceability between system and software work products

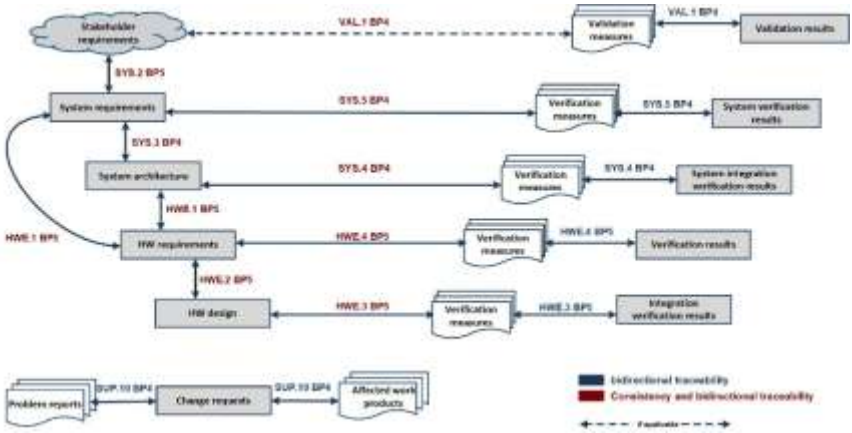


Figure 2-5: Consistency and traceability between system and hardware work products

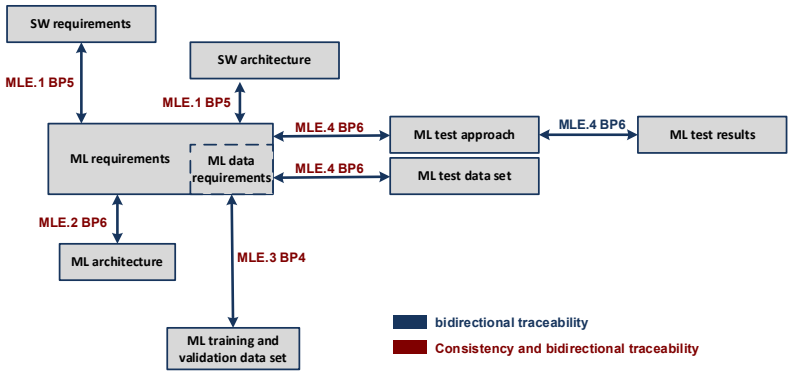


Figure 2-6: Consistency and traceability between ML work products

2.3.4.2 Other advantages of traceability

Traceability between or within work products of the same process is not addressed by the BPs at capability level 1. Instead, it may be considered in the context of GP 2.2.2 at capability level 2.

In addition, bidirectional traceability further supports:

- analysis of dependencies in both directions

- determination of requirements coverage
- determination of verification coverage
- status tracking of implementation of requirements and verification measures
- impact analysis and risk assessment of change requests on affected work products
- impact analysis and risk assessment for changing technology
- impact analysis on cost, schedule, effort, and technical impact

Rating Rules:

None.

2.3.4.3 Granularity of traceability

The following list defines allowed levels of traceability granularity:

- Requirements
  - single requirement
  - cluster of requirements
- Architecture
  - single software component
  - cluster of software components
- Software detailed design
  - single software unit
  - cluster of software units[1]
- Hardware design
  - single HW part
  - single HW component (i.e., a functionally coherent cluster of HW parts)
  - cluster of HW components
- Verification/validation measures
  - single verification/validation measure
  - a cluster of verification/validation measures
- Verification/validation results
  - single verification/validation result

- cluster of verification/validation results

- Single change request
- Single problem record

[1)] More detailed explanation for traceability of clusters of SW design elements see subchapter on SWE.3, “Traceability and Consistency.”

Rating rules:

[TAC.RL.1] If traceability is distinctly established between reasonable clusters of information instead of individual atomic elements, then the “Consistency and Traceability” BP shall not be downrated.

2.3.4.4 Methodology/approach for traceability

A PAM does not predefine any methodology/approach or tools. The same applies for achieving traceability. The selected methodology/approach for traceability however need to be appropriate for handling the given complexity, such as by tool support.

Rating rules:

[TAC.RL.2] If an automated tool-based approach for traceability is not used in favor of a manual approach with snapshot-based checks, then the “consistency and traceability” BP shall not be downrated.

2.3.4.5 Evidence for consistency

The Automotive SPICE® PAM requires *ensuring* consistency but not *reviewing* or *documenting* it. This means that the exact way this is done cannot be predefined. See also the information item 13-51 “Consistency evidence.”

Rating rules:

[TAC.RL.3] If there is no explicitly documented review record or analysis record providing evidence of consistency between related

information in favor of approaches such as performing pair working or group work, peer spot checks, explaining entries in revision histories in documents, or providing change commenting (via, e.g., meta-information) of database or repository entries, then the "Consistency and Traceability" BP shall not be downrated.

2.3.5 "Communicate" base practice

At capability level 1 it is required that agreement and communication is effective. A PAM cannot predefine a particular form. Therefore, the information item 13-52 "Communication evidence" is explained as:

- All forms of interpersonal communication, such as:
  - emails, also automatically generated ones
  - tool-supported workflows
  - podcasts
  - blogs
  - videos
  - forums
  - live chats
  - wikis
  - meetings, orally or via meeting minutes (e.g., daily standups)

Also, following both a push and pull principle can be acceptable.

To evaluate if communication is effective, the assessor must consider the context of the project and the size and structure of the organization.

Rating rules:

[COM.RL.1] If effective communication of agreed information at Capability Level 1 is not done based on baselines or by explicitly documented communication or review records, then BP "Communicate" shall not be downrated.

Moreover, note that there is no full semantical overlapping with GP 2.1.6 at capability level 2.

2.3.6 Verification process-oriented concepts

2.3.6.1 “Verification” instead of “testing”

The respective SYS, SWE, and HWE processes have been enhanced to address verification (being an umbrella term) instead of testing only.

Reasons:

- Especially at the system and hardware levels, testing is not the only verification approach. Rather, measurements (e.g., geometrical tolerances), calculations or analyses (e.g., strength/stress calculation using a finite element method), or simulations instead of using physical samples are other methods of verification. The same is true for mechanical or hardware development. Therefore, the umbrella term verification now forms the center of those processes’ purposes.

- The process SWE.4 “Unit Verification” was already an exception as a software unit can be verified coherently by means of a combination of static analysis, testing, and code reviews (a view that is also inherent in ISO 26262-6:2018 clause 9).

2.3.6.2 No more use of term “item” in verification processes

Since Automotive SPICE® 4.0, use of the term “item” has been eliminated.

The term “item” referring to an object-under-test conflicted with other standards like ISO 26262 “Functional safety for Road Vehicles” *[ISO26262]*. This automotive domain-specific Functional Safety standard refers to an “item” rather as

- a term representing a technical product or a distributed functionality from a logical-functional perspective, irrespective of how many systems will help implementing it (e.g., a new vehicle function such as adaptive cruise control, or a mechatronic vehicle-level system, such as an automatic side door access system);

- the “thing” on which HARA is performed, in order to

  - remove conflict with other standards,
  - bridge the language of different standards, and
  - enable a better alignment of Automotive SPICE® assessments and other types of assessments (e.g., ISO 26262 *[ISO26262]* safety audits) in practice.

2.3.7 No explicit notion of “specification” and “strategy” at level 1

Today, requirements or verification/validation measures are not necessarily contained in a physical single document but objects or entries in, for instance,

- a database, or
- repositories like application lifecycle management or product lifecycle management tools.

These entries are usually allocated to releases and product variants, which is meta-information expressed via, for example, attributes. Furthermore, requirements and verification measures for a particular product may come from various sources, like standard product kits or platform documentation and new features for customers. Additionally, selective baselining is possible for sets of entries in such repositories.

In this PAM this is emphasized by no longer talking about “specifications” in the respective processes but about “requirements” or “verification measures,” etc. This is further in line with ISO/IEC 330xx‘s new “information items” notion instead of “work product” indicators.

In addition, this will prevent the assessor from downrating if such information is not represented in one physical document.

Similarly, the former strategy BPs at capability level 1 have been removed in favor of reallocating their content to other existing BPs. Also, the former process-specific work product indicators 08-xx with

their work product characteristics have been removed in favor of reallocating their content to newly defined information items.

Reasons:

1. The extra strategy BP and plan work product indicators could be misinterpreted in a way that an explicitly written document would be required. In practice, this has resulted in downrating BP1 if such an explicit document is not available. In some contexts this led to “over-engineered” processes.
2. In a context where an explicitly written strategy document is necessary, the existence of a “strategy” BP and the “plan” work product indicator, respectively, could be misinterpreted by requiring exactly one single document, and/or following the same structure as given in the work product characteristics.
3. That “strategy” BP is the “plan” work product indicator could be misinterpreted by requiring a more systematic and controlled approach at capability level 1 already. This makes the defined semantical distinction between – and the message behind – capability levels 1 and 2 become elusive.

Consequently, assessment results on the same or on very similar contexts sometimes differed very significantly.

2.3.8 Terminology – “Affected Party” (Level 1) vs. “Involved Party” (Level 2)

Processes at capability level 1 use the term “affected party” in the context of BP “Communicate.” This is to indicate that for every process instance A there is a downstream process instance B that requires the technical (i.e. capability level 1) output of A as a necessary input. Otherwise, process instance B would not be able to proceed, or update its output.

In contrast, “involved party” at capability level 2 includes, but goes beyond “affected parties”. For example, there may be a stakeholder who

- is passively kept in the information loop (e.g., a line manager, steering committee);
- is providing input (e.g., a deadline, a particular resource) and only requiring a commitment, but no further active involvement.

Affected parties thus are a subset of involved parties.

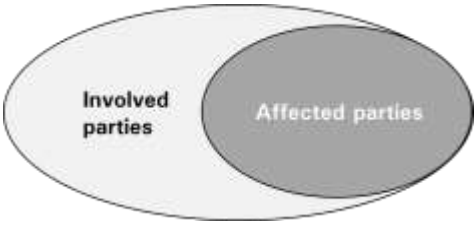


Figure 2-7: Affected parties vs. involved parties

2.3.9 No extra BP on evaluating alternative architectures

Since Automotive SPICE® 4.0, the base practice 'Evaluate alternative architectures' has been revised and integrated into SYS.3.BP3, SWE.2.BP3, and HWE.2.BP4. Documenting a rationale for each decision considered to be a key architectural decision of the chosen architecture is now required; Reason: it is considered of higher practical value to provide arguments why a given design was chosen rather than explaining which other particular approaches were not chosen. Furthermore, it can be considered that the former implies the latter.

2.3.10 Terminology – "Measure" vs. "Metric"

In the English version, the term "measure" can mean both

- "*to find the size, quantity, etc. of something in standard units, 'size/quantity'*" and "*… to judge the importance, value or effect of something*", respectively, and
- "*a plan of action or something done.*"

Since it was one of the objectives for Automotive SPICE® 4.0 to employ terminology more homogeneously, the decision was made to use the following terms and meaning consistently:

- Quantitative measurement – “metric
- “A plan of action” – “measure”
- “To act in an operational manner” – “action”

2.3.11 "Agree" and "Summarize and Communicate"

The information flow on the left side of the "V" is ensured through a base practice "Communicate agreed ‘work product x’." The term "agreed" here means that there is a joint understanding between affected parties of what is meant by the content of the work product.

The information flow on the right side of the "V" is ensured through a base practice "Summarize and communicate results." The term "Summarize" refers to abstracted information resulting from test executions made available to all affected parties.

These communication-oriented base practices do not require a planning-based approach, nor a formal approval, confirmation, or release, as this is targeted at by GP 2.1.6 on capability level 2. At capability level 1 the communication-oriented base practices mean that the work products (or their content) are to be disseminated to affected parties.

An overview of these aspects is shown in the following figure:

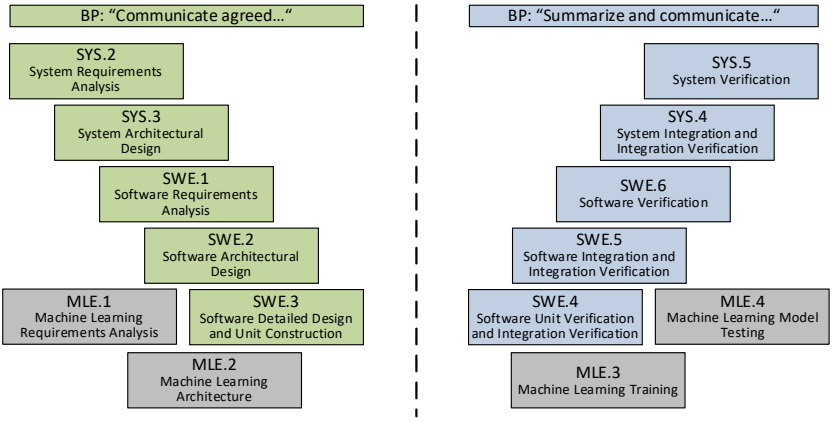


Figure 2-8: “Agree” vs. “Summarize and communicate”

2.3.12 "Element," "Component," and "Unit"

The following figure depicts the relationships between system elements, software components, and software units which are used consistently in the engineering processes. See the terminology in the Automotive SPICE® PRM/PAM *[AS41]* to learn about the definitions of these terms.

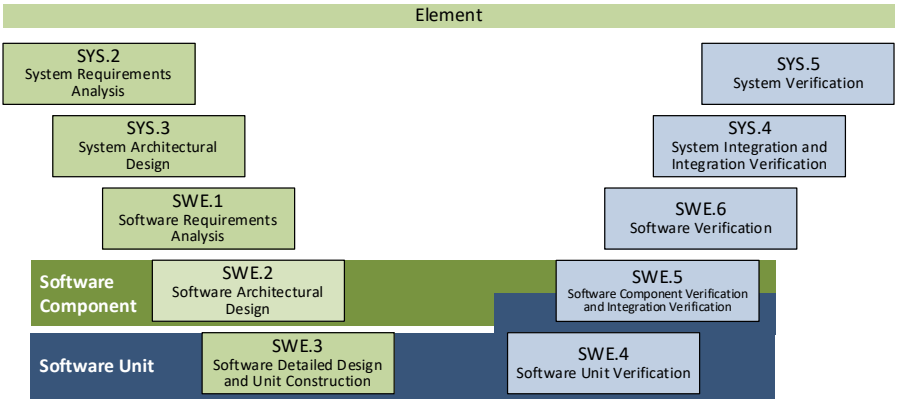


Figure 2-9: Element, component, and unit

2.4 Software unit behavior and unit integration, component behavior, and software component-level verification

Software unit integration

When a software component comprises many software units it may – depending on the context and nature of the software component – be necessary to perform *intra-component* unit integration verification first before the component itself is verified from a black-box perspective. It may have seemed obvious to add to SWE.4 three more BPs on the explicit specification, selection, and performing of software unit integration.

However, there are also contexts in which such stepwise software unit integration is not applicable or technically does not add value. In such contexts, rating such additional BPs as “F” is considered a falsification of the message behind the rating value “F”, which is: “There is objective evidence (found during the assessment) that there is an actual operational workflow without any significant risk.” Also, rating such BPs as ‘N’ would possibly, and unnecessarily, reduce the PA 1.1 rating. Furthermore, acc.to ISO/IEC 33020 *[ISO33020]* a rating of “not applicable” generally is not permitted. An alternative might have been to:

- Introduce two different integration processes, one for the unit and one for the component levels. However, this would have unnecessarily increased the number of processes and introduced replication of BPs (e.g., Select..., Communicate…, etc.), neither of which were goals for Automotive SPICE® 4.0.
- Combine SWE.2 and SWE.3. Still, this would not have made the left and right-side processes symmetrical as SWE.4 and SWE.5 would both trace to such a new combined process.
- combine SWE.2 and SWE.3, and to establish a new process such as “Unit Implementation” around the BP “develop software units”. This would have made the processes on the left and right-

hand side symmetrical. However, creating a full new process around one single BP “develop software units” appeared unnecessarily complex.

For these reasons, the decision for Automotive SPICE® 4.0 was to express both levels of integration, namely software unit integration and software component integration into the full software, within SWE.5. This was done by SWE.5.BP1 and SWE.5.BP4, respectively, talking about

- “software elements”, which is an umbrella term for software units and software components (see the Automotive SPICE® glossary), and
- “integrating the software elements hierarchically until the software is fully integrated.”

This should provide freedom for the assessed parties to define and explain which elements in their context are to be exactly integrated: software units alone, software components alone, or both.

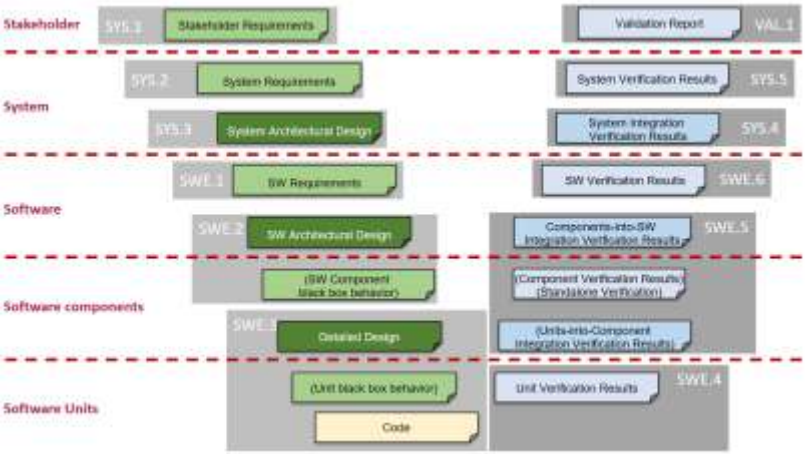
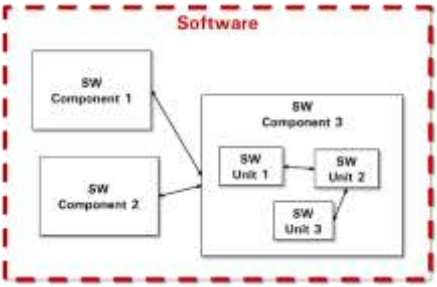
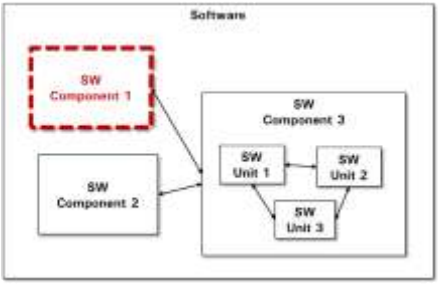
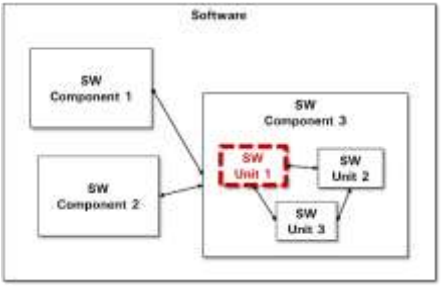
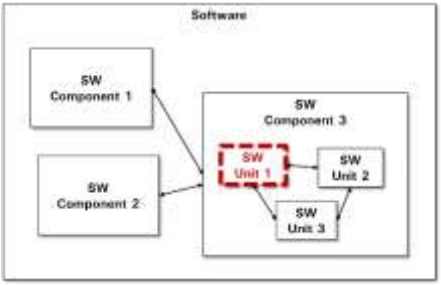


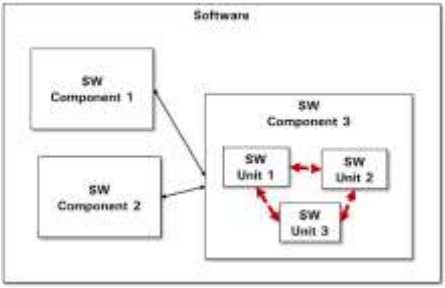
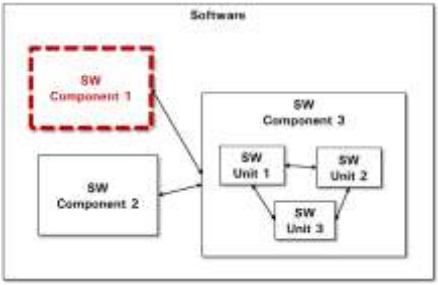
Figure 2-10: Software unit behavior and unit integration, component behavior, and software component-level verification

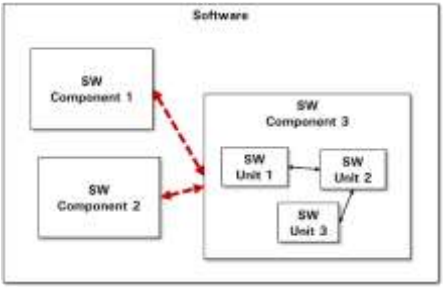
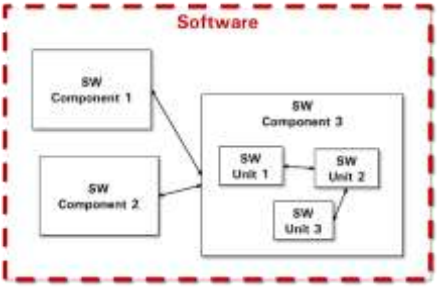
See also Table *2-1*

Table 2-1: SWE.x concepts since Automotive SPICE® version 4.0

Aspect	**As addressed in process**
Software requirements  	SWE.1
Definition of the behavior of a single software component (as opposed to interactions between components)  	SWE.2

Aspect	As addressed in process
Definition of the behavior of a single software unit 	SWE.3
Verification of a single software unit 	SWE.4

Aspect	As addressed in process
Integration, and integration verification, of software units into their component  	SWE.5
Verification of a single software component (prior to integration with other components)  	SWE.5

Aspect	As addressed in process
Integration and integration verification of software components  	SWE.5
Verification of the integrated software  	SWE.6

Software component standalone black-box verification

The “next step above” software unit integration is verifying a software component alone from a black-box perspective. However, this was not sufficiently expressed in Automotive SPICE® v3.1, see Table *2-1*.

For Automotive SPICE® 4.0 the decision was to embed this concept in SWE.5. It was not the decision to:

- Introduce a new process outside SWE.4, SWE.5, and SWE.6. Reason: this would also have unnecessarily increased the number of processes and introduced replication of BPs (e.g., Select..., Communicate… etc.) redundant, neither of which were goals.

- Add it to SWE.6. Reason: software component verification happens, from a lifecycle model perspective, after software unit integration but before the integration of all software components into the full software. Adding this concept to SWE.6 was considered to be less intuitive compared to the given solution of embedding it to SWE.5. Also, it would have made necessary traceability between SWE.6 and SWE.2 which, again, was not considered intuitive.

As a trade-off between avoiding a mass of processes and BPs and maintaining best possible intuitiveness, the SWE.5 process since Automotive SPICE® 4.0 addresses all integration levels (see Figure 2-10). It combines the logical flow of integration of software units into their joint software component, software component standalone verification and integration of all software components into the full software.

2.5 Application in specific environments

2.5.1 Model-based development

The approach of model-based development can be used for different purposes within the system and software development. For example, models can support the requirements elicitation process or the development of complex algorithms.

Refer to subchapter 2.3.4 for the generic concept of consistency and traceability.

2.5.1.1 Models need additional descriptions

Models can be used in different use cases within the development process (e.g., for requirements elicitation, architectural design, detailed design, code generation, verification); the use case of the

model must be defined and documented, such as “the system architecture is documented using SysML.”

Modeling notations may be graphical, textual, or a mixture of both and may differ depending on the use case for the model. The syntax and semantics of the notations shall be defined in a formal, semi-formal, or informal way.

Aspects (e.g., design decisions) that the modeling notations cannot express require additional descriptions in natural language (e.g., via text annotations). The corresponding information item characteristics (see Annex B in Automotive SPICE® PAM) give guidance for the aspects of the additional descriptions.

The following rating rules must be interpreted in the respective context, process, and use case (e.g., if the model is used for software requirement elicitation, the corresponding indicator is SWE.1.BP1, if the model is used for software detailed design, the corresponding indicators are SWE.3.BP1, SWE.3.BP2, SWE.3.BP3).

Rating rules:

> **[MBD.RL.1]** If the syntax and semantics of the model notation are not defined or not appropriate for the use case, the corresponding indicator shall be downrated.
>
> **[MBD.RL.2]** If the additional description is missing or insufficient, the corresponding indicator shall be downrated.
>
> **[MBD.RL.3]** If the additional description is documented in extra documents but associated with the model, the corresponding indicator shall not be downrated.

2.5.1.2 Consistency of additional descriptions

Aspects that cannot be expressed by the modeling notation might be missing, if not documented in some other appropriate form.

If the model itself is part of a development artifact, (e.g., for the use case of requirement elicitation the model is part of the requirement specification), it shall be ensured that this additional description in

natural language of the model is considered in the following development process.

Rating rules:

> **[MBD.RL.4]** If the additional description for the model is not considered in downstream processes, the corresponding indicator shall be downrated.

2.5.1.3 Models for code generation

If automated code generation is used (a.k.a. graphical programming), then the basis for the code generation is

- inherent in the design or
- derived from the design (in which case traceability between model and design has to be established).

Commonly, in the software design there is information that is not usable for code generation but is important to convey an understanding of the software. An example is textual annotations to graphical elements.

Unit verification performed at the model level shall provide evidence for consistency of the software units with the software detailed design and with the software requirements.

Traceability and consistency support the compliance of a model and code part. The consistency of additional descriptions with the model and/or with the auto-generated code must be established, such as by reviews.

Rating rules:

> **[MBD.RL.5]** If there is no or insufficient evidence for compliance of the auto-generated code with the detailed design, then SWE.3.BP4 shall not be rated higher than P.
>
> > *NOTE: this will include consistency with the non-functional software requirements by means of consistency and traceability between the detailed design and the software requirements.*
>
> **[MBD.RL.6]** If there is no static verification and unit testing performed on code automatically generated from the model by

a qualified tool chain (and without any modification after generation), then SWE.4.BP3 shall not be downrated.

NOTE: Qualified tool chain for the code generation means that there is evidence that the generated code is correct and consistent with the model.

[MBD.RL.7] If there is no static verification and unit testing performed on code automatically generated from the model by a qualified tool chain (but has been modified after the generation), then SWE.4.BP3 shall be downrated.

2.5.2 Agile environments

Agile software development is based on principles of the Agile Manifesto with the objective to create lightweight, adaptive development methods. Popular frameworks for agile software development are SCRUM, KANBAN, eXtreme Programming, and SAFe.

Automotive SPICE® describes meaningful process principles but does not predefine any concrete lifecycle model, method, tool, templates, metrics, proceedings etc. (the WHAT level). This means the Automotive SPICE® content resides at a higher level of abstraction than any process implementation (the HOW level) to allow for maximum freedom, and, also, for benchmarking. In contrast, agile methods rather reside at the HOW level. Therefore, Automotive SPICE® and agile approaches cannot, by definition, contradict each other. The only valid question would be to ask whether concrete process implementations, following or including agile methods or not, actually satisfy the Automotive SPICE® principles. Automotive SPICE® does not predefine any type of lifecycle model like V- or Waterfall-model. For details see *[IntAgile]*.

Agile methods may support Automotive SPICE® requirements and should be compliant with required rules and standards. For example, non-functional requirements, review and documentation criteria or coding guidelines are valid in an agile and non-agile lifecycle.

The rating rules in this chapter are based on practical experience and have no pretention of completeness.

The documented practical experience within this chapter are partly not specific to agile development (e.g., missing software architecture) but have been detected often in Automotive SPICE® Assessments of projects with agile development methods.

2.5.2.1 Planning in agile environment

Customer planning requirements are equal in agile and non-agile development. Projects have to ensure that the technical content of features is delivered and bugs are fixed as agreed and scheduled. The planning methods may differ.

Therefore, the agile project has to ensure that the project planning is in line with the customer release planning.

For example, an agile SCRUM project will ensure that the sequence of sprint cycles will deliver the needed functionality corresponding to the customer requirements. Namely, the planning must ensure that the agreed features are developed and tested within the sprints before the planned release, and the planning has to be consistent across affected parties and agreed plans.

Rating rules:

[AGE.RL.1] If evidences from project planning (e.g., backlog, burn down chart and/or sprint planning) show gaps regarding the release planning and this aspect is significant, then the indicators MAN.3.BP4, MAN.3.BP9, and SPL.2.BP1 shall be downrated.

Additionally, the remaining effort for function development until future deliveries and start of production shall be estimated and covered by available capacity to ensure that additional effort caused by underestimated tasks (e.g., user stories) is not summing up and impacts future project milestones.

[AGE.RL.2] If remaining effort for features, which are to be delivered in the current or next release, is not estimated, then MAN.3.BP5 shall be downrated.

2.5.2.2 Project lifecycle

The chosen project lifecycle should fit to the project scope, requirements, deliveries, complexity, etc. Therefore, it may be

necessary to create a lifecycle according to a standard process with tailoring to meet the project needs.

For example, the customer might continuously deliver requirements to the project and expect continuous integration by the project in order to monitor the progress of the product. An agile development process (e.g., SCRUM or Kanban) may support the customer requirements regarding progress monitoring and incremental requirements delivery.

2.5.2.3 Management of requirements

In practice, some projects manage the requirements in a change or tracking tool in which the requirements are managed within tasks or change requests only. These solutions may have the benefit of tracing requirements to tasks and make coding easier but have the disadvantage that no overview of all requirements exists. This impacts the requirements review effectiveness regarding cross-consistency. Furthermore, without an overview of requirements, the maintenance of requirements is very difficult regarding impact analysis of changes and confirming all requirements are implemented completely.

For example, a feature has different functions. In development, a first task is issued for development of the feature. During the development period, different change requests/tasks are assigned to the feature and implemented to add, change or delete functions of the feature. At project end the requirements of the feature can only be determined by assessing all tasks of the feature.

2.5.2.4 Risk management

Customers, company or project requirements often demands integrating risk management for the development projects, and this risk management needs to be integrated into the agile project.

For instance, risk management could be integrated into an agile project by incorporating risk assessments into sprint planning meetings and maintaining a risk register as part of the project backlog.

2.5.2.5 Architecture

An architecture must be defined that identifies the components to be traced to the related requirements.

Agile projects have to ensure that an architecture is developed and maintained and that traceability between architecture and requirements, architecture and detailed design, and architecture and integration verification is established.

An example of a proceeding for architectural evolution within an agile environment can be that basic architecture and architecture rules are defined at project start and this architecture then being incrementally developed further within sprints (for SCRUM based projects). For all architectural modifications an impact analysis is performed.

Rating rules:

[AGE.RL.3] If the system architecture is modified incrementally including impact analysis, then SYS.3.BP1 shall not be downrated.

[AGE.RL.4] If the software architecture is modified incrementally including impact analysis, then SWE.2.BP1 shall not be downrated.

2.5.2.6 Verification

Verification of system and software artifacts needs to be established in development projects. Agile methods may combine verification levels. The agile project has then to ensure that the process purposes of all relevant verification processes are fulfilled by the defined activities. In such cases the related processes cannot be downrated.

2.5.2.7 Independent quality assurance

Agile development methodologies may define generic role descriptions which need to be derived for the roles and responsibilities in the development project. By defining the responsibilities, the project has to ensure that work product and process quality assurance are performed at project level independently and performed objectively without conflicts of interest.

2.5.2.8 Pair programming

An agile practice denoting one programmer writing code while (at least one) another one is reviewing each line of code and the other developer is typing it. Developers may switch those roles.

Rating rules:

[AGE.RL.5] If the way pair programming is technically performed is not in conflict with code review requirements in terms of formal reviews, inspections, walkthroughs, then SUP.1.BP3 and SWE.4.BP3 shall not be downrated.

2.5.3 Development external to the assessed project (DEX)

2.5.3.1 General information

Automotive software-based systems are typically developed as a complex collaboration of system, hardware and software developers working in different organizations, and in development sites that can be distributed across different countries. These organizations include the assessed organization (project) and external organizations.

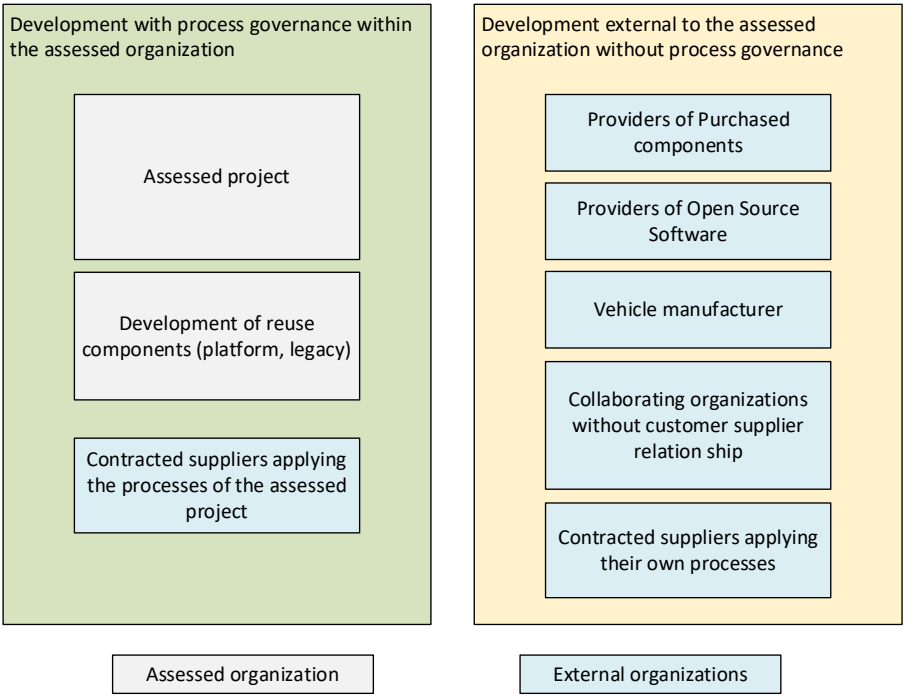


Figure 2-11: Collaborating entities

External organizations contribute to such product development based on contracts or commitments. Their activities are typically not performed under supervision of the assessed organization.

External organizations include:

- The vehicle manufacturer and its subsidiaries

- Contracted suppliers
- Contracted sub-suppliers
- Contracted collaborating organizations not in a customer-supplier relationship
- Third-party organizations

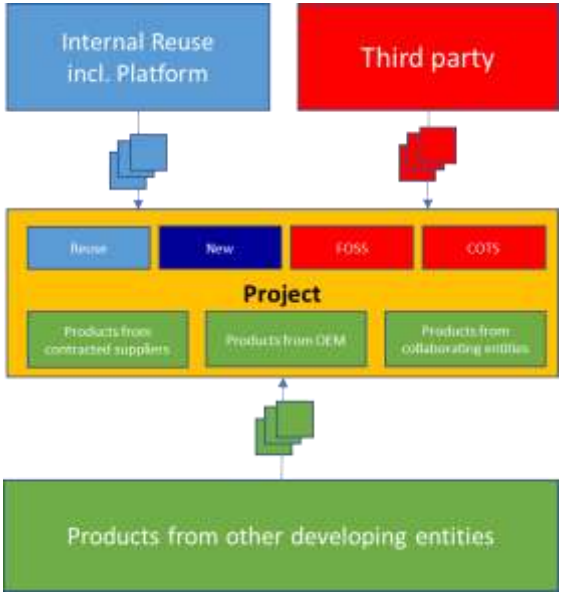


Figure 2-12: Interdependencies of a project integrating products from other parties

In an Automotive SPICE® assessment, the processes and practices shown in Table 2-2 apply in order to evaluate the processing of products and software developed externally to the assessed project. This includes products and software developed externally to the assessed project from, for example, collaborating entities, third-parties, and reuse.

Table 2-2: Applicable processes for products that are developed external to the assessed project

Business case	**Support from originator**	**To be evaluated in assessed project (if included in assessment scope)**						
		MAN.3 BP.7	**SUP.9 SUP.1**	**SYS.2 SYS.3 SYS.4 SYS.5**	**SWE.1 SWE.2 SWE.5 SWE.6**	**HWE.1 HWE.2 HWE.3 HWE.4**	**REU.2**	**ACQ.4**
Free and open-source software			X		X			
Purchased products (COTS)	X	X		X	X	X		
Reused products	X	X	X	X	X	X	X	
Products from vehicle manufacturer or its subsidiaries	X	X		X	X	X		
Products from contracted development supplier or sub-supplier	X	X		X	X	X		X
Products from (contracted) collaborating organizations that are not in a customer supplier relationship	X	X		X	X	X		

The engineering activities of automotive software-based systems within an organization are not necessarily performed at one location. In the context of a project for the development of a particular product the necessary engineering resources, supporting resources and management resources may be distributed across separate departments, locations, buildings, third-party service providers, etc.

In the planning phase of an assessment the sponsor and the lead assessor must determine whether locations and departments within an organization will be covered with one or separate assessments.

If all locations and departments are performing their work based on a common standard process, it may be optimal to include them all in the assessment scope. If one location is solely responsible, for instance, for software testing, the interviews for this process shall be performed with only that location.

When locations or departments have different processes, separate assessments could be performed, or a single assessment may be organized with defined process instances for the processes performed with the same purpose and outcomes (e.g., project management, quality assurance, configuration management).

2.5.3.2 Maintain effective collaboration

Responsible roles within the project will maintain effective collaboration and communication including the definition of a consistent set of responsibilities to achieve the project goals.

Depending on the assessment scope the following aspects must be evaluated for the interfaces regarding development activities that are performed and results that are provided externally to assessed project:

- scope of work for all collaborating entities
- definition of responsibilities
- interfaces between overall plans, sub-project plans and plans for support organizations
- monitoring of agreed commitments
- communication between all entities
- compatibility of status models for work products
- providing necessary work products to collaborating entities
- preconditions to integrate work products from collaborating entities
- escalation mechanisms when work product requirements are not met

- verification and validation measures for the integration of system, hardware or software elements developed by different collaborating entities

Based on vehicle manufacturer strategies, the vehicle manufacturer may deliver source or object code to the supplier's software project. This means that the customer is part of the distributed development.

Rating rules:

[DEX.RL.1] If the scope of work is not defined for all collaborating entities, then MAN.3.BP1 shall not be rated higher than L.

[DEX.RL.2] If the planning of the overall project and the collaborating entities show inconsistencies and this aspect is significant in the context of MAN.3.BP9, then MAN.3.BP9 shall be downrated.

[DEX.RL.3] If the monitoring of the overall project does not detect deviations in fulfillment of agreed commitments from the collaborating entities and this aspect is significant in the context of MAN.3.BP7, then MAN.3.BP7 shall be downrated.

[DEX.RL.4] If the information about the properties used for the exchange of configuration items appears to be incompatible, then SUP.8.BP2 shall be downrated.

[DEX.RL.5] If preconditions for work products from collaborating entities to be integrated are not fulfilled and no appropriate reaction was started to resolve the issue, then SYS.4.BP3, HWE.3.BP2 or SWE.5.BP4 shall be downrated.

[DEX.RL.6] If the supplier project does not comply with the agreements and the agreed rules for the products or software supplied by the customer and this aspect is significant in the context of MAN.3.BP7, then MAN.3.BP7 shall be downrated.

[DEX.RL.7] If the customer of the assessed organization does not comply with the agreements and the agreed rules for the products or software supplied by the customer, then MAN.3.BP7 shall not be downrated but the noncompliance of the customer shall be documented in the assessment report.

[DEX.RL.8] If escalation mechanisms across the sub-projects are not defined and this aspect is significant in the context of MAN.3.BP7 or SUP.1.BP7, then MAN.3.BP7 or SUP.1.BP7, respectively, shall be downrated.

2.5.3.3 Acceptance of software from collaborating entities

Evidence is needed that products or software from collaborating entities were verified according to pass/fail criteria defined in validation measures of the acquirer. These acceptance criteria may contain for example the review of the release documentation, fulfillment of coding guidelines and/or code coverage of manual and automated tests in compliance with the agreed requirements.

For software without any support from a third-party provider (e.g., Free and Open-Source Software [FOSS]), the project must define acceptance criteria based on their integration and test concepts.

Rating rules:

[DEX.RL.9] If the verification measures for system or software integration, respectively, do not include the verification of elements developed by different collaborating entities and this aspect is significant in the context of SWE.5.BP2 or SYS.4.BP1, then SWE.5.BP2 or SYS.4.BP1, respectively, shall be downrated.

[DEX.RL.10] If no pass/fail criteria are defined to check the compliance of third-party products or software and this aspect is significant in the context of SWE.5.BP1 or SWE.5.BP2, then SWE.5.BP4 shall be downrated.

[DEX.RL.11] If third party products or software is not checked for compliance with defined criteria that have to be fulfilled and this aspect is significant in the context of SWE.5.BP5, then SWE.5.BP5 shall be downrated.

2.5.3.4 Functional and non-functional requirements

The specification or the contractual basis of third-party products or software must cover functional and non-functional requirements.

The requirements of the third-party products or software will be in line with requirements of the project. In case of products or software

developed by a supplier based on project requirements the project has to transfer these requirements to the supplier and should use the associated tests as acceptance tests.

For “commercial-off-the-shelf products or software” the project has to ensure that the commercial-off-the-shelf products or software complies with the requirements specified for the purchased products or software. The specified requirements should form the basis for acceptance testing of this kind of third-party products or software.

The non-functional requirements may include, for example, quality requirements (such as specific coding guidelines or code metric targets) which are often used to support the verification process.

In case the third-party software comes without any support (e.g., FOSS) then the project must ascertain that non-functional requirements are met, or that the third-party software (e.g., non-automotive commercial-off-the-shelf software) is treated according to legacy software rules in the process on the management of products for reuse (see subchapter 3.32).

In the context of this guideline, “legacy software” was / has been developed in a previously finished project (previous with regard to the project in the assessment scope) and was produced at least once. In an assessment, the development process used when developing the legacy software maybe unknown or may differ from the process used in the assessed project.

Rating rules:

[DEX.RL.12] If the properties of the products or software from collaborating entities are not in line with the requirements for the project and this aspect is significant in the context of the consistency and traceability indicators of SYS.3, SWE.2 or HWE.2, then these indicators shall be downrated.

[DEX.RL.13] If legacy software is used in the assessed project and there is no evidence for measures proving that the legacy software fits to the project requirements, and this aspect is significant in the context of SWE.1.BP1, then SWE.1.BP1 shall be downrated.

2.5.3.5 Software architecture

The software from collaborating entities and its interfaces (e.g., external API) must be considered, and put in context, in the software architecture.

For example, a purchased operating system together with its interfaces, and how the operating system is connected to the relevant software architecture elements, has to be mentioned in the software architecture.

Rating rules:

[DEX.RL.14] If the interfaces of software from collaborating entities are not part of the software architecture and this aspect is significant in the context of SWE.2.BP1, then SWE.2.BP1 shall be downrated.

[DEX.RL.15] If dynamic aspects of software from collaborating entities are not reflected in the software architecture and this aspect is significant in the context of SWE.2.BP2, then SWE.2.BP2 shall be downrated.

[DEX.RL.16] If the external interfaces of third-party software are not reflected in the software architecture and this aspect is significant in the context of SWE.2.BP1, then SWE.2.BP1 shall be downrated.

[DEX.RL.17] If reused components (e.g., platform and/or legacy software) used in the assessed project are not consistently reflected in the software architectural design and this aspect is significant in the context of SWE.2 BP1, then SWE2.BP1 shall be downrated.

2.5.3.6 Managing of Free and Open-Source Software (FOSS)

FOSS is source code that allows users to use and modify the software for any purpose. In any case, the open-source license agreement has to be fulfilled by the project. Otherwise, the project does not have the right to integrate and use the FOSS (e.g., open-source licenses shall be transferred to the customer, whereas some open-source licenses require disclosing the complete source code of the developed system). FOSS normally does not come with support, so the project must define and check rules whether those free software elements and the license fit to the project.

Note: FOSS is source code under an open-source software license agreement (e.g., GNU General Public License [GPL]*).*

Note: The rules for managing open-source software within a company are often called open-source policy.

Rating rules:

[DEX.RL.18] If FOSS is not managed according to rules ensuring that the free and open-source software license agreement is fulfilled and this aspect is significant in the context of MAN.3.BP3, then MAN.3.BP3 shall be downrated.

2.5.4 Application parameters

2.5.4.1 Interpretation of terms

In the following, the terms “calibration parameters” and “application parameters” are used synonymously.

Automotive SPICE® 4.0 defines “application parameters” as follows:

> *An application parameter is a software variable containing data that can be changed at the system or software levels; they influence the system’s or software behavior and properties. The notion of application parameter is expressed in two ways: the specification (including variable names, the domain value range, technical data types, default values, physical unit (if applicable), the corresponding memory maps, respectively); the actual quantitative data value it receives by means of data application. Application parameters are not requirements. They are a technical implementation solution for configurability-oriented requirements*.

Application parameters can therefore generally be used for two scenarios:

Influencing the implemented system behavior

The software makes the system behave according to the stored application parameter data not containing any executable or interpretable code, such as

- the range of the window glass in a door system within which anti-trap protection shall be active;
- values for low idle speed, motor characteristic diagrams, etc.;
- product vehicle impacting system behavior, e.g., country codes, left-hand/right-hand steering, etc.

Code selection

Code variants can be determined at compile-time by, for example, preprocessor commands or preprocessor variable settings of, for instance, the programming language C; as a result, the built program only contains to-be-executed code. In contrast, the expected executed code can also be determined later, namely, at

runtime, depending on application parameter values evaluated if-clauses.

In both scenarios, the actual data set can be flashed into the system by, for example, diagnosis jobs or end-of-line.

In this document, compile-time variants are not addressed.

2.5.4.2 Application parameters and requirements

In Automotive SPICE® the processes SYS.2, SWE.1, and HWE.1 do not explicitly mention application parameters.

Reason: The SYS.2 process is about documenting requirements, meaning expectations free from design & implementation decisions from a black-box perspective (see also subchapter 2.3.3). Therefore, SYS.2 will not know whether the system is actually going to have software in it. This is a decision made in the context of SYS.3 (see also Automotive SPICE® PAM subchapter 3.4).

What SYS.2, SWE.1, and HWE.1 can require however is "configurability" of a particular aspect.

Simplified example:

- Req #1: "The undervoltage boundary shall be configurable from 0[V] to 3.4[V]."
- Req #2: "When the system detects undervoltage then the system shall shut down in less or equal 500[ms] with a tolerance of +50[ms]."

In contrast, introducing application parameters (including the definition of the parameters' variable names, technical data types, default values, etc.) is a software design decision for implementing such configurability requirements. Furthermore, software application parameters are only one out of several possible solutions for implementing a configurability requirement. An alternative implementation solution in hardware for the same requirement would be, for example, e-Fuses.

Consequently, deciding on how many application parameters will be implemented in the software to express this and specific logical

information (i.e., the parameters' variable names, technical data types, default values etc.) are design decisions.

Rating rules:

[APA.RL.1] If the implemented application parameters and their values in the detailed design are inconsistent with configurability requirements, then SYS.3.BP1 and SYS.3.BP2 shall be downrated.

[APA.RL.2] If the detailed design or the implementation does not include checking for allowed value ranges of application parameters, then SWE.3.BP2 or SWE.3.BP3, respectively, shall be downrated.

2.5.4.3 Dependencies between parameters

Application parameters may have complex interdependencies: For instance, a particular parameter A may be exclusive to parameter B and C. Since application parameters are possible software solutions for configurability requirements, such interdependencies represent variants at the requirements level.

Examples:

- A navigational system for customer A additionally offers Points-Of-Interest while the variant for customer B does not
- A fault diagnosis for a stuck relay is not required for a semiconductor solution of a power stage (e.g., pulse-width based activation an actuator)

Depending on the complexity, the mastering of such variants at the requirements level can range from labeling requirements by, for example, meta-attributes in tools up to approaches as "feature trees."

2.5.4.4 Application parameters for code selection at runtime may represent product variants

Application parameters may represent product variants. Therefore, the verification parties should use a product sample that correctly represents the desired variant. Otherwise, verification might fail. This

further emphasizes why studying the requirements by the verification personnel is necessary.

Rating rules:

None.

2.5.4.5 Treating application parameter information as configuration items

For any application parameter impacting software behavior representing software decisions at runtime, then the

a) variable names,
b) the domain value range,
c) technical data types,
d) default values,
e) physical unit (if applicable), and
f) corresponding memory maps

are part of configuration items and subject to baselines.

Rating rules:

[APA.RL.3] If application parameters including all aspects above are not treated as configuration items, then SUP.8.BP1 shall be downrated.

2.5.4.6 Quality assurance on parameter information

Quality assurance activities must not only include evaluating whether data ranges, default values, and final values are correct, but must also check for consistency of this information across all parameters. Quality assurance must also evaluate whether the chosen data values represent the desired product variants. This is particularly important if different parties are responsible for different application parameters (see “Responsibility for Application Parameters” chapter).

Example 1:

The customer wants feature F_1 only. Therefore, it was decided to choose product variant V_2. However, erroneously both parameters X and Y were activated, resulting in the product actually realizing F_1 *and* F_2, i.e., variant V_1. This error should have been detected, for instance, by design or code reviews against the table.

	Variant V_1	Variant V_2	Variant V_3
Feature F_1, activated by parameter X	x	x	-
Feature F_2, activated by parameter Y	x	-	-

Example 2:

The customer wants features F_1 and F_2 only. Therefore, it was decided to choose variant V_1. Correspondingly, parameters X and Y were set. However, during requirements reviews, design reviews, and code reviews it remained unnoticed that parameter Y also activates feature F_3, which was never wanted.

	Variant V_1	Variant V_2	Variant V_3
Feature F1, activated by parameter X	x	x	-

Feature F2, activated by parameter Y	x	-	-
Feature F3, also activated by parameter Y	-	x	-

Rating rules:

[APA.RL.4] If application parameters do not receive quality assurance with respect to technical correctness, product variant consistency, then BP2 of SUP.1 shall be downrated.

2.5.4.7 Change management related to application parameters

Furthermore, in the context of change request management (SUP.10) the impact of a change on application parameter information must be explicitly analyzed. For application parameters covering code selection at runtime, this means activating or deactivating features, thereby changing product variants, while for application parameters influencing the implemented system's behavior this means changing the product application.

Rating rules:

None.

2.5.4.8 Application parameters and testing

Verification personnel should know about the configurability of undervoltage in the example in 2.5.4.2 as the verification measures shall be consistently traced to the requirements (SYS.2, SWE.1).

Secondly, to prove configurability, the verification personnel will need to be able to modify application parameters. This is ensured by SYS.5.BP1 aspects a) to e), requiring:

a) “techniques,” e.g., equivalence classes and boundary values for the undervoltage example

b) “entry criteria,” such as the availability of, e.g., extra parameter files to be provided by, e.g, the software department

c) “infrastructure/environment setup” whereas alternatively, the testing personnel may use a flash adapter or a calibration tool together with, e.g., an *.a2l file (representing a parameter-address mapping) to be able to modify the parameters themselves

The fact that, in practice, the verification personnel may of course be supported or advised by a requirements or software engineer here does not change the fact that the above information is a verification (SYS.4, SYS.5) rather than a requirement concern (SYS.2). Recall here that a PRM and PAM do not represent lifecycle models (see Automotive SPICE® 4.0 subchapter 3.3.4).

Rating rules:

[APA.RL.5] If samples that are used to perform verification measures do not reflect the correct application parameter settings, then BPs on “Verify…” or “perform verification” in SWE.4, SWE.5, SWE.6, SYS.4, or SYS.5, respectively, shall be downrated.

2.5.4.9 Responsibility for application parameters

Application parameters for code selection at runtime

The responsibility of such application parameters for code selection at runtime (see 2.5.4.4) is upon the supplier. Therefore, they must not be altered by the customer, so no application parameter information is exposed.

Parameters for influencing the implemented system behavior

Often the division of responsibility for application parameters does not follow the exact customer-supplier boundary.

Examples:

- A controller device supplier defines, and implements, all application parameters but the customer retains the right to alter some of them after the supplier's delivery.

- Owners of different reusable standard software components maintain their own local parameters.

Some of the parameters shall not even be accessible to the customer. In such a situation, like for product liability purposes, the responsibility for each of the application parameters should be explicitly defined. This may be done, for instance, by an addendum to a development agreement interface.

Rating Rules:

[APA.RL.6] If application parameter values can be or are altered at the product level by any other party than the product but responsibilities are not clearly defined, then MAN.3.BP7 shall be downrated.

2.5.5 AI assistance in development

The use of AI assistant tools to support process activities and creation or update of output work products gets increasingly used in development. Thus, it is important for assessors to have an understanding how this should be evaluated in the context of an Automotive SPICE® assessment.

ASPICE is a process assessment model that defines what a process shall consider, not how a process must be performed. Therefore, there is no limitation or recommendation regarding tool usage by Automotive SPICE®.

For example, SYS.2.BP2 expects to "structure and prioritize the system requirements." This does not demand any particular way of doing it, which exact structures are needed or tool should be used. ASPICE may provide examples in the Notes; however, these are not normative requirements but represent guidance. The use of an AI engineering tool to support the creation and update of work products is a how-level decision. The ASPICE process itself can still be assessed based on the what-level practices.

A potential assessment situation could be:

During the interview session on SWE.3 “Software Detailed Design and Unit Construction” the software developer explains that he uses an AI tool to create the source code. After the AI tool has generated the source code, the developer reviews the results and corrects them if necessary. By doing this, the developer ensures the source code matches the detailed design and fulfills the coding guidelines. Based on the developer´s explanation, the assessor will perform spot checks to evaluate if the process indicator SWE.3.BP3 "Develop software units" is fulfilled. If no deviations are found, there is no negative impact on the rating of process attribute PA1.1 of the SWE.3 process.

Why are the evaluation and correction of AI-generated content important?

Current AI tools sometimes create acceptable and accurate outputs. This may lead to the expectation that any AI output can be used as-

is. However, less accurate and sometimes even unusable results are produced just as often as good ones. To address these deficits, additional manual verification and measures are required.

Using AI engineering tools accelerates development but introduces additional risks due to their limitations. The assessed organization shall ensure with clear criteria that risks from AI engineering tools will be adequately handled. How this is done – whether through manual review, test driven development, or any other approach – is up to the specific process implementation.

In an ASPICE assessment, the assessor team is responsible for considering the use of AI engineering tools adequately during the evaluation and rating.

Using AI assistant tools may impact, among others, configuration management (SUP.8) and quality assurance (SUP.1) processes, as well as work product management (PA2.2).

Examples to be considered during assessment include:

- defining new configuration items, e.g., prompts used to generate source code
- maintaining the list of development tools, e.g., approval of the usage of specific AI tools
- version control of AI tools, models, and settings, because their behavior changes over time
- ensuring required input/output work products used for AI assistance are under quality control

2.5.6 Maintenance

This chapter concerns provisions of development projects for a later maintenance of their work results as well as development activities during maintenance of a software-based system. Use cases for these activities may be:

- the consecutive adaption of features for enhancement of customer convenience
- extending operational design domains
- platform development for multiple customer projects with individual adaptions
- replacing underlying hardware elements
- fixing errors or incomplete releases
- eliminating cybersecurity vulnerabilities
- taking over and adapting project outcomes in other development projects

The scope of the guidelines given here is, on one hand, for performing assessments in the development phase until the first start of production (SOP) to check whether the organization put measures in place to support a later carryover or maintenance of their work results. On the other hand, the guidelines can be applied as well for assessing a maintenance organization or projects with carryover of work results from organizations external to the assessed organization. Please refer also to chapters DEX (2.5.3) and REU.2 (3.32).

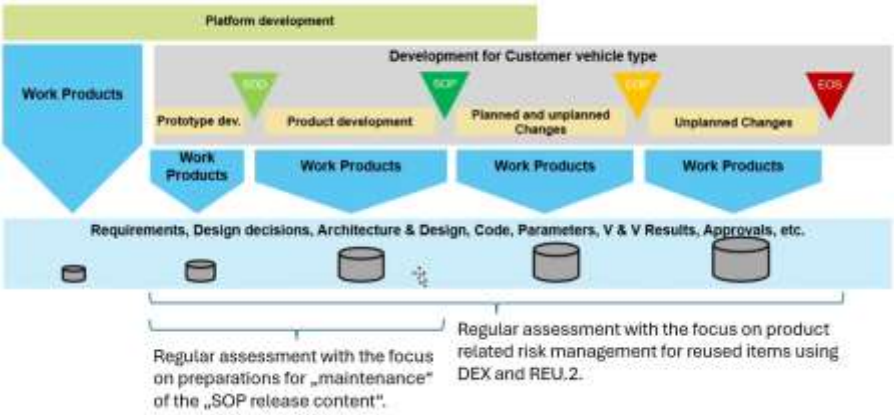


Figure 2-13 General overview of projects during their lifecycle

Many aspects relevant to the successful execution of maintenance are unfortunately not necessarily in scope of the project to be assessed. Processes concerning the overall supply chain, as well as specific contracting or legal clarifications are often performed by centralized departments or in the hands of customers. However, the active and running project to be assessed can still make important contributions. This chapter shall serve the assessors as a guide to use the existing framework and include additional indicators, for a successful preparation of a later maintenance phase, in order to determine potential risks and outline improvement potentials.

Conducting valid interview sessions with respect to maintenance (before SOP) is challenging because the execution of specific product maintenance activities like the creation of post-release patches may not be observed during the regular project execution. Additionally, the performers of the maintenance phase, including potentially involved parties at the customers and suppliers, are typically also not yet named (and cannot be interviewed). Assessing into a vague future should be avoided in the same way as making assumptions based on the pure execution of the development phase. Defining a new process is not required, since the PRM tailored to project needs offers for assessments in each lifecycle phase the needed processes to be evaluated. The goal is to help to establish an awareness if basic preparations have been considered

in the individual project process definitions, strategies, approaches and activities. Evidence of preparation can be found in, for example, how maintenance is anticipated in areas like

- project management,
- project agreements and documentation,
- configuration management,
- architectural design,
- software construction practices, or
- verification and validation.

The focus should be on how well the current project team has structured the product and its documentation for an eventual handover and long-term maintainability rather than on the future performance of a separate maintenance organization.

Project-, Problem Resolution-, Change Request- and Supplier Management

Even if the maintenance phase is decided to be carried out by a different team, in a different organization or country, the current project team is typically involved in the planning, preparation and executing of the handover. This may also include the identification of, for instance, the required skills and knowledge, project documentation including risk lists, and engineering artifacts relevant to the future maintenance team. Assessors should evaluate whether the current team has acknowledged the later maintenance team as a stakeholder. Indications for this may answer questions like: “Do the regulations for decision-making (e.g., change requests) consider the future maintainability/ potential limitations, and are sufficient levels of management identified as the maintenance may be a company level activity?” “Has an escalation and information management been agreed to by the customers and suppliers that includes regulations and contacts for post-development phases?” Are platform or supplier teams informed about customer project specific regulations and were the respective commitments obtained to ensure awareness and later availabilities?”

This may not only concern the development work results themselves but also aspects like backup and storage timelines, traceability of component sources (e.g., via BOM methods), reachability via diagnosis functions, assessments and auditability or even adapted approval responsibilities and proceedings. Moreover, how project-specific risks with respect to the challenges of the maintenance phase are handled and communicated to an overall organizational entity (e.g., maintenance department, organizational risk management) can be evaluated.

Project Documentation

Typically, the overall project documentation is sparse, and the project teams may argue that they are sufficiently aware of the general status, the ceremonies as well as documentation of architecture, design, code, models or other work results. A later maintenance organization, however, may have never participated in the active development phase of the assessed project. They may have a different cultural background leading to very different interpretations or might have been trained based on other paradigms, eliminating the likeliness that they would make the same assumptions (e.g., “30 degrees” in a requirement might be interpreted in Europe most likely to be Celsius while in the US it would be Fahrenheit).

The documentation and metrics (e.g., requirements, general coverage, comment code density) therefore should not only serve the current project team in the specific organizational unit but instead be fit for use in a more general manner. Considerations to be made may include, for instance, minimum criteria for documentation, glossaries, format, and applying internationally common language aspects and – if those are also checked as criteria in the regular work result – reviews of the project (as this may be considered for the SUP.1 rating).

In a maintainability-oriented project, the awareness about good documentation is key to enable human understanding across time, cultures, and organizational changes.

Configuration Management & Tools

Configuration Management (CM) is usually the backbone of successful projects, and its importance becomes even greater when the project is required to support a maintenance phase. In such cases, CM may not only ensure the integrity of all work products during development but also support the structured preservation of either the entire project environment and data for the maintenance phase or at least key artifacts. Further, Configuration-, Test-, Build- and Release Systems and toolchains may need to be documented in a way that they can be re-established when this becomes necessary. An additional strategic consideration could be to foresee redundant setups for the toolchains, in order to avoid a critical vendor dependency. Tool vendors may discontinue support for older software versions or move to cloud-based solutions which cannot be adapted to the original project demands. With maintainability in mind a setup/ tool/ license-oriented approach and risk management may cover all obligations of the full lifecycle and not just the immediate needs of the current development phase.

Some typical questions that could be interesting to a later maintenance organization: “Which compilers in which setting were in use?” “Which targets in which configuration had been used for testing/ building/ approval?” “Which work-arounds and hacks had been applied to reach certain aspects or execute on preliminary targets?” “Which operating system was used with which drivers and configurations?” “Which software tools were in use that were not on the official company IT-supported tool list?”

Information management may be established in a way that enables maintainers later to go beyond coverage reports and easily understand which artifacts of a baseline belonged logically to each other and form a consistent compound to be adapted. This could cover aspects like the applicable set of requirements, architecture, designs, known limitations/accepted risks in previous product releases as well as the specific units, components, test cases and the related variant and version information (e.g., which customer received what at which point in time based on which decisions, development state, tool setup and settings).

Cybersecurity concepts as well as established organizational information management policies, may conflict with a pure “we keep

everything” approach. This may have storage-cost-reasons or simply aim at preventing everyone from access to information like specific cybersecurity mechanisms, keys or privacy-related data (e.g., diagnostic data including geolocation of specific vehicles, real-world data used for training or testing, payment information in applications). In return it should be verified that this kind of information is available to a selected group of maintainers not only in the specific organization but also in alignment with customers and suppliers.

For projects utilizing machine learning, artificial intelligence or other data intensive approaches, it may be wise to clarify which data is necessary to be kept (post-SOP) to enable efficient re-training as well as to satisfy the release-oriented verification and validation strategies.

Architectural and Detailed Design

The ISO/IEC 25010 may be used in a project as meaningful foundation for the agreed upon quality properties of the work products and the related sub characteristics of maintainability: modularity, reusability, analyzability, modifiability and testability could be considered in the architectural principles and actively pursued throughout the design processes to “implement” maintainability. A process performance that emphasizes well-documented interface definitions as well as suitable documentation of the architecture and design decisions, directly supports future maintenance activities. This includes documenting known limitations, design trade-offs, accepted risks, further reasons for selecting a particular solution as well as potential variation points for future adaptations. As no perfect freedom from interference implementation can be assumed in any case, the maintenance-oriented architectural documentation should empower the maintainer to effectively understand the potential influences and risks of adaptations. Cybersecurity strategies or obligations often expect the ability to act fast, precisely and effectively in order to prevent further damage. Without proper architectural preparation and overall project documentation, especially unplanned updates may require very specific and rare knowledge that might not be available on short notice.

A project deemed to be relevant to maintenance may have considered in its architectures, for instance, providing the capability to independently switch off specific features or to add more features by ensuring upwards and downwards compatibility and a certain amount of available computing power and memory beyond the initial project scope. The successful and seamless application of patches may not only require the mechanisms for the update execution itself, but also sufficient extra space to download and store the update.

Use of Free and Open-Source Software (FOSS)

The inclusion of Free and Open-Source Software (FOSS) in a project and throughout the supply chain introduces additional

maintenance challenges and risks. Assessors can determine whether the project has thoroughly considered how updates will be acquired, especially in scenarios where the community dissolves, the software is forked, or future versions may become incompatible to the given project environment and boundaries (e.g., to be supported hardware).

Beyond the expectation to comply with current license regulations, a project might have to consider how to react to potential license model changes potentially decided by the communities in the future. This may directly impact the applicability of future patches to the given customer regulations (e.g., blacklists). A potential project approach could be that alternative options are directly foreseen/ enabled in the architecture (e.g., by usage of standardized interfaces and wrapper mechanisms). Furthermore, it could be necessary to consider how vulnerabilities in an unknown code base, without proper documentation, could be treated in case the communities' updates are insufficient, or not applicable at all.

Many companies easily overlook the aspect that no binding contracts with the communities exist and no specific resolution can be enforced. Thus, the project could be demanded to assess whether it is legally permitted to modify the FOSS code under applicable license terms, and whether the future maintenance team will have the necessary technical expertise and capacity to support these components. If these aspects are not properly addressed, FOSS may become a significant liability. In some cases, the risks may be high enough to conclude that FOSS should be avoided altogether, unless its use is supported by a clear, feasible, and sustainable management approach.

Verification and Validation (V&V)

The approaches for verification and validation (V&V) in maintenance related projects may not only consider the current and agreed requirements, but also whether the solution is enabled to support future extensions. This may include scenario-based checks if mechanisms for updates like flashing (e.g., Over-the-Air updating) are sufficiently working in various configurations and variants and also require cross-project customer (e.g., cloud/backend) and

supplier involvement in related campaigns. If modularity is a paradigm of the architecture, this should be also reflected in the V&V approaches by checking if the system is still stable, robust, reliable and functional in case elements are removed or exchanged.

Today`s V&V approaches often focus on the question if the defined package does what it is supposed to do, but seldomly on the question: “What happens if something is to be changed later?” In the context of cybersecurity, the V&V approaches may have to reflect, for instance, questions like if and how diagnosis and update interfaces can be monitored over time (e.g., up-to-date sufficiency of cybersecurity mechanisms). Thus, the overall setup may be required not only to be used in case of specific maintenance events but generally be kept up to date with new cases to cover the latest hacking approaches.

The V&V approach and toolchain may also require to be over-sized to be able to cover the lack of capacities at suppliers (e.g., out of business or it is contractually agreed that the V&V sets are handed over), FOSS community outcomes (e.g., no coverage agreements) or those of other temporal project partners. As stated for other processes, the overall documentation and setup may need to consider the fact that it must be possible to be utilized by future organizations. Topics like experience-based testing are a viable asset to a test strategy, while the project may need to find ways to document this experience to make it available to later generations of testers. V&V suites may be structured in a way that ensures they can be reused, understood, and adapted later. Just like source code, test code and documentation should be written clearly, in natural language, and self-explanatory, suitable for someone who did not originally create it. Furthermore, for data intensive projects, the approaches for V&V in the maintenance phase may need to be elaborated during project execution to enable the configuration management and storage of large data sets while avoiding the high costs of re-acquiring data.

3 Rating Guidelines on Process Performance (Level 1)

3.1 ACQ.4 Supplier Monitoring

The purpose is to track and assess the performance of an external contract-based supplier company against agreed commitments.

3.1.1 General information

The customer (which can also be a supplier acting as a customer) must introduce a supplier monitoring process for the following relationships with external contract-based suppliers:

- supplier develops a product or component based on the customer requirements
- supplier delivers a product or component with off-the-shelf sub-components based on customer requirements

Interfaces between supplier and customer have to be established for exchanging, monitoring, and tracking all relevant information between both parties. Even for component deliveries (e.g., commercial off-the-shelf), interfaces shall be set up and maintained for managing changes and problem reports.

3.1.1.1 Monitoring all contract-based suppliers

All project-relevant contract-based suppliers must be tracked and their performance against the agreed requirements assessed. Based on the context of the project, this may include suppliers for engineering services, commercial off-the-shelf products, firmware, etc. Excluded from ACQ.4 are suppliers delivering products without any support (e.g., open-source software).

3.1.1.2 Completeness of agreements with supplier

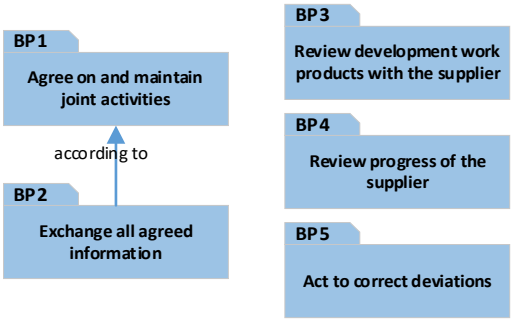
Agreements between supplier and customer have to be established and maintained, which cover:

- supplier's content and scope
- relevant requirements and standards from the customer
- interface agreement
- exchanged information between customer and supplier
- joint activities across the interfaces, such as joint problem and change management, and joint reporting and reviews
- responsibilities and stakeholders
- escalation mechanism

Examples for such agreed documents are DIAs (Distributed Interface Agreements), SOWs (Statements Of Work), license agreements, etc.

3.1.2 Rating rules within the process

The following figure shows the relationships between ACQ.4 base practices as well as their relationships to other processes:



Rating rules:

[ACQ.4.RL.1] If the indicator BP1 is downrated due to incomplete agreements about exchanged information between customer and supplier, the corresponding indicator BP2 shall be downrated.

3.1.3 Rating rules with other processes

None.

3.2 SPL.2 Product Release

The purpose is to control the release of a product to the intended customer.

3.2.1 General information

During product development, the functional content that is agreed upon with the customer or a development partner is usually implemented in an iterative and incremental way. The prioritization of the functionalities to be realized is done via the requirements analysis processes SYS.2, SWE.1, HWE.1 and MLE.1.

3.2.2 Rating rules within the process

3.2.2.1 Release scope

The sequence of implementing these functionalities is determined in the release scope. The release scope is not necessarily a separate document. The relevant release planning aspects can be part of the project´s schedule.

Rating rules:

> **[SPL.2.RL.1]** If the scope of the current release is not identified in detail (features and/or functions per release), the indicator BP1 shall be not rated higher than P.

A release package consists usually of the released product, the information about the product and the release, and supporting tools as needed.

3.2.2.2 Release note

Changes and improvements that are made to the content of the delivered product compared to previous releases shall be documented in the release note.

Rating rules:

> **[SPL.2.RL.2]** If the release notes do not describe changes compared to previous releases, BP6 shall be downrated.

3.2.3 Rating rules with other processes

The information regarding verification and validation results of the product must be taken into account.

Rating rules:

> **[SPL.2.RL.3]** If the content in the release notes is inconsistent with the results from VAL.1, SYS.4, SYS.5, SWE.4, SWE.5, SWE.6, HWE.3, HWE.4, then SPL.2.PA1.1 shall not be rated higher than P.

3.3 SYS.1 Requirements Elicitation

The purpose is to gather, analyze, and track evolving stakeholder needs and requirements throughout the lifecycle of the product and/or service to establish a set of agreed requirements.

3.3.1 General information

3.3.1.1 Direct or indirect traceability to customer requirements

Customer requirements specifications (being stakeholder requirements) may include sub-domain requirements or design constraints (e.g., software, hardware) that clearly do not address the system requirements (SYS.2) but instead the software and hardware sub-domains (SWE.1, HWE.1). Also, customer specifications often include design constraints, namely, exact demands on 'how' something must be done or implemented, thereby addressing the system architecture (SYS.3) or the software and hardware designs (SWE.2, SWE.3, HWE.2) directly. In contrast to design constraints, requirements generally describe the problem space leaving open the solution space.

Since customer requirements specifications are to be distilled and transformed by the supplier into system requirements SYS.2 (see Figure 3-1), such sub-domain customer requirements and design constraints must not “just” be taken as is without further evaluation. The original intention and problem behind such sub-domain customer requirements and customer design constraints are to be identified. This may lead to new and necessary (but formerly unidentified) system-level requirements that may or may not replace those customer sub-domain requirements and design constraints. This is exactly what is behind the purpose of SYS.1: eliciting the actual wants, needs, and intentions behind seemingly fixed stakeholder requirements, discussing them, and finally agreeing on the results. This helps to make sure that no “hidden” system requirement is neglected.

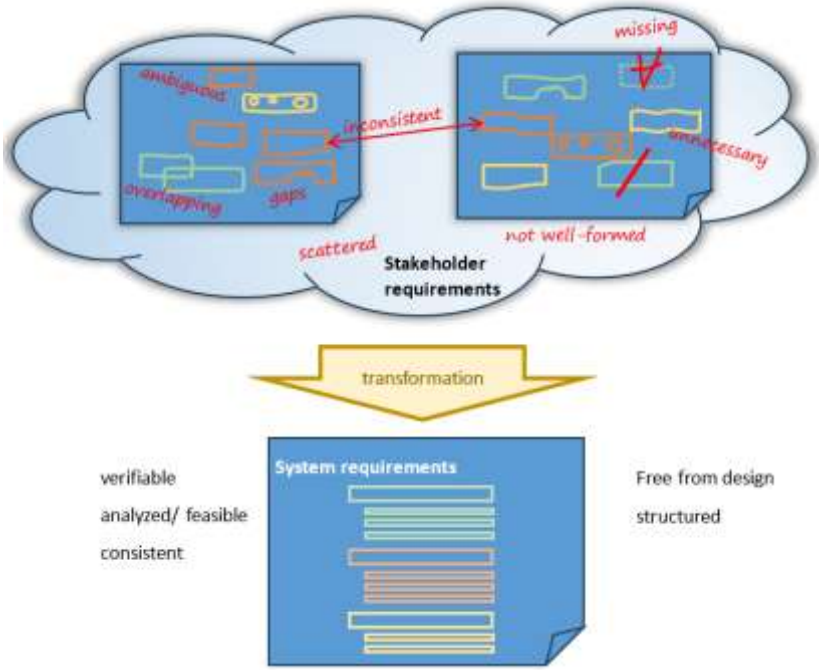


Figure 3-1: Transformation of stakeholder requirements into system requirements

In some cases, if after a sound evaluation clarity is reached on why

- the customer needs to insist on the sub-domain requirement or on design constraint – even if more economically or technologically more suitable design solutions might be possible – (note that the supplier is in charge of the overall solution space and the customer may not have a full overview of the impact of such design constraints on the supplier's overall solution), or
- the sub-domain requirement or design constraint, respectively, has no unidentified impact or side-effect on existing system requirements and system architectural decisions,

then the customer design constraint or customer sub-domain requirement may still actually be confirmed as such. In such cases,

the respective sub-domain requirement or design constraint does not need to be repeated at (i.e. copied to) different levels of requirements specifications or designs, respectively. Instead, it may be directly put into the corresponding sub-domain requirement specification (SWE.1/HWE.1) or reflected at the respective design level (SYS.3, SWE.2, SWE.3, HWE.2), with direct traceability to the customer specification only. However, this must be agreed to by the sub-domain and system representatives. Such agreements further represent and promote early interaction of different domain experts, thereby contributing to reducing requirements interpretation issues early.

3.3.1.2 Vertical tracing to stakeholder requirements

There may be customer requirements (being stakeholder requirements) that address or represent properties or characteristics of the direct physical end product. Other customer requirements may address work product or process characteristics like MISRA guidelines, coding guidelines, reaching Automotive SPICE® capability level x (see Table 3-1 for examples).

Only the former, but not the latter, are subject to

- vertical downward traceability. Their identification can be documented via, e.g., requirements structuring or tool-based attributes (see Table 3-1 for examples), and
- horizontal traceability to verification measures (as defined in the Automotive SPICE® PAM glossary) in SYS.4, SYS.5, SWE.4, SWE.5, SWE.6, HWE.3, and HWE.4.

In contrast, stakeholder requirements that address work product or process characteristics are still to be verified or validated, respectively (as any requirement must be). This can be done, for instance, by

- either commenting in, e.g., the customer specification that, e.g., process requirements are already satisfied by a supplier standard process or work instruction. In this case, they do not need to be added to the SYS.2 requirements,

- or if not already covered as stated above, by still adding them to the SYS.2 requirements to not lose their verification.

Note, however, that such verification is not subject to horizontal traceability to SYS.4, SYS.5, SWE.4, SWE.5, SWE.6. HWE.3, and HWE.4 because the evidence will not be verification measures according to the definition in the Automotive SPICE® PAM glossary (see Table 3-1 for examples). Further note that Automotive SPICE® is not a lifecycle model (see Automotive SPICE® PAM subchapter 3.3.4). This means that, for example a system testing department Is not automatically responsible for verifying, for example, process requirements.

Table 3-1: Non-exhaustive examples

	Vertical tracing?
Requirements for work products/artifacts, e.g.: • MISRA rules • coding guidelines • code metrics	Not traced, but to be evidenced by, e.g., corresponding verification results
Process requirements, e.g.: • Level 2 process capability according to Automotive SPICE® • compliance with process standards	Not traced, but to be evidenced by, e.g., assessment and audit reports
Direct functional end product characteristics, e.g.: • CAN matrix • behavior	To be traced and evidenced by, e.g., verification/validation results
Direct non-functional end product characteristics, e.g.: • weight • response times	To be traced and evidenced by, e.g., verification/validation results

Production requirements, e.g.: • soldering process • capability of inspection equipment	Not traced, but to be evidenced by verification results

3.3.2 Rating rules within the process

Rules for rating consistency between the BPs in this process are not defined. This is due to the nature of BPs as describing separate concerns that will be addressed individually. Furthermore, a process attribute shall be rated based on the process performance indicators, namely, not based on a subset.

3.3.3 Rating rules with other processes at level 1

None.

3.4 SYS.2 System Requirements Analysis

The purpose is to establish a structured and analyzed set of system requirements consistent with the stakeholder requirements.

3.4.1 General information

Stakeholder requirements can contradict one another, such as legal regulations versus specific customer needs. Such cases need to be clarified with the customer.

3.4.1.1 Iterative vs. incremental development

Normally, the functional content in the product changes iteratively and incrementally: it evolves across releases. The term “increment” can be understood as adding a feature or element that did not exist before (analogy: building a house). The term “iteration” can be understood as refining, or adapting, an existing feature or element (analogy: a sculptor working on a sculpture).

Therefore, the complete set of requirements of the final product does not necessarily have to be available at the project start. Rather, release scopes agreed with the customer will define increments and iterative rework. In this respect, requirements creation can be driven by release definitions over time.

3.4.1.2 Analysis of impact on the system context

SYS.2.BP1 specifies the requirements for the system under consideration alone, namely, the ones the system shall implement. In contrast, BP4 asks for the impact and consequences the system has on its system context because of those requirements. In requirements engineering the term “system context” is a defined technical term. Its meaning denotes anything outside or beyond the boundary of the system under consideration in SYS.2. Elements in the system context like

- human users,
- other mechatronic systems, or
- other controller devices

trigger the system's functionalities, are receivers and users of results of the system's functionalities, interact with the system, or have interfaces with the system. Note that "interface" here may not only refer to direct interaction interfaces but also indirect ones. For example, another system installed in very close proximity of the system under consideration may suffer from its heat or radiation emission.

In alignment with subchapter 2.3.2, examples are:

- vehicle – noise, exhaust, leakage (e.g., fuel, oil, water, gas, refrigerants)
- end user interfaces – stress, distraction, discomfort or fatigue because of poorly designed or over-designed HMIs
- mechatronic system – vibration, acoustics, forces (e.g., tailgate, automatic door access system), leakage (oil, refrigerant…), stored energy (e.g., pre-loaded springs), moving or rotating elements, kinetic energy, electrostatic and electromagnetic phenomena, electrically live parts, debris of worn parts, etc.
- ECU – signal quality, emission of heat or radiation, size conflicting with the designed mounting space, weight conflicting with connection technology used in the system context

Such impact on the system context needs to be communicated back to the owners of the respective elements in the system context. This impact may or may not lead to changes of the requirements of the system under consideration of the system context.

Note that for SWE.1 and HWE.1 the decision was to keep the term operating environment for two reasons:

1. The usage of the term "system" might not appear intuitive for processes that deal with software and hardware only.

2. Software runs on a target which is better expressed by using "operating environment."

3.4.1.3 Structuring of requirements and mapping to releases

A possible approach to prioritizing requirements is the allocation of requirements to releases. The use of such an approach will imply that the content of the next and future releases is defined.

Further ways of structuring requirements are, for example, using attributes in tools for categorization, a hierarchical headline structure with an adequate depth.

Rating rules:

> **[SYS.2.RL.1]** If there is no evidence for prioritization but a separate release plan consistently mapping system functionality to future releases, then SYS.2.BP2 shall not be downrated.

3.4.1.4 Traceability and consistency

See also subchapter 3.3.1 of SYS.1 and 2.3.4 here.

3.4.2 Rating rules within the process

3.4.2.1 System requirements

Rating rules:

> **[SYS.2.RL.2]** If different approaches of documenting requirements are used concurrently (e.g., word processor file, Application Lifecycle Management tool, database), then SYS.2.BP1 shall not be downrated.
>
> **[SYS.2.RL.3]** If not all system requirements are derived from, and traced to, the customer requirements but to internal standard requirements or to a product line/platform according to a reuse or application strategy, then SYS.2.BP1 and SYS.2.BP5 shall not be downrated.
>
> **[SYS.2.RL.4]** If not all system requirements of the final product are available because of release-driven incremental development, then SYS.2.BP1 and SYS.2.BP2 shall not be downrated.

3.4.2.2 Structuring of requirements

To support the understanding in subchapter 2.3.3.3:

Rating rules:

[SYS.2.RL.5] If the notions “functional” and “non-functional” are the only requirements structuring, categorization, or classification criterion, then SYS.2.BP2 shall be rated as N.

[SYS.2.RL.6] If the notions “functional” and “non-functional” are not used as a structuring, categorization, or classification criterion, then SYS.2.BP2 shall not be downrated.

3.4.2.3 Analysis of requirements

The indicator SYS.2.BP3 says “*Analyze system requirements. …and to support project management regarding project estimates.”*

This means, for example:

- Consider an analysis that was done on a set of 100 requirements together with the project manager during a project progress meeting. As a result, 40 out of the 100 requirements were decided to be rejected, therefore being attributed as such with an accompanying comment providing expectations.

- Consider a set of 10 requirements that are planned for the next release. The development team reports to the project manager that this is no longer feasible due to resource constraints. The decision is to not change the status of those 10 requirements but to reallocate them to future releases in agreement with the relevant stakeholders. This can be shown by comparing the release plans (which is the process context of MAN.3 but not SYS.2).

Analysis of requirements can be done by means of using, for instance, tool-based attributes, or comments added to the requirements text.

The analysis of system requirements is the basis for a correct implementation. Even though requirements sometimes appear very simple, a well-founded analysis shall be conducted for those

requirements. The scope and appropriateness of the analysis depends on the context of product (e.g., platform). The results of analysis can vary from a simple attribute to a complex simulation or the building of a demonstrator to evaluate the feasibility of system requirements.

Analysis of requirements may – but does not have to – be done as a part of a review.

The reason for having associated this BP with the supporting of project management regarding project estimates is the following: revising system requirements by means of an analysis may (re-)define the scope of work. Once this solution is settled, SYS.5 will select verification measures for addressing exactly that problem space. In other words, SYS.5 itself does not change that problem space, therefore does not alter the overall project's scope of work.

Rating rules:

[SYS.2.RL.7] If analysis results of requirements are not demonstrated by means of separate analysis reports or review records but by means of, e.g., tool-supported attributes or tool-supported commenting, then SYS.2.BP3 shall not be downrated.

[SYS.2.RL.8] If analysis of system requirements regarding technical feasibility is covered by effective risk management, then SYS.2.BP3 shall not be downrated.

[SYS.2.RL.9] If analysis results of system requirements regarding impact on estimates is not consistently used by project management (MAN.3), then SYS.2.BP3 shall not be downrated.

3.4.2.4 Traceability and consistency

System requirements are derived from stakeholder requirements. During the process of analysis of system requirements, inconsistencies between stakeholder requirements and system requirements may occur as the customer does not always update his requirements.

Rating rules:

[SYS.2.RL.10] If a system requirement is no longer consistent with stakeholder requirements because of a meaningful adaptation, but the stakeholder does not adapt his respective requirement correspondingly and evidence of the agreement is available, then SYS.2.BP5 shall not be downrated.

See also subchapter 3.3.1 of SYS.1 and 2.3.4 here.

3.4.3 Rating rules with other processes at level 1

None.

3.5 SYS.3 System Architectural Design

The purpose is to establish an analyzed system architecture consistent with the system requirements.

3.5.1 General information

The architecture is mainly non-functional requirements-driven, and system architectural design decisions may lead to iterative system requirements rework. This can be the case if, for example, it is found that two non-functional requirements cannot technically be met at the same time.

Example:

A signal shall be processed by the system with a cycle time of 10 msec, while throughput is limited to 1000 bus messages per minute (resulting best-case in 60 msec per bus message).

3.5.1.1 Specifying a system architecture

The system architectural design is the highest level of a design description of the system encompassing a collection of views dealing with different aspects (e.g., static structure, functional decomposition dynamics etc.). These views are architecture visualizations required for communication, discussion, reviews, analysis, evaluation, planning, change request analysis, impact analysis, maintenance, etc. of the system.

There is no common definition of which views are required and no criteria for the completeness of such views. However, essential views are

1. (at least one) static view providing an overview of the structure; and
2. (at least one) dynamic view describing the designated behavior behind system functionalities.

In most cases the system architectural design is a graphical representation of the system supplemented by textual explanations.

Static system architecture views allow the recursive decomposition of the system into manageable elements with high cohesion and low coupling. This decomposition supports the assignment of requirements to the architectural elements and will help the organization to distribute the work. An architectural design may need to include elements that are developed externally, such as platform, third-party parts, COTS, etc.

At the stage of system architectural design, the allocation may reasonably be done at the level of suitable requirement clusters (e.g., a chapter representing a system service or use case in a requirements specification) but not necessarily at the level of a single requirement.

3.5.1.2 Analysis of the system architecture (BP3) vs. SYS.4

Note for SYS.3.BP3, it explains that techniques for analyzing the system architectural design can be, for instance, prototyping, simulations, or qualitative analyses (e.g., FMEA approaches). This might raise the questions whether SYS.3.BP3 and SYS.4 are somewhat redundant, and if the process purposes SYS.3 and SYS.4 are overlapping. However, this is not the case, because:

- SYS.4 verifies if a given physical sample reveals the characteristics defined by the system design. This means, the “object-under-verification” is the physical sample.
- In contrast SYS.3.BP3 determines whether the system architectural design itself, i.e. whether the documented information is appropriate. This means the “object-under-analysis” is not a physical sample but the documentation representing the design decisions.

3.5.2 Rating rules within the process

3.5.2.1 Analyzing the system architecture

The following BP has been introduced in SYS.3 (and similarly in SWE.2):

> *SYS.3.BP3: Analyze system architecture. Analyze the system architecture regarding relevant technical design aspects related to the product lifecycle,*

and to support project management regarding project estimates, and derive Special Characteristics for hardware elements. Document a rationale for the system architectural design decision.

This is to be able to reflect, for example:

- cybersecurity, such as vulnerability analyses
- functional Safety, such as safety analyses and dependent failure analyses according to ISO 26262 *[ISO26262]*
- robustness needs

The reason for having associated this BP with the supporting of project management regarding project estimates is the following: Revising system architectural solutions because of an architectural analysis may (re-)define the scope of work. Once this solution is settled, SYS.4 will select verification measures for addressing exactly that solution space. This is to say, SYS.4 itself does not change that solution space and therefore does not alter the overall project's scope of work.

Rating rules:

> **[SYS.3.RL.1]** If non-quantitative analysis approaches or techniques are used and adequate but no quantitative ones, then SYS.3.BP3 shall not be downrated.

3.5.2.2 Traceability and consistency

See also subchapter 3.3.1 of SYS.1 and 2.3.4 here.

3.5.3 Rating rules with other processes at level 1

None.

3.6 SYS.4 System Integration and Integration Verification

The purpose is to integrate systems elements and verify that the integrated system elements are consistent with the system architecture.

3.6.1 General information

3.6.1.1 Why there is no “production data compliant sample” BP in SYS.4/SYS.5

The processes HWE.3 and HWE.4 include such a BP because hardware production data compliance does not automatically imply design compliance. In contrast, SYS.4 and SYS.5 do not have such a BP, as there is no corresponding notion of ‘system production data’. Furthermore, the process purposes of SYS.4 and SYS.5 are about verifying that a sample is compliant with the design and requirements, respectively. Therefore, a “production data compliance sample” BP in SYS.4/SYS.5 would be redundant with these process purposes.

3.6.1.2 Specify verification measures for system integration

SYS.4.BP1 requires the identification of the necessary verification infrastructure and environment setup. This may be supported using simulation of the environment such as hardware-in-the-loop simulation, vehicle network simulation, digital mock-up.

Verification results can support the update of simulation models.

3.6.1.3 Selecting verification measures

Evidencing the selection of verification measures could be achieved by, for instance,

- a document depicting the release content, including a list of verification measures to be done for it, or
- a meeting with the verification personnel and, e.g, a person responsible for the system design.

3.6.1.4 Analysis of the system architectural design (BP3) vs. SYS.4

See subchapter 3.20.1.3.

3.6.2 Rating rules within the process

3.6.2.1 Verification measure definition

Rating rules:

> **[SYS.4.RL.1]** If entry/exit criteria are reasonably specified for a set of verification measures instead of each individual verification measure, then SYS.4.BP1 shall not be downrated.

3.6.2.2 Automation of verification measures

Rating rules:

> **[SYS.4.RL.2]** If a verification measure is automated and the correctness, completeness, and consistency of the corresponding scripts and programs are not addressed in the verification measure definition, then SYS.4.BP1 shall be downrated.

3.6.2.3 Exploratory testing vs. traceability/consistency

The testing state of the art does not only comprise testing derived from requirements, but also exploratory testing based on experience, such as “error guessing based on knowledge”. This is valuable as it adds to the quality of the product. Therefore, exploratory tests that are based on experience cannot, by definition, be traced to the system requirements.

Still, traceability is needed between exploratory test cases and their results.

Rating rules:

> **[SYS.4.RL.3]** If verification measures represent exploratory tests, which, by definition, cannot be traced to the system architectural design, then SYS.4.BP4 shall not be downrated.

3.6.2.4 Traceability and consistency

See also subchapter 3.3.1 of SYS.1 and 2.3.4 here.

3.6.3 Rating rules with other processes at level 1

3.6.3.1 Verification measures selection vs. release plans

Rating rules:

> **[SYS.4.RL.4]** If selection of verification measures is properly done but based on an inadequate or incomplete release plan, then SYS.4.BP2 shall not be downrated.

3.7 SYS.5 System Verification

The purpose is to ensure that the system is verified as consistent with the system requirements.

3.7.1 General information

3.7.1.1 Why no “production data compliant sample” BP in SYS.4/ SYS.5

See subchapter 3.6.1.

3.7.1.2 Specify system verification measures

SYS.5.BP1 requires the identification of the necessary verification infrastructure and environment setup. This may be supported using simulation of the environment, such as hardware-in-the-loop simulation, vehicle network simulation, and digital mock-up.

Verification results can support the update of simulation models.

3.7.2 Rating rules within the process

3.7.2.1 Verification measure definition

Rating rules:

> **[SYS.5.RL.1]** If entry/exit criteria are reasonably specified for a set of verification measures instead of each individual verification measure, then SYS.5.BP1 shall not be downrated.

3.7.2.2 Automation of verification measures

Rating rules:

> **[SYS.5.RL.2]** If a verification measure is automated and the correctness, completeness, and consistency of the corresponding scripts and programs are not addressed in the verification measure definition, then SYS.5.BP1 shall be downrated.

3.7.2.3 Exploratory testing vs. traceability/consistency

The testing state of the art not only comprises testing derived from requirements but also exploratory testing based on experience, such as “error guessing based on knowledge.” This is valuable as it adds to the quality of the product. Therefore, exploratory tests that are based on experience cannot, by definition, be traced or consistent with the software requirements.

Still, traceability is needed between exploratory test cases and their results.

Rating rules:

> **[SYS.5.RL.3]** If for those verification measures which represent exploratory tests, no traceability to the system requirements is available, then SYS.5.BP4 shall not be downrated.

3.7.2.4 Traceability and consistency

See also subchapter 3.3.1 of SYS.1 and 2.3.4 here.

3.7.3 Rating rules with other processes at level 1

3.7.3.1 Verification measures selection vs. release plans

Rating rules:

> **[SYS.5.RL.4]** If selection of verification measures is properly done but based on an inadequate or incomplete release plan, then SYS.5.BP2 shall not be downrated.

3.8 SWE.1 Software Requirements Analysis

The purpose is to establish a structured and analyzed set of software requirements consistent with the system requirements and the system architecture.

3.8.1 General information

3.8.1.1 Iterative vs. incremental development

Normally the functional content in the product changes iteratively and evolves incrementally across releases. The term “increment” can be understood as adding a feature or element that did not exist before (analogy: building a house). The term “iteration” can be understood as refining, or adapting, an existing feature or element (analogy: a sculptor working on a sculpture).

Therefore, the complete set of requirements of the end product does not necessarily have to be available at the project start. Instead, release scopes agreed with the customer will define increments and iterative rework. In this respect, requirements creation can be driven by release definitions over time.

3.8.1.2 Impact on the operating environment

SWE.1.BP1 specifies the requirements for the software under consideration alone, namely the ones the software shall implement. In contrast, BP4 asks for the impact and consequences the software has on its operating environment because of those requirements. Its meaning denotes anything outside or beyond the boundary of the software under consideration in SWE.1. elements in the operating environment like

- human users, e.g., in case of infotainment systems,
- the target on which the software is running, or
- stress, distraction, discomfort or fatigue resulting from poorly designed or over-designed HMIs.

Such impact on the operating environment needs to be communicated back to be able to make changes. Otherwise, this

impact may be used to iterate the requirements of the software under consideration.

3.8.2 Rating rules within the process

3.8.2.1 Software development without system requirements

In case of software development only, the software requirements may refer directly to the stakeholder requirements. Consequently, consistency and bidirectional traceability must be ensured between stakeholder requirements and software requirements.

Rating rules:

> **[SWE.1.RL.1]** In the case of software development only, if the traceability and consistency from software requirements to stakeholder requirements are established then SWE.1.BP5 shall not be downrated.
>
> **[SWE.1.RL.2]** If some software requirements are not derived from system requirements but from platform or product line requirements and are not in contradiction to other requirements, then SWE.1.BP1 shall not be downrated.
>
> **[SWE.1.RL.3]** If software requirements are not derived from system requirements but from stakeholder requirements which do not affect system requirements or the system architecture and this is agreed with software and system representatives, then SWE.1.BP1 shall not be downrated.

3.8.2.2 Structuring of requirements

Software requirements can be grouped or categorized to support an overview and prioritization. See also subchapter 2.3.3.2 here.

Rating rules:

> **[SWE.1.RL.4]** If “functional” and “non-functional” are the only requirements categorization or classification criterion, then SWE.1.BP2 shall be rated as N.

3.8.2.3 Requirements mapping to releases

A possible approach to prioritizing requirements is the allocation of requirements to releases. The usage of such an approach will imply that the content of the next and future releases is defined.

Rating rules:

> **[SWE.1.RL.5]** If there is no direct evidence of prioritization of the software requirements but a separate release plan is consistently mapping these software requirements to the future releases, then SWE.1.BP2 shall not be downrated.

> **[SWE.1.RL.6]** If the software requirements that are mapped to a particular release do not match with the system requirements mapped to the same release, then SWE.1.BP2 shall be downrated.

3.8.2.4 Analysis of requirements

See also subchapter 3.3.1 of SYS.1.

The indicator SWE.1.BP3 requires

"*Analyzing software requirements. …and to support project management regarding project estimates.*"

This means, for example:

- A set of 100 requirements exists. An analysis was done together with the project manager during a project progress meeting. As a result, 40 out of the 100 requirements were decided not to be considered, therefore being attributed as "rejected" with an accompanying comment providing expectations.

- A set of 10 requirements were planned for the next release. The development team reports to the project manager that this is no longer feasible due to resource constraints. The decision is to not change the status of those 10 requirements but to reallocate them to future releases. This can be shown by comparing the

release plans (which is the process context of MAN.3 but not SWE.1).

Analysis of requirements can be done by means of using, for example, tool-based attributes or comments added to the requirements text.

The analysis of software requirements is the basis for a correct implementation. Even though requirements sometimes appear very simple, a well-founded analysis must be conducted for those requirements. The scope and appropriateness of the analysis depends on the context of product (e.g., platform). The result of analysis can vary from a simple attribute to a complex simulation or the building of a demonstrator to evaluate the feasibility of software requirements.

The reason for having associated this BP with the supporting of project management regarding project estimates is the following: Revising software requirements by means of an analysis may (re-)define the scope of work. Once this solution is settled, SWE.6 will select verification measures for addressing exactly that problem space. This is to say, SWE.6 itself does not change that problem space, therefore does not alter the overall project's scope of work.

Rating rules:

[SWE.1.RL.7] If analysis results of requirements are not demonstrated by means of separate analysis reports or review records but by means of, e.g., tool-supported attributes or tool-supported commenting, then SWE.1.BP3 shall not be downrated.

[SWE.1.RL.8] If the analysis of software requirements about technical feasibility is covered by risk management, then SWE.1.BP3 shall not be downrated.

[SWE.1.RL.9] If analysis results of software requirements regarding impact on estimates is not consistently used by project management, then SWE.1.BP3 shall not be downrated.

3.8.2.5 No redundant traceability paths

SWE.1.BP5 offers the possibility of having two paths for traceability:

a) between software requirements and the system architecture (SYS.3)
b) between software requirements and system requirements (SYS.2)

However, redundancy (i.e., using the two traceability paths for the very same software requirement at the same time) is neither intended by this BP nor meaningful. Furthermore, it is not intended to express that all or most of the software requirements should be traced to system requirements directly as a default. Which path appears more appropriate must depend on the actual content of the software requirement itself.

Example 1: system requirements ↔ system architecture ↔ software requirements

Consider the system requirements demanding a particular and coherent system service. As an architectural solution, different parts of software run on different microcontrollers or (maybe including dual core microcontrollers) on different PCBs, for example. Traceability between software requirements and system architecture would be needed here

- to allocate different software behavior to the different microcontrollers,
- as there are different communication mechanisms in between the different pieces of software.

Example 2: system requirements ↔ software requirements

The system interface requirements define a particular CAN matrix to be used. Such requirements can be traced to corresponding software requirements directly.

Rating rules:

[SWE.1.RL.10] If traceability is established for one path only but not for the other redundant path, SWE.1.BP5 shall not be downrated.

3.8.2.6 Traceability and consistency

See also subchapter 3.3.1 of SYS.1 and 2.3.4 here.

3.8.3 Rating rules with other processes at level 1

None.

3.9 SWE.2 Software Architectural Design

The purpose is to establish an analyzed software architecture, comprising static and dynamic aspects, consistent with the software requirements.

3.9.1 General information

3.9.1.1 The aim of software architectural design

Automotive SPICE® emphasizes that requirements are to be systematically broken down into lower levels via design decisions at the respective levels. The idea behind that is:

a) not to get overwhelmed with too much information thereby making mistakes ("divide and conquer" principle)
b) providing software documentation for future staff so that they do not have to inefficiently learn how the software works from the potentially massive source code
c) making sure implicit knowledge is not lost in the documentation and in the final product
d) to ensure requirements at the respective level are completely covered
e) finally, there is no superfluous software element (i.e., the SW does not contain more than the requirements need)

Also recall here the general legal obligation to produce supporting documents, product liability risks, etc.

In this respect, the software architectural design supports the above-mentioned expectations by describing the realization solution for the software requirements.

3.9.1.2 Software architectural design

Beyond a static and a dynamic view, there is no common definition on which views are required and no criteria for the completeness of the sum of views. The number and kind of views highly depends on the size and complexity of the product. There are some approaches in the industry that specify the kind of information that is required for

the view (“viewpoints” which are collections of patterns, templates, and conventions for constructing one type of view) and the integration of the views in a complete architectural design description.

In most cases the software architectural design is a graphical representation of the software supplemented by textual explanations.

Static software architecture views allow the decomposition of the software into manageable elements with high cohesion and low coupling. This decomposition supports the assignment of requirements to these architectural elements and will help to organize the distribution of the work packages to the developers. Software architectural elements developed outside of the assessment scope (e.g., open-source software, platform software, third-party software, etc.) are also to be included as dedicated elements in the software architectural design and have to be considered as well for interface analysis, dynamic behavior, resource consumption objectives, etc.

As appropriate the architectural elements are detailed further in the architectural design down to the components as the lowest level elements. The components consist of one or more units and are subject of the software detailed design process (SWE.3) (see “Annex C Terminology” of the PAM for definition of the terms element and component).

3.9.1.3 Regarding interrupts

Although, according to the state of the art, interrupt service routines shall not contain any domain logic behavior or complex algorithms, interrupt handling still represents parallel control or even data flows. Especially high interrupt loads may cause interferences in the application.

Therefore, Automotive SPICE® v4.0 considers it important to treat relevant interrupt handling as software units to reflect them in the software design, and the dynamic design in particular.

3.9.2 Rating rules within the process

Rules for rating consistency between the BPs in this process are not defined. This is due to the nature of BPs as describing separate concerns that will be addressed individually. Furthermore, a process attribute shall be rated based on the process performance indicators, namely, not based on a subset.

3.9.2.1 Analyzing the software architecture

The following BP has been introduced in SWE.2 (and similarly in SYS.3)

> *SWE.2.BP3: Analyze the software architecture. Analyze the software architecture regarding relevant technical design aspects, and to support the project management regarding project estimates. Document the rationales for the software architectural design decisions.*

This is to be able to reflect, for example:

- cybersecurity, such as vulnerability analyses
- functional Safety, such as Safety Analyses and Dependent Failure Analyses according to ISO 26262 *[ISO26262]*
- robustness needs

The reason for having associated this BP with the supporting of project management regarding project estimates is the following: Revising software architectural solutions because of an architectural analysis may (re-)define the scope of work. Once this solution is settled, SWE.5 will select verification measures for addressing exactly that solution space. In other words, SWE.5 itself does not change that solution space and hence does not alter the overall project's scope of work.

Rating rules:

> **[SWE.2.RL.1]** If non-quantitative analysis approaches or techniques are used and adequate but no quantitative ones, then SWE.2.BP3 shall not be downrated.

3.9.2.2 Traceability and consistency

See also subchapter 3.3.1 of SYS.1 and 2.3.4 here

3.9.3 Rating rules with other processes at level 1

None.

3.10 SWE.3 Software Detailed Design and Unit Construction

The purpose is to establish a software detailed design, comprising static and dynamic aspects, consistent with the software architecture, and to construct software units consistent with the software detailed design.

3.10.1 General information

3.10.1.1 The aim of a software detailed design

First, see subchapter 3.9.1.1 “The aim of software architectural design” for SWE.2.

The detailed design is a step in between the software architecture and the implementation in the source code. It makes the software even more comprehensible, thus further contributing to knowledge documentation. Furthermore, it helps to break down the software architectural solution into smaller pieces that are easier to understand, implement and verify ("divide and conquer" principle). Without a detailed design the step from requirements to the code level would still be too big, involving the risk of losing software understanding for others, and impacting maintainability. The detailed design specifies what each software unit shall do and how it interacts with other units, abstracting from programming-language specifics while still guiding the implementation.

The detailed design is not a mere graphical representation of the source code logic. The representation of a unit in the detailed design can use any means that are effective for understanding, such as UML diagrams, mathematical formulas, textual descriptions.

3.10.1.2 Detailing out software components

The software detailed design refines the components specified in the Software Architectural Design process into software units and their interfaces. These software units that are not further refined on the design level and their interfaces are the basis for generating or developing the source code for the derived software units.

The detailed design for a component describes the approach to satisfy the mapped software requirements by describing how software units will be organized both statically and dynamically. This includes describing how different software units will interact.

Since Automotive SPICE® 4.0 the SWE.3 BPs 1 and 2 explicitly focus on the software units' own intended technical or domain knowledge-oriented behavior:

- BP1: Specify the static aspects of the detailed design. For each software component specify the behavior of its software units, their static structure and relationships, their interfaces including…
- BP2: Specify the dynamic aspects of the detailed design. Specify and document the dynamic aspects of the detailed design with respect to the software architecture, including the interactions…

3.10.1.3 Views on software detailed design

Detailed design comprises at least a static and a dynamic view. There is no common definition which views are required nor criteria for the completeness of the sum of views. There are approaches in the industry that specify the kind of information that is required for the other ("viewpoints" which are collections of patterns, templates, and conventions for constructing a type of view) and the integration of the views in a complete detailed design description.

In most cases, the software detailed design mixes graphical representation with comprehensive textual explanations.

3.10.1.4 What a "software unit" is

In the first place, "software unit" is not an implementation-level term but a logical modeling-level term (see SWE.3.BP1). The logical modeling level represents the detailed software design, and a detailed software design is always a semantical abstraction from the source code but not identical with the source code itself. Also, the software architectural design and detailed design are created using the application domain language and entities. Consequently, the view of a software unit being an "inseparable coherent piece of

behavior” that is also “verifiable standalone” makes it an application domain knowledge perspective in the first place. This is independent of

- how many C functions will realize the software unit, and
- of the *.h and *.c files the software units finally “physically” represented.

Examples:

a. A motor driver *.c file with several C (sub-)functions transforming logical motor commands into IO signals (for the direction of rotation and for PWM / duty cycles for setting the motor speed) can be considered a software unit. A “motor driver” is an application domain entity name with exactly that coherent expected behavior.
b. A single C function implements a UML state machine (that is defined for a software Unit in the design model) by means of several switch-case statements (see Example 2 in subchapter 3.11.1.1 “The purpose of code coverage”). This single C function can be considered a software unit, given that it still includes the considered application domain behavior.

In both examples, at the code level these software units may of course be further divided up into many smaller C functions or even *.c files. In example (b.) specifically all behavior in a case-block might be factored into their own C subfunctions, which makes sense. However, doing so does not change the fact that – from the application domain knowledge perspective – the software unit is still the sum of all those subfunctions.

A software unit shall be verified against its specification provided in the detailed design (and not against the source code itself). Therefore, carrying such decomposition too far, however, may even introduce code review inefficiency when having to switch between many code files.

As a result, it can be concluded that a software unit can be either a single subroutine or several subroutines (e.g., a single C function but

also an entire *.c file containing several C functions) and that the decision of a software unit boundary must also be application domain-driven.

3.10.1.5 Code metrics vs. software unit boundaries

A further consequence of the fact that source code only represents an implementation solution for a software unit specification in the detailed design, code complexity metrics are not a driver for determining a software unit boundary.

3.10.1.6 Traceability and consistency

First, see also subchapter 3.3.1 of SYS.1 and 2.3.4 on traceability for clusters of information. Generally, it is necessary to understand which software requirement is, finally, represented in the detailed design. Reasons are, for example,

- comprehension of the logic of the software,
- efficient impact analysis in the context of changes,
- proof that there is no dead code, and
- all software requirements are covered.

For that purpose, during software requirements analysis the two following traceability options can be considered, depending on the actual content of the requirement:

Option A:

Traceability via software architecture.

Realizing a requirement that requires dynamic behavior and interactions between software components and, consequently, their software units.

Example:

Consider the software requirement:

“The SW shall write to the MOTOR-POWERING-IO the command START in 60[ms] with a tolerance of +/- 2[ms] when it has detected a

bus frame content in RECEIVING-IO and the SW is in state OPERATIONAL"

This requirement represents a coherent software service; however, many SW components and units will be involved. Traceability via the *static* design model (i.e., references between that requirement and each of the individual components/units) may result in an overwhelming number of links; furthermore, such a set of links to individual static elements is rather inconclusive with respect to judging requirement coverage and consistency. In contrast, looking at a sequence through the components/units allows judging whether the requirement really is completely and consistently satisfied (see Figure 3-2).

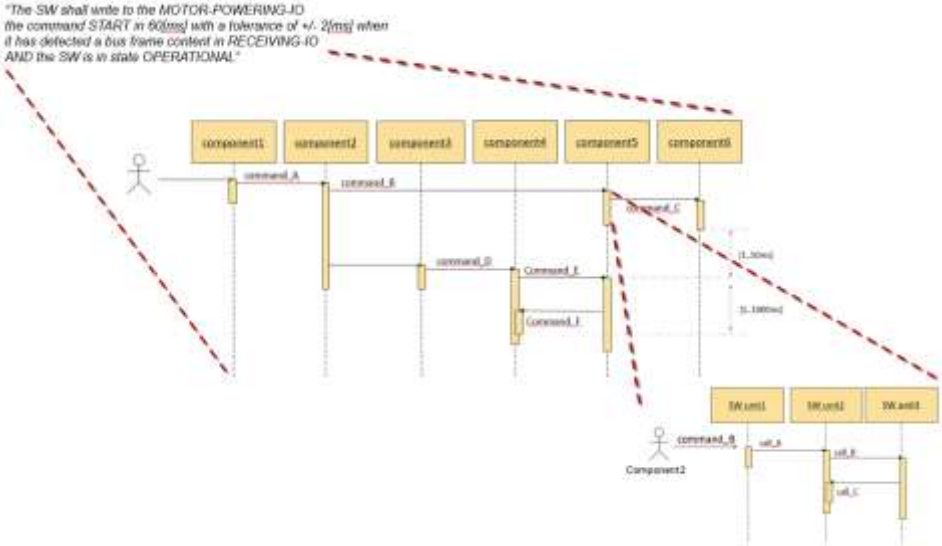


Figure 3-2: Traceability and consistency between SW requirements and SW components and SW units via dynamic interaction models

The advantage of this approach is even maximized if

- atomic individual requirements are grouped to represent coherent end-2-end system/software services, as in use cases,
- and then modeling a sequence of software-/system-internal behavioral flow behind the uses cases. This further minimizes the number of traceability links or references while maintaining consistency.

Option B:

Traceability between a particular software requirement and a software detailed design element.

Example: CAN matrix or basis software aspects, such as a list of diagnosis identifiers inherent in an Autosar configuration.

Software interface requirements may demand using a defined CAN matrix. Since there will be, for example, a particular set of software units decoding such messages, direct traceability may be intuitive.

Since Automotive SPICE® 4.0 SWE.3.BP only demands "*…consistency and … traceability between the **SW detailed design** and the SW requirements,*" but not to individual SW units in the detailed design or code directly. Therefore, creating dynamic models such as UML/SysML Sequence or Activity Diagrams and assigning it to the requirement does represent traceability to SW components and SW units (see Figure 3-2). Additionally, in practice the term "software unit" may sometimes erroneously be interpreted as the implementation of the software unit in source code. Nonetheless, this interpretation is not intended because the source code is the implementation solution of a software unit specified in the detailed design.

3.10.1.7 Strengthening of "SWE.3.BP3 Develop Software Units"

In SWE.3, BP3 emphasizes principles according to which the code will be developed, already reflecting such principles at coding time. In fact, there are coding principles that can be expected at CL1. Note 7 in SWE.3 suggests that such coding principles are, for instance, "no implicit type conversions," "one entry and one exit point

in subroutines," and "range checks" or general design-by-contract. Moreover, coding principles that can be considered at CL1 relate to the robust, error-free and technically correct behavior of the final software product. Consequently, in SWE.4 software unit static verification and code reviews, respectively, can then also verify whether those coding principles have been adhered to.

Examples for coding principles that can be expected at CL1:

- no implicit type conversions
  (to avoid value range under-/overflows)
- one entry and one exit point in subroutines
  (to avoid systematic faults)
- encapsulation at the code level as opposed to, e.g, global visibility of variables
  (to avoid systematic faults)
- defensive programming to mitigate systematic faults, e.g.,
  - range checks or general design-by-contract and
  - an 'enum' in C, the values of which are being initialized with a certain Hamming distance to increase robustness against memory corruption.

This supports the consideration of coding principles at an earlier point in time from a development lifecycle perspective.

Note that this strengthening is not to introduce redundancy, nor is it overlapping, with CL2. Other coding principles to be considered regarding GP 2.2.1 are ones that are not generally necessary because they depend on the specific assessed context. The following examples for coding principles depending on the product business strategy (e.g., platform development) could be expected at CL2:

- maintainability and comprehensibility by means of, e.g., naming conventions and commenting templates
- portability
- scalability
- reusability

This contrasts with a context about developing and maintaining a very customer-specific legacy product for only one particular application, whereby none of the above-mentioned principles would necessarily apply.

A further advantage of strengthening SWE.3.BP2 is that it should now receive a greater attention by assessors. Previously, during assessments this BP was often rated as F based on the mere existence of code.

3.10.2 Rating rules within the process

3.10.2.1 Boundary of a software unit

Rating rules:

> **[SWE.3.RL.1]** If a software unit in the detailed design is represented in the code by a cluster of programming language (sub-)routines but not by single atomic (sub-)routines, then SWE.3.BP1 and SWE.3.BP3 shall not be downrated.

3.10.2.2 Code metrics vs. programming language routine boundaries

Another consequence of the above is that code complexity metric results are not always a valid reason to determine the size and structure of programming language (sub-)routines that implement the software unit. Particularly, considering one single code metric alone for such a purpose should be avoided.

Example: complexity according to McCabe[1]

> Original assumption: a value of $M_{McCabe} > 10$ is less comprehensible for programmers, testers, and reviewers, and has a higher likelihood of systematic faults. However, this

[1] Arthur H. Watson, Thomas J. McCabe, “Structured Testing: A Testing Methodology Using the Cyclomatic Complexity Metric”, NIST Special Publication No. 500-235, 1996

assumption cannot be considered generally true: code sample (a), below, reveals M_{McCabe} = 14 while code (b) reveals M_{McCabe} = 3. However, code (a) is more comprehensible and therefore maintainable than code (b). Moreover, it is not obvious why code sample (a) should be divided up further into smaller code pieces for the sole purpose of reducing M_{McCabe}.

(a) M_{McCabe} = 14

```
const String getMonthName (const int number)
{
  String result;

  switch (number)
  {

    case 1:  result = "Jan";    break;
    case 2:  result = "Feb";    break;
    …

    case 12: result = "Dec";    break;
    default: result = "unknown month";

   }

  return result;

}
```

(b) M_{McCabe} = 3

```
const String getMonthName (const int number)
{
  string[] months = new string[]
  {
    "Jan",
    "Feb",
    …
    "Dec"
  };

char result[] = "unknown month";
if ((number >= 1) && (number <= sizeof (months)))
  {
    result months[number - 1];
  }
  return result;
}
```

Therefore, a combination of selected code metrics will provide meaningful hints on where refactoring should – or not – be discussed to:

- Achieve “clean”, comprehensible, and maintainable code
- Decide on the necessity of particular software unit verification measures (see SWE.4)

This also means that if such a code metric combination exceeds a defined combined target but there is justifiably still no didactical or conceptual advantage in connection with the application domain knowledge and terminology, then the software unit boundary should not be redefined. Note, however, that this shall not be an excuse for unstructured code fragments.

Rating rules:

> **[SWE.3.RL.2]** If code metric targets for a programming language (sub-)routine are formally violated but there are reasonable context-specific arguments why the size and boundary of that routine are acceptable, then SWE.3.BP1 shall not be downrated.

3.10.2.3 Dynamic behavior

For the description of the internal behavior of the software units, graphical representations (e.g., UML) and/or textual explanations abstracting from the implemented source code are to be used.

Rating rules:

> **[SWE.3.RL.3]** If a software unit is of such a low complexity from the technical application domain knowledge perspective that its dynamic behavior can be adequately described in text without use of a graphical notation, and only such a textual explanation is available, then SWE.3.BP2 shall not be downrated.

3.10.3 Rating rules with other processes at level 1

None.

3.11 SWE.4 Software Unit Verification

The purpose is to verify that software units are consistent with the software detailed design.

3.11.1 General information

Software unit verification covers not only software unit testing aspects but also unit verification aspects such as static verification of units.

3.11.1.1 The purpose of code coverage

As is clear from SWE.3 and SWE.4, the implementation of a software unit in the source code level shall be verified against the unit specification in the detailed design. A software unit's source code level shall not be verified against the code itself as this does not prove if the unit works correctly according to the application domain logic. This would just prove that the code works as programmed, or that the compiler works correctly.

A recurring question is whether during SWE.4 a 100% code coverage (e.g., statement coverage, branch coverage, MC/DC coverage) of the unit shall be achieved.

Answer:

Generally, the purpose is not to achieve a 100% coverage of all software unit code as a verification objective on its own. The purpose rather is to check if a particular test case did touch exactly those parts in the software unit code it was supposed to, based on the test case purpose and definition. In other words, code coverage represents accompanying information that helps judging completeness of the selected test cases. This means that code coverage alone is not a verification objective. See also ISO 26262-6:2018 *[ISO26262],* clause 9.4.4 here.

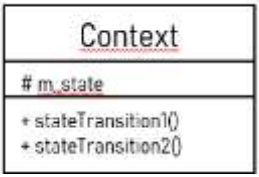
In Example 1 below, when testing `stateTransition1()` with the aim of checking whether the state change is performed entirely and correctly, a code coverage of 100% is expected. The reason is that

in Example 1 each single state transition is represented by its own C++ method.

In Example 2 below, however, when testing `stateChange()` with the same goal of checking whether the state change for `MY_EVENT1` is performed entirely and correctly, a code coverage of <100% is expected. The reason is there is only one single C++ method including all state changes represented by the various switch-case branches; therefore, the code relevant for `MY_EVENT1` is only a subset of the code.

This is one of the reasons why SWE.4.BP2 requires a selection of unit test cases, supported by SWE.3.BP1 Note 1 stating that "*a software unit in the detailed design may be, at the code level, represented by a single subroutine* (e.g., Example 2) *or a set of subroutines* (e.g., Example 1)".

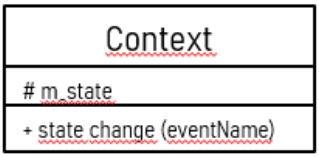
Example 1: Possible state machine implementation for a class




```
void stateTransition1()
{
    If(STATE_A == m_state)
    {
    exit_CURRENT_STATE();
    state = NEXT_STATE;
    entry_NEXT_STATE();
    do_NEXT_STATE();
    }
}
```

```
void stateTransition2()
{
    If(STATE_B == m_state)
    {
    exit_CURRENT_STATE();
    state = NEXT_STATE;
    entry_NEXT_STATE();
    do_NEXT_STATE();
     }
}
```

Example 2: Alternative state machine implementation




```
void state_change (possibleEventsEnum event)  {
switch (m_state)
{
    case <stateName1>:
     if (MY_EVENT1 == event)
     {
        exit_ <stateName1>();
        m_state = <stateName4>;
        entry_ <stateName4>();
        do_ <stateName4>();
     }
    break;
    case <stateName2>:
     if (MY_EVENT2 == event)
     {
        exit_ <stateName2>();
        m_state = <stateName6>;
        entry_ <stateName6>();
        do_ <stateName6>();
     }
    break;
    default: // invalid state
}
```

3.11.2 Rating rules within the process

3.11.2.1 Define software unit verification measures

Rating rules:

> **[SWE.4.RL.1]** If it is made reasonably plausible based on context-specific arguments that a particular verification measure is not necessary, then SWE.4.BP1 shall not be downrated.

3.11.2.2 Automation of verification measures

Rating rules:

> **[SWE.4.RL.2]** If a verification measure is automated and the correctness, completeness, and consistency of the corresponding scripts and programs are not addressed in the verification measure definition, then SWE.4.BP1 shall be downrated.

3.11.2.3 Exploratory testing vs. traceability/consistency

The state-of-the-art testing not only comprises testing derived from requirements, but also exploratory testing based on experience, such as “error guessing based on knowledge.” This is valuable as it adds to the quality of the product. Therefore, exploratory tests that are based on experience cannot, by definition, be traced or consistent with the software requirements.

Still, traceability is needed between exploratory test cases and their results.

Rating rules:

> **[SWE.4.RL.3]** If verification measures representing exploratory tests, which by definition cannot be traced to the detailed design or have no such traceability, then SWE.4.BP4 shall not be downrated.

3.11.2.4 Traceability and consistency

See also subchapter 3.3.1 of SYS.1 and 2.3.4 here.

3.11.3 Rating rules with other processes at level 1

3.11.3.1 Verification measures selection vs. release plans

Verification measure selection can technically be done properly based on technical expertise with respect to the software product and design and based on experience. This may be the case even if the project's release content is not well-defined.

Rating rules:

> **[SWE.4.RL.4]** If selection of verification measures is properly done but based on an inadequately or incompletely defined release content, then SWE.4.BP2 shall not be downrated.

3.12 SWE.5 Software Component Verification and Integration Verification

The purpose is to verify that software components are consistent with the software architectural design, and to integrate software elements and verify that the integrated software elements are consistent with the software architecture and software detailed design.

3.12.1 General information

3.12.1.1 The scope of SWE.5

For understanding the concepts of software unit integration and the standalone verification of software components, see subchapter 2.4.

The term “integrated software” as used in the context SWE.5 refers to the sole technical software product, or sample, on which verification is performed. This term alone therefore

- does not address documentation,
- nor imply that SWE.4 must have been done prior to SWE.5 as a PAM does not represent a lifecycle model.

3.12.2 Rating rules within the process

3.12.2.1 Verification measure definition

Rating rules:

> **[SWE.5.RL.1]** If entry/exit criteria are reasonably specified for a set of verification measures, SWE.5.BP1 and SWE.5.BP2 shall not be downrated.

3.12.2.2 Automation of verification measures

Rating rules:

> **[SWE.5.RL.2]** If a verification measure is automated and the correctness, completeness, and consistency of the corresponding scripts and programs are not addressed in the

verification measure definition, then SWE.5.BP1 or SWE.5.BP2, respectively, shall be downrated.

3.12.2.3 Exploratory testing vs. traceability/consistency

The state-of-the-art testing not only comprises testing derived from requirements but also exploratory testing based on experience, such as “error guessing based on knowledge.” This is valuable, as it adds to the quality of the product. Therefore, exploratory tests that are based on experience cannot, by definition, be traced or consistent with the software requirements.

Still, traceability is needed between exploratory test cases and their results.

Rating rules:

> **[SWE.5.RL.3]** If verification measures representing exploratory tests, which, by definition, cannot be traced to the architectural design and have no such traceability, then SWE.5.BP6 shall not be downrated.

3.12.2.4 Traceability and consistency

See also subchapter 3.3.1 of SYS.1 and 2.3.4 here.

3.12.3 Rating rules with other processes at level 1

3.12.3.1 Verification measures selection vs. release plans

Rating rules:

> **[SWE.5.RL.4]** If selection of verification measures is properly done but based on an inadequate or incomplete release plan, then SWE.5.BP3 shall not be downrated.

3.13 SWE.6 Software Verification

The purpose is to ensure that the integrated software is verified to provide evidence for compliance with the software requirements using verification measures consistent with the software requirements.

3.13.1 General information

The aim of software verification is to confirm that the integrated software is consistent with the software requirements, which means taking a black-box view of the software. The object-under-verification is the integrated software, not the verification environment. This implies that any verification environment can be applicable.

3.13.2 Rating rules within the process

3.13.2.1 Verification measure definition

Rating rules:

> **[SWE.6.RL.1]** If entry/exit criteria are reasonably specified for a set of verification measures instead of each verification measure, then SWE.6.BP1 shall not be downrated.

3.13.2.2 Automation of verification measures

Rating rules:

> **[SWE.6.RL.2]** If a verification measure is automated and the correctness, completeness, and consistency of the corresponding scripts and programs are not addressed in the verification measure definition, then SWE.6.BP1 shall be downrated.

3.13.2.3 Exploratory verification vs. traceability/consistency

The state-of-the-art testing not only comprises testing derived from requirements but also exploratory testing based on experience, such as “error guessing based on knowledge.” This is valuable as it adds to the quality of the product. Therefore, exploratory tests that are

based on experience cannot, by definition, be traced or consistent with the software requirements.

Still, traceability is needed between exploratory test cases and their results.

Rating rules:

> **[SWE.6.RL.3]** If verification measures representing exploratory tests, which, by definition, cannot be traced to the software requirements, have no such traceability, then SWE.6.BP4 shall not be downrated.

3.13.2.4 Traceability and consistency

See also subchapter 3.3.1 of SYS.1 and 2.3.4 here.

3.13.3 Rating rules with other processes at level 1

3.13.3.1 Verification measures selection vs. release plans

Rating rules:

> **[SWE.6.RL.4]** If selection of verification measures is properly done but based on an inadequate or incomplete release plan, then SWE.6.BP2 shall not be downrated.

3.14 VAL.1 Validation

The purpose is to provide evidence that the end product, allowing direct end user interaction, satisfies the intended use expectations in its operational target environment.

3.14.1 General information

3.14.1.1 Motivation behind the process purpose

The process VAL.1 Validation centers around “intended use,” thereby addressing the product’s end users. It therefore excludes looking at pure embedded software products, an ECU, or a drive (comprising a motor and an ECU) – none of which provides a direct end user interface.

In the absence of legal requirements (e.g., a maximum closing force of 100N for window regulators, or homologation requirements), the target expectations behind validation may be of an exploratory, or even subjective, nature.

Example 1: Automatic transmission as a mechatronic system

> Meeting the defined gear-shifting time constraints is considered “verification,” given that these can be measured objectively. In contrast, providing an adequate gear-shifting feeling is a “validation” concern, as it requires feedback from end users or their representatives.

Example 2: Automatic side door access systems

> There are no legal closing force requirements. Therefore, how much closing force represents an intolerable user harm considering the concrete inertia, kinematics, spring rates, and thickness of rubber seals, etc. is a matter of “validation”, such as by means of accident simulations. In contrast, the angle at which the automatic door movement support is to be triggered is a matter of decision which can be objectively measured, thus representing “verification.”

Note that the possibility of being able to write up a requirement in the first place does not serve as a distinction criterion for differentiating between verification and validation. This is to say, it is not possible to argue it is about verification whenever one is able to specify a requirement. Related to the two examples above, a requirement could still be written about:

1) defining certain max. acoustics and vibration to express a gear-shifting “feeling,”
2) or a maximum closing force, respectively.

Still, determining whether these requirements are “adequate” would be a matter of validation because they must be approximated. This is because of limitations, and the nature, of requirements engineering in terms of dealing with potentially unidentified needs, or identification of appropriate requirements only in an iterative manner.

3.14.2 Rating rules within the process

3.14.2.1 Validation measure definition

Rating rules:

> **[VAL.1.RL.1]** If entry criteria are reasonably specified for a set of validation measures instead of each individual validation measure, then VAL.1.BP1 shall not be downrated.

3.14.2.2 Exploratory validation vs. traceability/consistency

Exploratory validation measures that are based on experience cannot, by definition, be traced or consistent with stakeholder requirements.

Still, traceability is needed between exploratory test cases and their results.

Rating rules:

> **[VAL.4.RL.2]** If validation measures representing exploratory tests, which, by definition, cannot be traced to requirements,

have no such traceability, then VAL.1.BP4 shall not be downrated.

3.14.2.3 Traceability and consistency

See also subchapter 3.3.1 of SYS.1 and 2.3.4 here.

3.14.3 Rating rules with other processes at level 1

3.14.3.1 Validation measures selection vs. release plans

Rating rules:

[VAL.1.RL.3] If selection of validation measures is properly done but based on an inadequate or incomplete release plan, then VAL.1.BP2 shall not be downrated.

3.15 MLE.1 Machine Learning Requirements Analysis

The purpose is to refine the machine learning-related software requirements into a set of ML requirements.

3.15.1 General information

The Machine Learning Requirements Analysis process uses the software requirements that were processed in the Software Requirements Analysis process and the elements of the software architecture as an input.

Results of this analysis are specified functional and non-functional Machine Learning requirements (ML requirements) including Machine Learning data requirements (ML data requirements).

3.15.2 Rating rules within the process

Since the ML requirements belong to the group of software requirements, the rating recommendations from SWE.1 Software Requirements Analysis process should be considered here (see 3.8).

ML requirements are derived from the software requirements categorized to be implemented in an ML-based software element. ML requirements consist of ML data requirements that are the main input for the SUP.11 Machine Learning Data Management and other ML requirements which are input for the other MLE processes.

ML data requirements shall address:

- data characteristics to be covered and their expected properties (e.g., distributions, accuracies, resolution, etc.)
- non-functional requirements (e.g., regarding labeling quality, integrity of data)
- structure, format and origin of ML data

Other ML requirements should address:

- functional parts to be implemented for training and testing the ML model
- hardware-related ML functions
- receiving signals from electronic sensors
- non-functional requirements for training and deployment (e.g., performance, quality requirements)

ML requirements must be granular, understandable, and verifiable. Unclear or generic requirements have to be clarified with the system or software requirement owner.

Rating rules:

> **[MLE.1.RL.1]** If data characteristics or non-functional requirements of the ML data requirements are not addressed, then MLE.1.BP1 shall not be rated higher than P.

3.16 MLE.2 Machine Learning Architecture

The purpose is to establish an ML architecture supporting training and deployment, consistent with the ML requirements, and to evaluate the ML architecture against defined criteria.

3.16.1 General information

The goal of this process is to establish an ML architecture. The ML architecture must be designed in response to the problem that the ML model is intended to address. ML models are very good at identifying patterns, but some are better suited for specific problems than others. As an example, often convolutional neural networks are used for object detection.

The ML architecture must contain all necessary ML architectural elements like hyperparameter ranges and initial values, details of the ML model, and possible other software parts which are necessary for MLE.3 Machine Learning Training.

For the ML architecture, the resource consumption objectives are required to be derived from ML requirements for all resource-critical elements and may differ between the trained ML model and the deployed ML model.

The training is often done in a specific training environment defined in the ML training and validation approach (see MLE.3). Also, for this environment resource consumption objectives should be defined to ensure feasibility of the ML architecture.

3.16.2 Rating rules within the process

The ML architecture must consider not only the ML model itself but also any potential additional software required to train, deploy, and test the ML model.

Typical examples of necessary ML architectural elements are pre- and postprocessing components like data augmentation and ground truth evaluation. It should also be considered that some of these ML architectural elements are required for training but will not be available once the ML model is deployed. Such (classical) software components should be developed according to SWE.3 Software Detailed Design and Unit Construction and SWE.4 Software Unit Verification. Evaluation of these ML architectural elements (e.g., pre- and postprocessing) should be documented.

Often different ML models are considered and trained. Hyperparameters like learning rate, loss function, model depth, regularization constants will allow the configuration of the ML model. The rationale for different hyperparameters and initial values should be provided and the decisions taken documented.

The ML architecture also has to consider the interfaces between the different ML architectural elements. Typically, interfaces are documented in terms of name, type, range, default value, unit, resolution, and direction.

Rating rules:

> **[MLE.2.RL.1]** If the ML architecture does not consider elements necessary to train, deploy, and test the ML model, then MLE.2.BP1 shall not be rated higher than P.

3.17 MLE.3 Machine Learning Training

The purpose is to optimize the ML model to meet the defined ML requirements.

3.17.1 General information

Machine Learning uses an ML model capable of performing a functional mapping of an input to an output tensor of data. The quality of the mapping is optimized by adjusting internal parameters of the ML model until the deviation of output tensors from expected values is better than a predefined threshold. Usually, these parameters are the weights of a weighted sum or average as input to the activation function of a neuron and the hyperparameters as defined by the ML architecture (see MLE.2).

Even simple tasks lead quickly to a high-dimensional optimization problem, because the number of weights depends on the sizes of input and output tensors, the number of layers, and other aspects. Therefore, the training process of a ML model consumes high amounts of memory and computing power and even more if floating point operations are needed.

ML validation as part of the training process supports the optimization of the hyperparameters during MLE.3 Machine Learning Training. The term “validation” has a different meaning than VAL.1.

Due to the complexity of the task, the training process is usually an iterative process which can require changes of the ML architecture (see MLE.2), the ML training and validation approach, or the ML training and validation data set. ML training might present certain challenges, including overfitting, underfitting, or biases in the data, which could impact the performance of the model. Even with experience, it cannot be ensured from the beginning that a defined ML architecture achieves the required quality immediately with the first training. Therefore, iterative changes are not an indication of failure with MLE.2 or MLE.3 but an inherent part of the process to establish an ML model which eventually satisfies all ML requirements.

The data set for ML training and validation must be created from the ML data collection provided by SUP.11 according to the ML training and validation approach. Deviating leads to training results that are not ensured to meet the ML requirements.

3.17.2 Rating rules within the process

Machine learning training requires already for achievement of PA 1.1 a careful preparation of the training environment. Especially the required HW resources and optimization approaches need to be defined to achieve the targeted optimum in a reasonable time while preventing problems like overfitting.

The data set for ML training and validation of the achieved capability of the ML model in the training cycle must be carefully selected based on predefined criteria to support the training goal and prevent common problems (e.g., bias). Be aware that a separate data set for ML training and validation is not necessarily required at the start of training for some ML validation approaches (e.g., k-fold cross validation). If validation is required for the training process, dedicated validation data must be available.

Rating rules:

> **[MLE.3.RL.1]** If the ML training and validation approach uses validation techniques that do not require separated ML training and validation data sets at the start of ML training, then MLE.3.BP1 shall not be downrated.

3.18 MLE.4 Machine Learning Model Testing

The purpose is to ensure compliance of the trained ML model and the deployed ML model with the ML requirements.

3.18.1 General information

The Machine Learning Model Testing process focuses on testing the agreed upon trained ML model to ensure compliance with the ML requirements. Therefore, an ML test approach is specified and an ML test dataset is created from the ML data collection provided by SUP.11 based on ML data requirements. After successfully testing the trained ML model, a deployed ML model is derived and tested as well.

The deployed ML model will be integrated into the target system and may differ from the trained ML model, which often requires powerful hardware and uses interpretative languages.

Testing an ML model is done by comparing results of test data computed using the trained or deployed ML model with expected results and non-functional ML requirements (e.g., KPIs) with defined pass/fail criteria specified in the ML test approach.

ML test results supplying a meaningful summary of the computed results for the used test data are required evidence for test execution.

The test data set has to be created from the ML data collection provided by SUP.11 according to the ML test approach.

The ML test data set shall be used for final testing of the trained ML model and the deployed ML model and must not be used for training. This means that no major changes/optimization are performed based on the ML test data set. Because with every optimization some information over the data set leaks into the model quickly resulting in overfitting to the used data set.

If the test fails and optimization of the ML model is needed, it must be verified that the ML test data set is still reliable to ensure

compliance with the ML requirements; hence, a change of the ML test data set might be needed.

3.18.2 Rating rules within the process

3.18.2.1 ML test approach

In general, all ML testing activities should be in line with the ML test approach.

Rating rules:

> **[MLE.4.RL.1]** If the ML test data set is used to perform major changes/optimization of the ML model, then MLE.4.BP1 shall not be rated higher than P.

3.19 HWE.1 Hardware Requirements Analysis

The purpose is to establish a structured and analyzed set of hardware requirements consistent with the system requirements and the system architectural design.

3.19.1 General information

3.19.1.1 Scope of the HWE processes

See subchapter 2.3.1.

3.19.1.2 Iterative vs. incremental development

Normally the functional content of the product changes iteratively and evolves incrementally across releases. The term “increment” can be understood as adding a feature or element that did not exist before (analogy: building a house). The term “iteration” can be understood as refining, or adapting, an existing feature or element (analogy: a sculptor working on a sculpture).

Therefore, the complete set of requirements of the end product does not necessarily have to be available at the project start. Rather, release scopes agreed with the customer will define increments and iterative rework. In this respect, requirements creation can be driven by release definitions over time.

3.19.2 Rating rules within the process

3.19.2.1 Hardware development without system requirements

In case of hardware development only, the hardware requirements may refer directly to the stakeholder requirements. Consequently, consistency and bidirectional traceability must be ensured between stakeholder requirements and hardware requirements.

Rating rules:

> **[HWE.1.RL.1]** In the case of hardware development only, if the traceability and consistency from hardware requirements to stakeholder requirements is established then HWE.1.BP5 shall not be downrated.

[HWE.1.RL.2] If hardware requirements are not derived from system requirements but from platform requirements, then HWE.1.BP1 shall not be downrated.

[HWE.1.RL.3] If hardware requirements are not derived from system requirements but from stakeholder requirements that do not affect system requirements or architecture and this is agreed with hardware and system representatives, then HWE.1.BP1 shall not be downrated.

3.19.2.2 Structuring of requirements

Hardware requirements can be grouped or categorized to support an overview and prioritization. See also subchapter 2.3.3.2 here.

Rating rules:

[HWE.1.RL.4] If “functional” and “non-functional” are the only requirements categorization or classification criterion, then HWE.1.BP2 shall be rated as N.

3.19.2.3 Requirements mapping to releases

A possible approach to prioritizing requirements is the allocation of requirements to releases. The usage of such an approach will imply that the content of the next and future releases is defined.

Rating rules:

[HWE.1.RL.5] If there is no direct evidence of prioritization of the hardware requirements, but a separate release plan is consistently mapping these hardware requirements to the future releases, then HWE.1.BP2 shall not be downrated.

[HWE.1.RL.6] If the hardware requirements that are mapped to a particular release do not match with the system requirements mapped to the same release, then HWE.1.BP2 shall be downrated.

3.19.2.4 Analysis of requirements

The indicator HWE.1.BP3 requires “*Analyzing hardware requirements. …and to support project management regarding project estimates.”* This means, for example:

- A set of 100 requirements exists. An analysis was done together with the project manager during a project progress meeting. As a result, 20 out of the 100 requirements were decided not to be used, therefore being attributed as “rejected” with an accompanying comment providing expectations.

- A set of 10 requirements were planned for the next release. The development team reports to the project manager that this is no longer feasible due to resource constraints. The decision is not to change the status of those 10 requirements but to reallocate them to future releases. This can be evidenced by a comparison of the release plans (which is the process context of MAN.3 but not HWE.1).

Analysis of requirements can be done by means of using, for example, tool-based attributes, or comments added to the requirements text.

The analysis of hardware requirements is the basis for a correct implementation. Even though requirements sometimes appear very simple, a well-founded analysis has to be conducted for those requirements. The scope and appropriateness of the analysis depends on the context of the product (e.g., platform). The results of analysis can vary from a simple attribute to a complex simulation or the building of a demonstrator to evaluate the feasibility of hardware requirements.

The reason for having associated this BP with the supporting of project management regarding project estimates is the following: revising hardware requirements by means of an analysis may (re-)define the scope of work. Once this solution is settled, HWE.4 will select verification measures for addressing exactly that problem space. In other words, HWE.4 itself does not change that problem space, therefore does not alter the overall project’s scope of work.

Rating rules:

[HWE.1.RL.7] If analysis results of requirements are not demonstrated by means of separate analysis reports or review

records but by means of, e.g., tool-supported attributes or tool-supported commenting, then HWE.1.BP3 shall not be downrated.

[HWE.1.RL.9] If the analysis of hardware requirements with respect to technical feasibility is covered by risk management, then HWE.1.BP3 shall not be downrated.

[HWE.1.RL.10] If the analysis results of hardware requirements regarding impact on estimates are communicated but not consistently used by project management, then HWE.1.BP3 shall not be downrated.

3.19.2.5 Traceability and consistency

HWE.1.BP5 offers the possibility of having two paths for traceability:

a) between hardware requirements and system architectural design (SYS.3)
b) between hardware requirements and system requirements (SYS.2)

However, redundancy, namely using the two traceability paths for the very same hardware requirement at the same time, is neither intended by this BP nor meaningful. Furthermore, it is not intended to express that all or most of the hardware requirements should be traced to system requirements directly as a default. Which path appears more appropriate must depend on the actual content of the hardware requirement itself.

See also subchapter 3.3.1 of SYS.1 and 2.3.4 here.

3.19.3 Rating rules with other processes at level 1

None.

3.20 HWE.2 Hardware Design

The purpose is to provide an analyzed design, that is suitable for manufacturing, and to derive production-relevant data.

3.20.1 General information

3.20.1.1 Scope of the HWE processes

See subchapter 2.3.1.

3.20.1.2 Why no extra processes for HW architectural design and HW detailed design?

In hardware engineering practice:

- HW architectural design begins at the block diagram level, being the starting point for the detailed design. Detailed hardware design is the level of information from which physical HW instances can be created, i.e., initial block diagrams do not reveal that level of detail.

- The entire HW design process is performed iteratively. Technical details that originate from lower design levels like schematics or layout (detailed design) might be added to block diagram models (architectural design) to provide further information for distinct verification and testing that are aimed to be done at the architectural level.

Also, note that the following assumptions would not serve as a motivation for separating HW architectural and detailed design at the level of a PRM:

1) "In their development processes companies may have extra activities for architectural and detailed design, mostly done iteratively with HW detailed design."

- A PRM/PAM does not represent a lifecycle model (see Automotive SPICE® 4.0 subchapter 3.3.4)
- A PRM/PAM is at the process-WHAT-level, while processes in companies are at the process-HOW-level. Therefore, it is the

assessor's responsibility to map Assessment Indicators in a PAM to the assessed context (see Automotive SPICE® 4.0 subchapter 3.3.3)

2) "Two processes would provide a better overview, meaning a more orderly partitioning of topics."

- A PRM/PAM – by definition – does not represent a lifecycle model. Therefore, it is the assessor's responsibility to map Assessment Indicators in a PAM to information presented by projects and organizational units (see Automotive SPICE® 4.0 subchapter 3.3)

Also note that this HWE PRM/PAM does not represent an ECU level (see subchapter 2.3.2).

For these reasons, at the level of a PRM, there is no necessity to separate HW architectural design and HW detailed design into two processes. The BPs needed to assess architectural and detailed design remain within HWE.2.

This is also consistent with the following models:

- ISO 26262-5:2018
- Swedish Standard SS 7740:2018[2]
- PISA[3]

---

[2] The choice of the Swedish Standard SS 7740:2018 (being a PRM/PAM aiming for integrating elements from Automotive SPICE® PRM v 4.5 and PAM v2.5, and particular process-related clauses in ISO 26262:2011 1st Ed) was to also have a single hardware design process only (SE.ENG.5). This single process comprises BPs for both hardware architectural design and hardware detailed design.

[3] In the PISA model (Process Improvement Scheme for Automotive, as proposed by the System & Software Evaluation Centre, National Research Council of Italy), the "hardware segment" consists of four processes. Only one of them "…*pertains to the definition of electronics design, including the preparation of the physical layout*", namely HW1. There is no separation into HW architectural design and hardware detailed design at the process level; a distinction between HW architectural and detailed design is internal to HWE.1.

- AIDA[4]

3.20.1.3 Analysis of the hardware design (BP4) vs. HWE.3

A note in HWE.2.BP4 explains that techniques for analyzing the hardware design can be, for instance, simulations, calculations, or analytical approaches. These techniques are also admissible for verification measures in the context of HWE.3. This might raise the questions whether HWE.2.BP4 and HWE.3 are somewhat redundant, and if the process purposes HWE.2 and HWE.3 are overlapping. However, this is not the case:

- HWE.3 verifies if a given physical (production data compliant hardware sample reveals the characteristics defined by the hardware design. This means, the “object-under-verification” is the physical hardware sample.

- In contrast HWE.2.BP4 determines whether the hardware design itself, i.e., documented information such as schematic and layout etc., is appropriate. This means, the “object-under-analysis” is not a physical hardware sample but documentation representing the design decisions.

The reason for having associated this BP with the supporting of project management regarding project estimates is the following: revising hardware design solutions because of a design analysis may (re-)define the scope of work. Once this solution is settled, HWE.3 will select verification measures for addressing exactly that solution space. In other words, HWE.3 itself does not change that solution space and thus does not alter the overall project’s scope of work.

[4] Similarly to the Swedish standard SS7740, the Italian AIDA model explains itself both as a reference for reaching compliance with ISO 26262:2018 and as a PAM for processes assessment. It defines a single PRM process “hardware design,” i.e. no separation of a HW architectural and detailed design; the process “hardware architectural metrics” only covers the ISO 26262 *[ISO26262]* clauses on HW architectural metrics and evaluation of safety goal violations due to random hardware failures.

3.20.1.4 ISO 26262 “Evaluation of HW elements” is not an alternative for HWE.3 and HWE.4

The processes HWE.3 and HWE.4

- do not take a single HW element perspective. This is because the term ‘hardware element’ can denote a HW part, a HW component, or the complete hardware (see glossary).

- They are not restricted to safety-related products or contexts, so for hardware development the HWE processes represent what ISO 26262 calls “evidence of compliance with standards that support quality management” (ISO 26262-8:2018 *[ISO26262],* clause 5.3.2, example 2).

Clause 13 in ISO 26262-8:2018 *[ISO26262]* addresses how to proceed with a procured individual HW element that is supposed to be used in a safety-related product. Therefore, hardware part evaluation according to ISO 26262-8:2018 *[ISO26262],* clause 13 is complementary to HWE.3 and HWE.4 and does not contradict them.

3.20.1.5 Communicate agreed hardware architecture and hardware detailed design

Apart from verification personnel, further important stakeholders can be manufacturing. Integrating this party in the information flow supports ensuring that Special Characteristics and relevant production data are properly verified and controlled in production.

3.20.2 Rating rules within the process

3.20.2.1 Traceability and consistency

See also subchapter 3.3.1 of SYS.1 and 2.3.4 here.

3.20.3 Rating rules with other processes at level 1

None.

3.21 HWE.3 Verification against Hardware Design

The purpose is to ensure that the production data compliant hardware is verified to provide evidence for its compliance with the hardware design.

3.21.1 General information

3.21.1.1 Scope of the HWE processes

See subchapter 2.3.1.

3.21.1.2 General explanation

Integration in terms of software lifecycle processes is understood as a stepwise assembly of a product, and performing tests alongside or in between the assembly steps. This notion is not always applicable per se to hardware development. Rather, a HW often is fully assembled first, and then HW testing is performed on the fully assembled hardware by, for instance, using measuring points inside the HW to test the inputs and outputs with variations.

Furthermore, testing of a single HW element always includes the testing of the interfaces as such tests need electrical input signals and output load. This implies that “testing a single HW element in isolation” inherently includes “testing the interfaces between HW elements,” as they are not conceptually distinct elements.

The notion of “reusing HW components” might be understood as integrating a physical – and already verified – HW component. However, reusing a HW component is not a “physical activity” in the sense that a HW component is “taken off the shelf and soldered onto a PCB.” Instead, this refers to reusing parts of HW design drawings or models during the creation of the HW design.

Examples:

- A voltage measurement solution is taken from an earlier schematic design, or from a model database, and placed into another schematic.
- A model library contains components for re-use in the chip design.

This will also require verification of the “re-used” HW component as the influences of the rest of the (new) surrounding hardware must be considered.

Thus, in order to avoid confusion and speculation the term “integration” is not used in the context of HWE.3.

The processes HWE.3 and HWE.4 can be mapped to ISO 26262-5:2018 *[ISO26262],* clause 10.

3.21.1.3 Rationale for BP “Ensure use of compliant samples”

What would be the consequences if HWE.3 did not have this base practice?

- Upon not-OK verification results one would not know whether this is due to design flaws or production (or sample construction workshops, as applicable) errors. The latter is not in the scope of the HWE PRM/PAM (see subchapter 2.1.1). SPICE models remain PRMs/PAMs for development.

- It would be economically disadvantageous to spend effort on HWE.3 just find out later that this was wasted because the verification was performed on an incorrect sample in the first place. (Note that this is not an argument at the abstraction level of a PRM/PRM, however still a reasonable one.)

In reality:

- Sample construction workshops (German: “Musterbau”) sometimes deliver samples that are not compliant (e.g., the exact soldering paste might not have been available, or because of the manual activities).

- Electronics production generally has varying manufacturing quality, or even deviations, even in the presence of state-of-the-art production quality plans, production traceability, etc.

For these reasons, it is necessary to include an "interfacing base practice" within HWE.3 to ensure that the delivered sample matches the specifications of the ordered sample (means: hardware production data compliant). To satisfy HWE.3.BP3, one out of many

possibilities certainly is, for example, EOL testing. However, as mentioned above, such processes or aspects are not in the scope of HWE PRM/PAM.

Comparison with configuration management:

This BP could be viewed as some sort of “HW baseline integrity check” and therefore be replaced by an editorial pointer to the Automotive SPICE® process SUP.8. However, SUP.8 encompasses more than just a single baseline audit base practice. In addition, the definition of a “HW baseline” (which is another base practice in SUP.8) actually happens in the context of HWE.2. For better usability and intuitiveness of the HWE PRM/PAM, the respective BPs in HWE.3 and HWE.4 are introduced, and kept, instead of pointers to SUP.8.

3.21.1.4 The meaning of stepwise hardware integration testing

There is no conceptual distinction between “testing a single HW element” and “testing interfaces between HW elements” as explained above. Therefore, examples on how to understand the suggestion of “stepwise verification” as stated in HWE.3.BP1 Note 2 are

- additional measuring points used only for the purpose of checking internal signals (e.g., for boundary scan tests, contacts for needle bed adapter in production test);

- power supply module testing:
  - connecting a power supply (with current limiting) to the PCB
  - measuring the voltage at the power supply test points on the PCB to ensure that the correct voltage levels are being supplied.
  - checking for any abnormal heating of components on the power supply circuit.

3.21.1.5 Verification measures do not require using identical hardware samples

The need for physical hardware samples depends on the actual content of the verification measure. There are verification measures

that do not require – or cannot even be performed on – physical hardware samples, such as calculations, simulations, reviews, and analyses; still, simulation models can be improved based on measurements with real physical samples.

This is why HWE.3.BP2 “Ensure use of compliant samples” is distinct from HWE.3.BP4 “Verify hardware design.” As per BP text, the latter does not demand using physical samples. Correspondingly, the former only implies that if physical samples will be needed, then they shall be production data compliant.

3.21.1.6 Specify verification measures

HWE.3.BP1 requires the identification of the necessary verification infrastructure and environment setup. This may be supported using simulation of the environment. This environment can be hardware-in-the-loop simulation, vehicle network simulations, or digital mockups.

Verification results can support the updating of simulation models.

3.21.1.7 Verification logs as evidence for verification results

A large amount of logged data may be generated, which will be available via, for example, verification logs. Also, if verification is performed manually, the results may be provided in different levels of detail.

Verification results need to be meaningfully abstracted or derived from such log data. Still, for BP6 “Summarize and communicate,” the verification results will need to be further summarized.

3.21.2 Rating rules within the process

3.21.2.1 Verification measure definition

Rating rules:

> **[HWE.3.RL.1]** If entry/exit criteria are reasonably specified for a set of verification measures instead of each individual verification measure, then HWE.3.BP1 shall not be downrated.

3.21.2.2 Automation of verification measures

Rating rules:

> **[HWE.3.RL.2]** If a verification measure is automated and the correctness, completeness, and consistency of the corresponding scripts and programs are not addressed in the verification measure definition, then HWE.3.BP1 shall be downrated.

3.21.2.3 verification vs. traceability/consistency

The state-of-the-art testing not only comprises testing derived from requirements, but also exploratory testing based on experience, such as “error guessing based on knowledge.” This is valuable as it adds to the quality of the product. Therefore, exploratory tests that are based on experience cannot, by definition, be traced or consistent with the hardware requirements.

Still, traceability is needed between exploratory test cases and their results.

Rating rules:

> **[HWE.3.RL.3]** If verification measures representing exploratory tests, which, by definition, cannot be traced to the hardware design, have no such traceability, then HWE.3.BP4 shall not be downrated.

3.21.2.4 Traceability and consistency

See also subchapter 3.3.1 of SYS.1 and 2.3.4 here.

3.21.3 Rating rules with other processes at level 1

3.21.3.1 Verification measures selection vs. release plans

Rating rules:

> **[HWE.3.RL.4]** If selection of verification measures is properly done but based on an inadequate or incomplete release plan, then HWE.3.BP3 shall not be downrated.

3.22 HWE.4 Verification against Hardware Requirements

The purpose is to ensure that the complete hardware is verified to provide evidence for compliance with the hardware requirements.

3.22.1 General information

3.22.1.1 Scope of the HWE processes

See subchapter 2.3.1.

3.22.1.2 BP "Ensure use of compliant samples"

Why is this base practice also needed in HWE.4? Will the samples used for HWE.3 not be the same as the ones used in HWE.4? This might be but generally is not the case. From a HW development lifecycle perspective, reasons are, for instance:

- In early development phases breadboards are used which are typically not used anymore in the context of HWE.4.

- The samples for HWE.3 and HWE.4 do not always have the same assembly placement/mounting options (German: "Bestückung") due to different verification goals, verification environments, and HW delivery purposes in HWE.3 and HWE.4.

3.22.1.3 Why HWE.4 does not require samples whose compliance to the HW design have been verified

It is not necessary to use the very same physical samples for both HWE.3 and HWE.4. Reason:

The BP "Ensure use of production data-compliant samples" exists in both HWE.3 and HWE.4. It ensures detecting varying manufacturing quality, or even manufacturing deviations to make sure that these are tolerable enough for samples to be used for verification. As long as this is the case for any sample, in fact different samples may be used in HWE.3 and HWE.4, respectively. In other words: the very same sample does not necessarily need to undergo both HWE.3

and HWE.4; if HWE.3 verifies a production data-compliant sample X is compliant with the hardware design, then it can be concluded that another production data-compliant sample Y, which successfully underwent HWE.4, automatically is design-compliant, too (and vice versa).

Also remember that a PAM is not a lifecycle model and therefore cannot predefine any order of processes or sample processing (see Automotive SPICE® PRM and PAM subchapter 3.3.4). However, both design compliance and hardware requirements compliance must still be proven by means of physical samples.

3.22.1.4 Verification measures do not require using identical hardware samples

The need for physical hardware samples depends on the actual content of the verification measure. There are verification measures that do not require –or cannot even be performed on – physical hardware samples, such as calculations, simulations, reviews, and analyses; still, simulation models can be improved based on measurements with real physical samples.

This is why HWE.4.BP2 “Ensure use of compliant samples” is distinct from HWE.3.BP4 “Verify hardware design”. As per BP text, the latter does not demand using physical samples. Correspondingly, the former only implies that if physical samples will be needed, then they shall be production data compliant.

3.22.1.5 Specify verification measures

HWE.4.BP1 requires the identification of the necessary verification infrastructure and environment setup. This may be supported using simulation of the environment. This environment can be hardware-in-the-loop simulation, vehicle network simulations, or digital mockups.

Verification results can support the update of simulation models.

3.22.1.6 Verification logs as evidence for verification results

A large amount of logged data may be generated and made available, for example, via verification logs. Also, if verification is

performed manually the results may be provided in different levels of detail.

Verification results need to be meaningfully abstracted or derived from such log data. Still, for the purpose of BP6 “Summarize and communicate,” the verification results must be summarized further.

3.22.1.7 Analysis of the hardware design (BP4) vs. HWE.3

See subchapter 3.20.1.3.

3.22.2 Rating rules within the process

3.22.2.1 Verification measure definition

Rating rules:

> **[HWE.4.RL.1]** If entry/exit criteria are reasonably specified for a set of verification measures instead of each individual verification measure, then HWE.4.BP1 shall not be downrated.

3.22.2.2 Automation of verification measures

Rating rules:

> **[HWE.4.RL.2]** If a verification measure is automated and the correctness, completeness, and consistency of the corresponding scripts and programs are not addressed in the verification measure definition, then HWE.4.BP1 shall be downrated.

3.22.2.3 Exploratory verification vs. traceability/consistency

The state-of-the-art testing comprises not only of testing derived from requirements but also exploratory testing based on experience, such as “error guessing based on knowledge”. This is valuable as it adds to the quality of the product. Therefore, exploratory tests that are based on experience cannot, by definition, be traced or consistent with the hardware requirements.

Still, traceability is needed between exploratory test cases and their results.

Rating rules:

> **[HWE.4.RL.3]** If verification measures representing exploratory tests, which, by definition, cannot be traced to the hardware requirements, have no such traceability, then HWE.4.BP5 shall not be downrated.

3.22.2.4 Traceability and consistency

See also subchapter 3.3.1 of SYS.1 and 2.3.4 here.

3.22.3 Rating rules with other processes at level 1

3.22.3.1 Verification measures selection vs. release plans

Rating rules:

> **[HWE.4.RL.4]** If selection of verification measures is properly done but based on an inadequate or incomplete release plan, then HWE.4.BP3 shall not be downrated.

3.23 SUP.1 Quality Assurance

The purpose is to provide independent and objective assurance that work products and processes comply with defined criteria and that non-conformances are resolved and further prevented.

3.23.1 General information

As stated in the purpose, the quality assurance process covers all independent quality assurance activities for relevant work products (i.e., not just software source code) and processes based on defined criteria.

Note that agile methods and practices are not in conflict with Automotive SPICE®. For further details see also subchapter 2.5.2.

Quality assurance must also be extended to the quality of supplier deliveries according to the project scope and context. Supplier-related quality assurance activities must be clearly identified and may also include assessments.

3.23.2 Rating rules within the process

3.23.2.1 Ensure independence of quality assurance

Independence of quality assurance means that it must be unbiased and free of conflict of interest.

This relates also to the four-eyes principle, such as self-monitoring of the author of a work product is not sufficient. Additionally, independent quality assurance (e.g., a quality engineer role) within the project must not be taken by the project manager or a developer of the same project; however, they may take such responsibilities in a different project.

In teams where independence in terms of organizational structure cannot be effectively ensured (e.g., small organizations), as a minimum independent reporting and escalation paths must be established to the appropriate levels of management. This is also valid if quality assurance activities are performed by externally contracted persons.

Another perspective related to independence beyond organizational structure is the financial or resources perspective. Authority for financial decisions may negatively influence seemingly adequate operational or organizational independence.

3.23.2.2 Define criteria for quality assurance

Criteria for quality assurance including the appropriateness of work products and process performance shall be identified based on the project context. The criteria may consider internal rules and expectations, but also external inputs such as customer requirements and standards.

From the identified criteria, methods for ensuring the quality of all relevant work products and processes are to be decided on.

3.23.2.3 Assure quality of work products

To assure the quality of work products, typically work product reviews (e.g., peer review, walkthrough, inspection) as verification methods are performed. These reviews are based on predefined review methods and criteria. The review participants must further possess sufficient expertise to review the work products (e.g., testers may review the requirements to ensure testability). The review participants need to be documented.

In practice, quality engineers may also participate in selected work product reviews of the system and software engineering processes. This may raise the effectiveness of performing the reviews and controlling the quality of work products. However, non-participation of quality engineers in such reviews does not necessarily lead to a weakness in BP3 if there are other ways established to assure and control these work products by the independent quality assurance.

By evaluating SUP.1 BP3, evidence shown during the system and software engineering process interviews may be also considered. This means weaknesses identified related to ensuring the consistency of the system or software engineering related work products may be considered during evaluation of this base practice.

Rating rules:

> **[SUP.1.RL.1]** If activities for work product quality assurance do not make use of any type of review-oriented method, then BP3 shall be downrated.
>
> **[SUP.1.RL.2]** If quality assurance of work products checks for pure existence of work products only without considering any criteria for content correctness and structure, then BP3 shall not be rated higher than P.
>
> **[SUP.1.RL.3]** If quality assurance of work products (BP3) is rated N or P, then PA 1.1 shall not be rated higher than L.

3.23.2.4 Assure quality of process activities

Quality assurance of process activities aims to discover non-conformances in process performance. This may include process assessments, audits, checks of correct application of methods and tools, and the adherence to defined processes, reports and lessons learned that will improve processes for future process performance.

Rating rules:

> **[SUP.1.RL.4]** If quality assurance of process activities is based on performing process assessments (either by a customer or internally) only, the indicator BP4 shall be downrated.
>
> **[SUP.1.RL.5]** If quality assurance of process activities (BP4) is rated N or P, then PA 1.1 shall not be rated higher than L.

3.23.2.5 Ensure resolution of non-conformances

Non-conformances identified in any kind of quality activities, such as reviews must be resolved.

Often there is a lack of understanding that the initiator of the non-conformance alone cannot determine what exactly has to be improved. Therefore, joint coordination between initiator and resolver may be necessary in a timely manner.

Non-conformances have a priority and a due date for resolution. The resolutions shall be agreed upon.

Prevention of non-conformances ensures they will not occur systematically again. Various measures of technical, mathematical, or organizational nature can support process quality assurance for this purpose. Supporting techniques may be continuous monitoring, cross-functional collaboration, root cause analysis with 5-why, lessons learned, etc.

Rating rules:

> **[SUP.1.RL.6]** If non-conformances related to work products are neither identified nor documented, then BP3 shall be downrated.
>
> **[SUP.1.RL.7]** If non-conformances related to process activities are neither identified nor documented, then BP4 shall be downrated.

3.23.2.6 Escalate non-conformances

Non-conformances that cannot be resolved by project staff or those that are delayed too much shall be escalated to an appropriate level of management. This is to cover all relevant stakeholders (e.g., technical and quality management, customer, suppliers). Escalations are resolved by means of corrective actions ensured by these stakeholders; note that not each and every non-conformance will be subject to escalation.

Established criteria of urgency and impact can help identifying the appropriate level of management. A defined status model for escalations can help tracking the escalations to completion.

The resolution of escalated non-conformances may be an indication for the effectiveness of escalation measures.

Rating rules:

> **[SUP.1.RL.8]** If escalations are not followed up on by effective corrective actions, then BP7 shall not be rated higher than P.

3.24 SUP.8 Configuration Management

The purpose is to establish and maintain the integrity of relevant configuration items and baselines and make them available to affected parties.

3.24.1 General information

Configuration management covers the identification of configuration items, namely relevant inputs and work products of relevant stakeholders of processes and the control of modifications. Furthermore, it covers the management of baselines for different disciplines, sites, processes, etc.

Configuration management for different domains (such as hardware engineering, system or software development) may differ. Different approaches and “de facto” standard tools may be appropriate. Nevertheless, there should be an overall consistent baseline set available.

Configuration management importance and complexity rises with the size of the organization, number of interfaces, and the number of work products that need to be maintained. The chosen configuration management approach must be suited to handle the given complexity adequately.

3.24.2 Rating rules within the process

3.24.2.1 Identification of configuration items

As a starting point, all work products needed to produce the final product should be considered as configuration management items. Example: while requirements, design, test cases and results etc. may be identified configuration items, a meeting agenda or minutes may not be. This may also include input work products from external sources like externally-procured software libraries. Moreover, it is not an obligation to include development tools themselves as configuration items (though at least their version used should be documented) unless they may become a part of the product to be controlled.

The further identification of configuration items needs to consider the organization, domains and respective stakeholders. As there can be several reasons to include or exclude configuration items, criteria shall be defined. Such criteria can be derived for example from formal requirements (e.g., in safety or security), policies, application parameters, categories such as documents, requirements, source code, deliveries etc. As configuration items identification can support quality assurance (SUP.1) and vice versa, quality-driven criteria may also be given or derived for it.

For configuration management it is crucial to establish control of changes to the relevant product it is intended to support. Therefore, the selection criteria should be verifiable on the outcome of the configuration item identification.

Rating rules:

> **[SUP.8.RL.1]** If the configuration items identification fails to include stakeholder needs related to the product(s) to be controlled, the indicator BP1 shall not be rated higher than P.
>
> **[SUP.8.RL.2]** If the identification of configuration items does not sufficiently cover the relevant work products (BP1), and this results in relevant work products not being controlled, baselined and reported, the indicators BP4, BP5 and BP6, respectively, shall be downrated.

3.24.2.2 Configuration management mechanisms

To support organizations for the availability of configuration items, the configuration management enables and supports the parallel working on configuration items. For each product domain, different configuration management practices and tools may be required to fit those needs.

Configuration management mechanisms need to be adequate for the number and composition of configuration items to be managed. A rather small number of configuration items can be managed in a simplified form.

Configuration items may have very different characteristics regarding the effort required for their creation, change and

maintenance. For example, changes to hardware configuration items (e.g., schematics and layout for printed circuit boards) typically require higher effort and time, while software-related configuration items like configuration files can be changed faster.

Performing the SUP.8 process should establish a “configuration management system.” However, that does not necessarily mean one single tool or repository for all configuration items. Configuration items may be stored in different repositories (e.g., hardware schematic drawings within their dedicated design tool, software code in a versioning environment, software design models as a CASE tool database element), thus indicating a need for an individual version control tool, for instance.

However, the resulting complexity of a rising number of configuration items stored in multiple locations may not be adequate in every context. Distributed storage and handling still must follow the idea of an integrated “configuration management system” that supports its mechanisms, such as control, baselining, and reporting, across all subsystems.

The software domain benefits from the usual lower effort and time needed to make changes to their configuration items but should benefit from parallel work as much as possible without conflicts. As this may result in a high complexity to be managed, different and additional practices may be necessary depending, for example, on the size of the development team and the resulting amount of actual parallel work performed. For instance, branching and merging is a practice to create different versions (branches) of a set of configuration items. This allows for making changes and then optionally merging them back or even forward to other configuration items. Branches can satisfy different purposes such as single release creation and maintenance, stabilization and troubleshooting of versions, or preparation of versions that can be supported for long time bug fixing only (frozen branch). Branching and merging guarantees valid statuses of configuration items and organizes parallel changes.

Specifically for software source code, branching and merging often refers to a codebase. Here, branching and merging further supports

software quality-related objectives (e.g., dependent on the level of release a branch is leading to, software metrics specific for various branches, different definitions of code test and review coverage).

However, branching and merging generally relates to any configuration items, not only to the software source code level.

Rating rules:

> **[SUP.8.RL.3]** If there is no dedicated configuration management tool in place but the established procedure is adequate for the complexity of the product to be developed, then BP3 shall not be downrated.
>
> **[SUP.8.RL.4]** If the established mechanisms for configuration management cannot support the complexity related to the product, the indicator BP3 shall be downrated.

3.24.2.3 Baselines

Configuration baselines are frozen (i.e., read-only) sets of configuration items at a specific point in time. They can act as reference point for future changes and support rollbacks when necessary. Baselines can be driven by purpose and time.

Expectations for defining and establishing baselines are:

- Internal and external baselines triggers are possible. For example, internal baselines can provide requirements and design information to testing and verification departments. An externally triggered baseline is, for example, necessary to fulfill SPL.2.BP8 for the release package delivered to the intended customer.

- Baselines may be created to cover different sub-domains, disciplines, sites, processes etc., e.g., hardware vs software, design vs test work products. In this case, baselines should be organized hierarchically: if there is, e.g., a software baseline including all relevant software files and development documentation, and another one for hardware, then a higher-level baseline needs to connect these two subdomain baselines

(e.g., via a product bill-of-material). Furthermore, it is important to consider that these two subdomain baselines and the higher-level baseline may use different configuration management subsystems. On the highest level an overall baseline should cover all required configuration items regardless of type, domain, etc.

- The baselines contain complete and consistent sets of configuration items required to reproduce the progress achieved between the baselines.

3.24.2.4 Completeness and consistency

Completeness and consistency of configuration items and baselines is important to ensure the overall quality and integrity. It requires an appropriate set of measures for ensuring their completeness and consistency.

This can be supported by using traceability information, results of verifications, data of version control systems and change control systems for verifying that changes are properly documented.

For example, dedicated configuration audits may verify the suitability of configuration items and integrity of the baselines for the respective domain, organization, and product.

Rating rules:

> **[SUP.8.RL.5]** If baselines for different disciplines or processes are not consistent, or if overall baselines do not exist, the indicator BP7 shall be downrated.
>
> **[SUP.8.RL.6]** If establishing baselines (BP5) is downrated, the indicator BP7 shall be downrated.

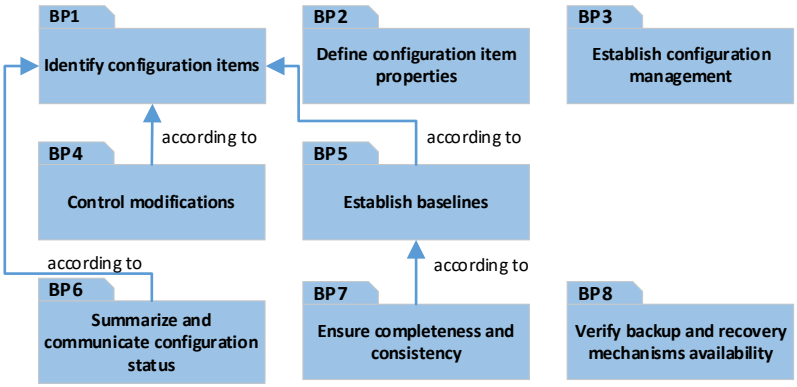
3.24.2.5 Verify backup and recovery mechanisms' availability

Backups are an aspect of ensuring the availability and integrity of configuration items and baselines for all stakeholders. Factors that influence the availability and performance of backups are, for

example, frequency of changes, storage location, their retention period, and the recovery process.

Backup and recovery mechanisms are recognized as an important foundation of configuration management. This is often performed and maintained centrally by an IT organization, and as such often regarded as being outside of the scope of the configuration management personnel of a project. If backup and recovery is based on, or represented by, internal IT policies or strategical decisions (e.g., outsourced IT), then this may not be rated negatively.

Backup and recovery mechanisms must not be confused with archiving.



3.25 SUP.9 Problem Resolution Management

The purpose is to ensure that problems are identified, recorded, analyzed, and their resolution is managed and controlled.

3.25.1 General Information

The Problem Resolution Management Process covers the management of all issues where, for example, more than one stakeholder is involved, or which are not resolved immediately.

For instance, a customer may be the initiator of a problem report that is addressed by the development project. In some cases, there even is no direct interaction between the customer and the development project, where the interface is managed by a customer support organization, for example. Such levels of interfaces need to be considered.

3.25.2 Rating rules within the process

3.25.2.1 Problem identification

The identification of problems includes the following aspects:

- if applicable, project lifecycle phase in which problem is recorded (e.g., during prototype construction, series development
- if applicable, initiating communication interfaces between project-specific disciplines, affected domains and subprojects (e.g., software platform, AI build, hardware sample)
- supporting information required, e.g., for reproducibility, frequency of problem occurrence or observed effects and patterns.

3.25.2.2 Determination of cause and impact

The expectations for an adequate cause and impact determination of problem records cover the following aspects:

- the systematic evaluation of the root causes
- the systematic evaluation of potential effects of the identified problems on other systems (e.g., use of standard software components across different projects)
- the systematic consideration of common problems, and potential common causes, in the same application (e.g., re-use of erroneous code in software clones, variants)

Rating rules:

> **[SUP.9.RL.1]** If the identification and recording of problems (BP1) is rated P or N due to an imprecise description, then BP2 shall be downrated.

3.25.2.3 Authorize urgent resolution action

When

- the timeframe for resolving of a resolution is not suitable or too short, or
- regular problem management and resolution is not possible short-term,

then urgent resolution might be necessary. Urgent resolution requires special authorization.

Examples are releasing ad-hoc instructions, product call-backs, or implementing a workaround. To prevent further damage and/or harm, urgent resolution actions may also include unconventional approaches, such as disabling certain functionalities or the entire systems.

However, such short-term and workaround actions (i.e., divergent to the defined process) still need to be synchronized with permanent problem resolution (i.e., according to the defined process). Urgent resolution action is to avoid – or limit – harm but not serve an excuse for not following systematic problem resolution.

Rating rules:

[SUP.9.RL.2] If rules for urgent problem resolution are defined, but urgent resolution has so far not been needed, then BP3 shall not be downrated.

3.25.2.4 Alert notification

Preparing for an alert notification as a means for problem resolution involves communicating the problem to relevant customers and stakeholders, independent of the originator of the problem.

This step identifies issues in connected or distributed projects, systems, and related variants. Proactive alert notification may be needed to inform their direct customers, receiving platforms or even authorities.

Such notifications may be triggered based on criticality, type, or root cause of the problem. In highly-automated development environments, also problems rated as less critical/urgent may lead to the information of affected parties (e.g., by using common problem management tools).

For all alert notifications to be effective, usable and suitable descriptions with clear and concise language are necessary, reflecting the recipient's level of understanding.

In the context of deciding on alert notifications, the precision and comprehensiveness of the problem description should be taken into account.

3.25.2.5 Multi-dependencies and status tracking to closure

In the context of SUP.9, problems may be organized hierarchically, meaning a set of problems may be identified to be sub-problems of a more general problem.

Furthermore, as SUP.9.BP5 says, a problem might be resolved by means of change requests and/or development tasks. Especially in presence of ALM and/or PLM tools, such development tasks and change requests can be represented by their own repository/database objects. This signifies change requests and development tasks are often directly linked to the problem.

Thus, the status tracking of a problem needs to be consistent with the status of all consequent change requests and tasks, etc. The closure of a problem, in particular, will require the closure of its associated change requests and tasks.

In Figure 3-3, problem A is resolved by means of a change request, which in turn is processed by means of development tasks 1 and 2. The status situation, however, is not consistent: a problem cannot be closed if its resolution actions are not closed first. Since, a problem can – but does not have to be resolved – via change requests, problem B in Figure 3-3 shows another inconsistent example.

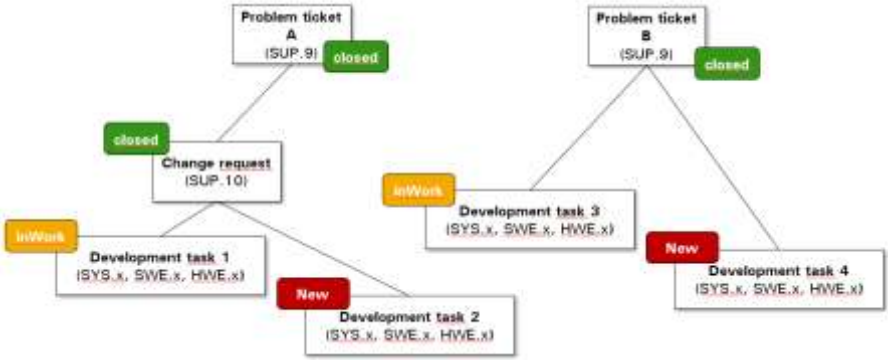
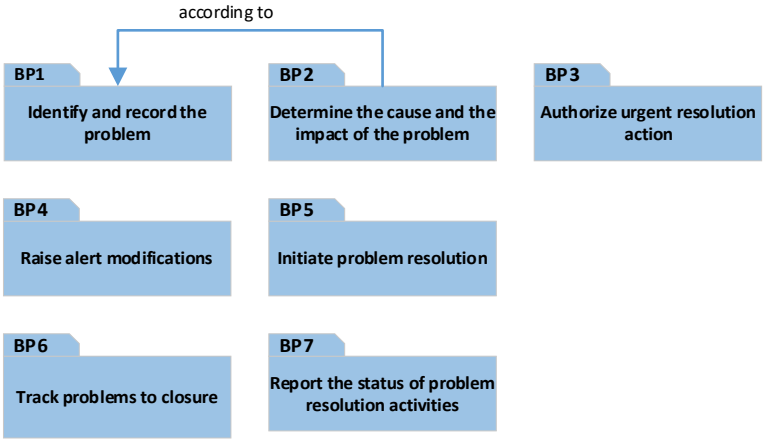


Figure 3-3: Examples of inconsistent status between problems and other associated work items

Any problem can be tracked using the status information assigned to it. When the status of a problem is set to resolved, the problem solving shall be confirmed by the problem report initiator by early and objective evaluation if applicable (e.g., in system demos or inspection meetings) or shall be documented in a reproducible manner.

Rating rules:

> **[SUP.9.RL.3]** If the status of a problem is set to resolved while associated work items (such as change requests or development tasks etc.) are not yet completed, then SUP.9.BP6 shall be downrated.


according to
BP1
Identify and record the problem
BP2
Determine the cause and the impact of the problem
BP3
Authorize urgent resolution action
BP4
Raise alert modifications
BP5
Initiate problem resolution
BP6
Track problems to closure
BP7
Report the status of problem resolution activities

3.26 SUP.10 Change Request Management

The purpose is to ensure that change requests are managed, tracked and implemented.

3.26.1 General information

The Change Request Management Process covers the management of all change requests.

Change request management may be using the same workflow approach as Problem Resolution Management (SUP.9) or an independent, fully separated one. In both cases, the decision authority and interaction of their stakeholders is of high importance according to the organizational and/or project-specific aspects like affected disciplines (e.g., system, software, electronics), affected domains (e.g., platform, COTS-Software), internal and external stakeholders or affected sites.

Change requests might be initiated by problems, which needs to be visible by means of a corresponding traceability and considered during tracking (see also subchapter 3.25.2.5).

3.26.2 Rating rules within the process

3.26.2.1 Change request identification and recording

The identification and recording of change requests include the following aspects:

- if applicable, project lifecycle phase in which change request is recorded and needed (e.g., during prototype construction, series development)
- if applicable, initiating communication interfaces between project-specific disciplines, affected domains and subprojects (e.g., software platform, AI build, hardware sample)
- supporting information required, such as alternatives and variable content for a change, references or demonstrators

Traceability between change requests, problems, affected work products and corresponding baselines must be ensured over all affected disciplines and all affected domains considering the project-specific complexity.

3.26.2.2 Analysis and assessment of change requests

The expectations for an adequate analysis of change requests cover these aspects:

- The input from all relevant stakeholders (internal and external) is considered including technical aspects and potential side effects, such as degraded functionality or compatibility problems.
- Feasibility, risks, complexity and impact regarding the potential changes are systematically evaluated and documented.
- Modification and potential alternatives are documented.
- Acceptance criteria for confirming implementation are established (e.g., selection of existing regression test case(s), newly developed test case, review of all modified work products).
- The change request is compliant with agreed regulations and policies.

The analysis of change requests should be capable of identifying affected work products. The process performance indicator PA 1.1 of this process therefore should reflect the importance of the analysis result.

Rating rules:

> **[SUP.10.RL.1]** If the analysis omits addressing potential side effects, the indicator BP2 shall not be rated F.
>
> **[SUP.10.RL.2]** If the identification and recording of changes (BP1) is rated P or N due to insufficient content, then BP2 shall be downrated.

3.26.2.3 Decision authority

Due to often more sensitive information like actual efforts, timelines, delegation, subcontracting or different stakeholders participating, the formation of a decision authority may become mandatory.

This decision authority, which is for simplification considered as *change control board* (CCB) is expected to cover these aspects:

- All affected disciplines are appropriately represented.
- All required stakeholders are represented (e.g., project manager, tester, customer sales manager, Product Owner).
- The participants have the necessary authority to take decisions.
- CCB takes decisions in time; delegates issues, if necessary.
- Agreement and approval in suitably timely manner, for supporting the alignment of changes with planned releases (see SPL.2 BP1).
- Depending on the organizational/project structure and/or constraints (e.g., platform responsibility, budget, effort), there may be multiple CCB's, such as hierarchical or organizational CCBs which may have to be represented as well.

A decision authority depends on analysis results for its approval and can be expected to provide its share, but not to provide a verification or repetition of such analysis results.

Rating rules:

> **[SUP.10.RL.3]** If not all relevant disciplines or stakeholders are represented in the approval authority, the indicator BP3 shall not be rated F.
>
> **[SUP.10.RL.4]** If it is apparent that approval decisions are not taken or not taken in time without justification, the indicator BP3 shall be downrated.
>
> **[SUP.10.RL.5]** If the analysis of the change request (BP2) is rated P or N, the indicator BP3 shall not be rated higher.

3.26.2.4 Parallelism and traceability of change requests

Change request management often creates parallel work activities, such as changes relating to other change requests, work products, problems, etc. This parallelism may be reflected in handling, linking of parallel or parent/child relationships of change requests to each other or even to tasks and problems.

Said parallelism may quickly become complex, challenging the management of all related disciplines and processes throughout the different states or workflow of them until closure.

Traceability of change requests should enable a structured handling of such a parallelism. The process performance indicator PA 1.1 of this process should reflect the importance of the traceability.

Rating rules:

> **[SUP.10.RL.6]** If the rating of establishing bidirectional traceability (BP4) is downrated due to missing dependencies between change requests and affected work products, the indicator BP2 shall be downrated.

3.26.2.5 Confirmation of Implementation

When confirming change request after implementation the following aspects may need to be considered:

- A review of the implemented change requests ensures that all relevant processes (e.g., SYS, SWE, MLE, MAN, and SUP) are applied and corresponding work products are updated accordingly.
- Subsequent activities, actions and tasks are reflected, such as trainings, inspect and adapt meetings, reporting, etc.

Rating rules:

> **[SUP.10.RL.7]** If the confirmation of implemented changes misses that relevant processes are not applied, the indicator BP5 shall be downrated.
>
> **[SUP.10.RL.8]** If the confirmation of an implemented change request is not including agreed acceptance criteria or policies, the indicator BP5 shall be downrated.
>
> **[SUP.10.RL.9]** If the analysis of change requests (BP2) is rated P or N due to missing information regarding their

implementation confirmation, the indicator BP5 shall be downrated.

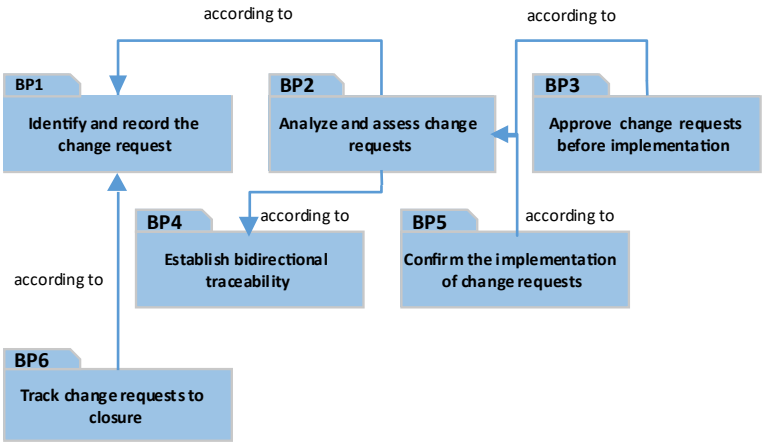
3.26.2.6 Track change requests to closure

Any change request action is to be tracked to closure of which there may be more than one final state. Final closure may depend on feedback from the initiator of the change itself, which should therefore be sought in early and objective evaluation, for example, in system demos or inspection meetings.

Rating rules:

[SUP.10.RL.10] If the initiator of the change request is not sufficiently authorizing the closure of the change and this is substantial regarding the project, the indicator BP6 shall be downrated accordingly.

[SUP.10.RL.11] If the initial recording of change requests (BP1) is rated P or N due to missing information about the initiator or reason, the indicator BP6 shall be downrated.



3.27 SUP.11 Machine Learning Data Management

The purpose is to define and align ML data with ML data requirements, maintain the integrity and quality of all ML data, and to make them available to affected parties.

3.27.1 General information

Machine learning needs data for the training process of MLE.3 and the testing activities of MLE.4. To ensure success of the training process, data of a controlled quality are required which are aligned with the ML data requirements of MLE.1.

Because of cost and effort, ML data are often not only collected and categorized for usage in a single project. Instead, data collection and categorization might be performed continuously by a dedicated organizational entity. A project would then make use of the ML data pool provided by the organization.

SUP.11 Machine Learning Data Management is related only to the data management activities which are required by the MLE process group of the assessed project.

3.27.2 Rating rules within the process

3.27.2.1 Establish an ML data management system

The management of ML data requires an ML data management system which includes a suited lifecycle management to mark the status of a data item. This system could be in simple cases reduced to a storage system and metadata maintained on the data itself, provided the file type supports metadata.

The assessor needs to judge the suitability of this ML data management system based on the ML data requirements and the amount of data collected for the ML training and test.

For the rating, the assessor needs to understand especially the interfaces to provide and categorize data.

For example, object detection in video might require a way to label video sequences by company external workers for supervised training. Reinforcement learning might require an interactive approach (e.g., question answering) to judge the correctness of output created by the model, which in this case could be part of SUP.11, depending on the setup, even if the judgement is part of the MLE.3 Machine Learning Training process. In both cases, the ML data management system must generally provide an opportunity to import all data and store it.

Rating rules:

> **[SUP.11.RL.1]** If required ML data management activities are not supported by the ML data management system, then SUP.11.BP1 shall be downrated.

3.27.2.2 Develop an ML data quality approach

ML data with known quality is an important factor for the success of the MLE group. This requires an approach which defines the ML data quality criteria to be met and how to check that the ML data satisfies the ML data quality criteria. One important aspect of ML data quality criteria is the avoidance of biased data. Biases to avoid may include sampling bias (e.g., gender, age) and feedback loop bias.

Usually, the amount of data required for the ML training and test is so high, that the analysis of the quality of the ML data uses statistical methods.

ML data quality criteria shall be defined before data is used for the ML processing/training. Examples of ML data quality criteria are relevant data sources, reliability and consistency of labeling, completeness against ML data requirements.

3.28 MAN.3 Project Management

The purpose is to identify and control the activities and establish resources necessary for a project to develop a product in the context of the project's requirements and constraints.

3.28.1 General information

The purpose of the process Project Management is to cover all aspects of planning, monitoring and tracking.

In Automotive SPICE® 4.0, all planning activities are covered in MAN.3 Project Management and PA 2.1 Process Performance Management process attribute of the respective processes.

Release planning and the management of release baselines represent the determining of functional content to be implemented and are addressed in SPL.2 Product Release and SUP.8 Configuration Management.

3.28.1.1 Changed concept in Automotive SPICE® 4.0 (define and monitor)

The formulation "define and monitor" is used for the base practices BP4 (work packages), BP5 (estimates and resources), BP6 (skill, knowledge and experience), BP7 (interfaces and commitments), and BP8 (schedule).

The term "define" addresses the setup of artifacts or documented information where the term "monitor" covers the continued re-evaluation of artifacts and documented information.

To ensure consistency, adjustment is done based on issues found in monitoring of all the aforementioned aspects. Consistency here means that all planning aspects demonstrate feasibility of the project.

3.28.2 Rating rules within the process

3.28.2.1 Scope of work

The scope of work must cover the motivation (goals), the boundaries including project and product scope, and the constraints of the project. Describing only the product to be developed is not sufficient.

Rating rules:

[MAN.3.RL.1] If the scope of work (BP1) is a product description only, the indicator BP1 shall not be rated higher than L.

[MAN.3.RL.2] If the scope of work (BP1) is not appropriately documented and updated during the project lifecycle, the indicator BP9 shall not be rated higher than L.

[MAN.3.RL.3] If the required content of the scope of work (BP1) is distributed over several work products, the indicator BP1 shall not be downrated.

3.28.2.2 Defining project planning artifacts

Based on the scope of work, the project lifecycle (BP2) is defined according to the size, complexity and the context of the project. The lifecycle defines major milestones of the project like project start, sample deliveries or start of production, and it defines the development phases. The lifecycle may be standardized on organizational level and adapted to project specific conditions or developed solely for one particular product development.

It is not always recommendable to set up a detailed work package planning for the entire project lifecycle, since there are many changes to the scope of work during the conduct of a project. Therefore, as a rule of thumb, an appropriate detailed planning of work packages encompasses the next two releases.

New concept: IIC – work package (instead of work breakdown structure) work packages can also be:

- tickets in a tracking system

- entries in a planning tool
- cards on a Kanban board

Dependencies of work packages have to be documented in the work package definition. Usually this includes a precondition that must be present before starting the work package and/or a certain output that serves as precondition for another work package.

Dependencies become critical when they include a certain risk of missing project milestones.

The planned effort for work packages shall be determined based on a reproducible estimation.

Not acceptable examples are a simple “best guess” by project manager or estimates by a single person only without any further review or without involvement of affected parties.

Another aspect that must be considered is an adequate size for the work packages. The size of the work packages should not exceed the time of one – max. two – monitoring cycles to ensure proper monitoring of the work packages. As an alternative, the monitoring of work packages refers to defined status information to evaluate the degree of completion.

The necessary resources should include, for example, people, development tools, hardware samples, infrastructure and test equipment.

Skills are specific characteristics of people like the ability to communicate, to learn new things, leadership, etc. Knowledge includes also process, project and product specific training. Experience is a result of long-term practicing certain activities.

Interfaces (BP7) of the project can be

- a development partner, such as
  - other development parties contracted by customer, working on the same system,
  - other development parties contracted by the assessed organization, working on the same system (see ACQ.4),
  - the customer, working on the same system,

- internal departments (sales, purchasing, quality management etc.),
- service providers (for, e.g., infrastructure, cloud services),
- platform development, or
- other development sites.

For all the interfaces that have an impact on the results of the assessed project, the commitments have to be documented and monitored. In case of any deviation an escalation mechanism shall be effective.

Rating rules:

[MAN.3.RL.4] If the dependencies between work packages are not identified, the indicator BP4 shall not be rated higher than L.

[MAN.3.RL.5] If any of the following:

- start and end date
- planned effort and actual effort
- correction of effort or end date if work package is not completed on time

is missing for work packages, the indicator BP4 shall not be rated higher than P.

[MAN.3.RL.6] If the estimation approach used and the origin of the estimates are not reasonable, the indicator BP5 shall not be rated higher than P.

[MAN.3.RL.7] If the size of work packages is larger than two monitoring cycles of the project and the progress of work packages cannot be measured, the indicator BP4 shall be downrated.

[MAN.3.RL.8] If critical dependencies in the schedule are not determined, the indicator BP8 shall be downrated.

[MAN.3.RL.9] If training for process, project and product specific topic is not provided to project participants, the indicator BP6 shall be downrated.

[MAN.3.RL.10] If more than two development partners are involved and agreements and commitments are not documented and signed, the indicator BP7 shall be downrated.

3.28.2.3 Monitoring

A proper monitoring cycle ensures a timely detection of deviations regarding work packages (BP4), estimates (BP5), skills, knowledge and experience (BP6), agreed interfaces and commitments (BP7), and schedule (BP8). As a rule of thumb, a weekly monitoring is in most the cases appropriate. In the context of the project a more frequent monitoring might be needed (e.g., when project is in task force mode). The monitoring of skills, knowledge and experience may be decoupled from the monitoring of the other planning aspects.

Tracking of corrective actions may also be linked to SUP.9 Problem Resolution Management.

Rating rules:

[MAN.3.RL.11] If the monitoring cycle is not appropriate to detect deviations of planned versus actual planning items, the respective indicators for monitoring (BP4, BP5, BP6, BP7, BP8) shall not be rated higher than P.

3.28.2.4 Actual project progress

In practice a project manager cannot resolve all issues that arise during project monitoring. Resource issues are frequently decided by higher level management, while schedule deviations may be discussed with the customer. It is essential for the success of a project that mechanisms for communication and escalation with all involved stakeholders are effective.

3.28.2.5 Release management

Releases and their management are not dealt with in a single process only but represent a topic distributed across several processes:

- Generally, the project must define which information, work products, and products have to be delivered to or received from all relevant stakeholders (MAN.3.BP7).
- The planning of releases is based on the work packages (BP4), and the schedule (BP8) in MAN.3 and on the release plan in SPL.2 Product Release.
- The release must be built from configured items (SPL.2.BP4) which relates to configuration management that ensures integrity (SUP.8). Deadline information of product releases will be part of schedules (MAN.3.BP8).
- Release planning is also covered in the requirements processes (SYS.2.BP2, SWE.1.BP2, HWE.1.BP2 and MLE.1.BP2) which expect a mapping of requirements to specific releases (see Note of those BPs).

Rating rules:

[MAN.3.RL.12] If product release deadlines or milestones are not consistent with the release scope, the indicators BP8 and BP9 shall be downrated.

[MAN.3.RL.13] If for the current and next release the expected activities are defined and monitored completely, the indicators BP4 and BP8 shall not be downrated.

[MAN.3.RL.14] If links between different types of planning information are not supported by tools, this shall not be used to downrate the indicator BP9.

3.28.2.6 Consistency of planning information

For the rating of the project management process, it is important that the definition and monitoring of project attributes like work packages, estimates and resources, project interfaces and dependencies will be evaluated individually.

All these attributes have strong dependencies that require to maintain consistency between them. Therefore, the adjustment of project management work products is combined into one Base Practice to ensure consistency.

Activities of the master project and subprojects have to be aligned and consistent, such as project plans for the different engineering domains. Dependencies between these plans must be easily identified and mapped. Adjustments to activities have to be considered in all relevant planning artifacts.

For project management, explicit links between plans and schedules, for example, are not required. Consistency can be reached by comparing planned versus actual and if needed adjusting planning information.

Rating rules:

[MAN.3.RL.15] If planning information of sub-projects is not consistent with the overall planning, the indicator BP9 shall be downrated.

3.28.2.7 Estimation of change requests and problem resolution

During an automotive development, change requests, risk treatment activities, problems, quality issues and defect removals can be anticipated. This needs to be reflected in the project planning information.

Rating rules:

[MAN.3.RL.16] If the definition of work packages, effort and resources, and the definition of schedule(s) do not sufficiently reflect change requests, risk treatment activities, problems, quality issues and defect removals, the indicators BP2, BP8 and BP5 shall be downrated.

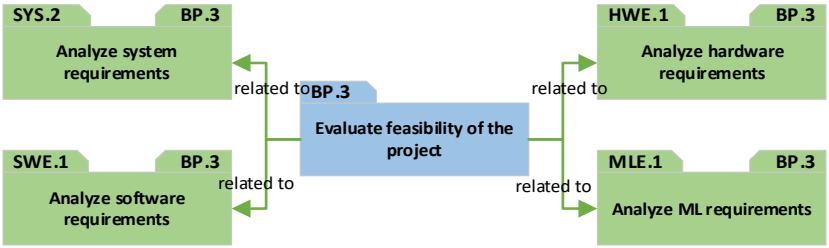
3.28.3 Rating rules with other processes

Rating rules:

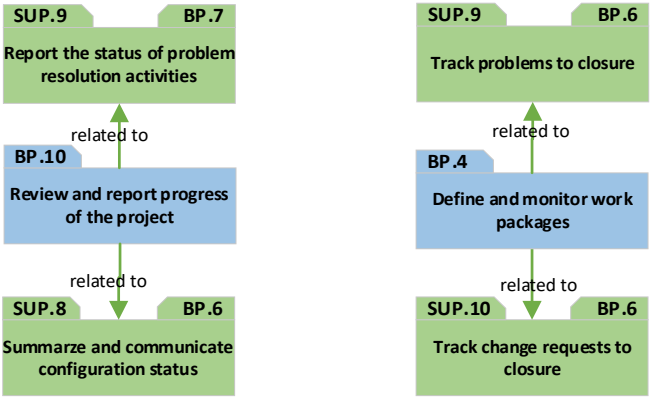
[MAN.3.RL.17] If the definition of risk treatment activities (MAN.5.BP5) is downrated due to incomplete definition of risk treatment activities, then the indicator BP4 shall be downrated.

[MAN.3.RL.18] If the definition of Risk treatment activities (MAN.5.BP5) is downrated due to insufficient identification of

required project resources for risk treatment activities, then the indicator BP5 shall not be rated higher than L.



[MAN.3.RL.19] If one of the related BPs in the requirement processes for system (SYS.2.BP3), hardware (HWE.1.BP3), software (SWE.1.BP3) or machine learning (MLE.1.BP3) is downrated due to a missing or incomplete analysis regarding technical feasibility, the indicator BP3 shall be downrated.



[MAN.3.RL.20] If one of the related BPs regarding status of configuration items (SUP.8.BP6) or regarding the status of problems (SUP.9.BP7) is downrated due to a missing or incomplete report, the indicator BP10 shall be downrated.

[MAN.3.RL.21] If one of the related BPs regarding tracking of problems (SUP.9.BP6) or regarding tracking of change requests (SUP.10.BP6) is downrated due to a missing or incomplete status tracking, the indicator BP4 shall be downrated.

3.29 MAN.5 Risk Management

The purpose is to regularly identify, analyze, treat and monitor process related and product-related risks.

3.29.1 General information

Development projects must deal with events or incidents that potentially have a negative impact on achieving the project goals, like in terms of schedule, cost, quality, and functional content. In Automotive SPICE®, such events and incidents are called "undesirable events."

To perform effective risk management for the development project, determining the risk management scope is required. This includes potential incidents that can occur during the project lifecycle regarding the activities performed under responsibility of the project, or regarding relevant work products or regarding resources of the project. The risk scope is strongly dependent on the context of the project.

3.29.2 Rating rules within the process

3.29.2.1 Sources of risks

Risk management should consider at least process-related and product-related undesirable events for which the risk has to be evaluated.

Examples for process-related undesirable events are:

- schedule deviations
- project progress not according to the plan
- unavailability of resources, including human resources
- commitments from development partners not fulfilled.

Examples for product-related undesirable events are:

- defects delivered to customer
- chosen platform is unsuitable for customer application
- missing requirements
- impact of changes to system behavior

Rating rules:

[MAN.5.RL.1] If risk management does not consider process-related undesirable events, the indicator BP1 shall be downrated.

[MAN.5.RL.2] If risk management does not consider product-related undesirable events, the indicator BP1 shall be downrated.

[MAN.5.RL.3] If undesirable events and sources of risks are not updated on a regular basis, the indicators BP1 shall not be rated higher than P.

[MAN.5.RL.4] If risks are not monitored and updated regularly, the indicator BP6 shall be downrated.

3.29.2.2 Identify potential undesirable events and determine risks

For risk identification, knowledge of – and experience with – the product, its operating environment and the project must be available. Also, reuse of components from former projects may involve product-related risks.

To evaluate the risk (of an undesirable event) all possible influencing effects must be considered. The risk is characterized by probability of occurrence and the severity of impact that is related to potential undesirable events. The characterization supports the prioritization of risks and their mitigations. Usually, probability and severity are rated according to a discrete scale (e.g., “high,” “medium,” “low”) that is easy to understand and to reproduce. These ratings are then combined to represent a risk value.

Rating rules:

[MAN.5.RL.5] If aspects of reused development results are not considered in the identification of undesirable events, BP2 shall be downrated.

[MAN.5.RL.6] If severity of impact and probability of occurrence of undesirable events are not evaluated in a reproducible way, the indicator BP3 shall be downrated.

3.29.2.3 Risk treatment

The risk value is the basis for prioritization of the risks and for the application of risk treatment. For potential incidents with low risk value, risk treatment can be limited to monitoring of the risk.

Concepts for risk treatment may include:

- experiences from problem resolution to avoid re-occurrence
- experiences from previous projects
- accepting the risk
- transferring the risk
- countermeasures to reduce the impact or the probability of the risk

When risk mitigation failed and the incident occurs, the problem resolution process is typically applied to solve the problem.

Rating rules:

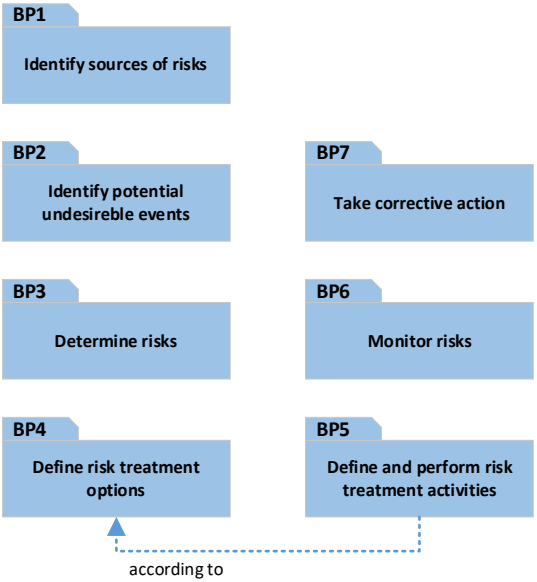
[MAN.5.RL.7] If the definition of risk treatment activities is not suitable to evaluate the progress and effectiveness of risk treatment activities, then BP5 shall not be rated higher than P.

3.29.2.4 Monitor risks

Risk monitoring has to be performed regularly in relation to the project milestones and the release plan.

Rating rules:

[MAN.5.RL.8] If monitoring of risk and progress of the mitigation activities is not performed regularly (e.g., synchronized with project monitoring cycle), then BP6 shall not be rated higher than L.



[MAN.5.RL.9] If BP4 is downrated due to missing risk treatment options, the indicator BP5 shall be downrated.

3.29.3 Rating rules with other processes

None.

3.30 MAN.6 Measurement

The purpose is to collect and analyze data relating to the work products developed and processes implemented within the organization and its projects, to support effective management of the processes.

3.30.1 General information

As MAN.6 Measurement was not included in the former VDA scope the experience regarding typical pitfalls in applying this process in an assessment is very limited. Therefore, in this guideline the number of rating rules for this process is rather low.

3.30.2 Rating rules within the process

3.30.2.1 Key metrics

To understand the behavior of processes, their characteristics and limitations it is necessary to measure certain key metrics. These metrics shall reflect information needs of the management. Examples of information that may be used in process related metrics are:

- lead time
- resource consumption
- review coverage and verification coverage vs. defects.

For each metric, the following must be documented:

- the formula (if applicable),
- input data,
- frequency of reporting, and
- the control limits or threshold values.

Rating rules:

[MAN.6.RL.1] If the metric specification is completely documented in terms of the topics listed above, BP2 shall not be downrated.

3.30.2.2 Information needs

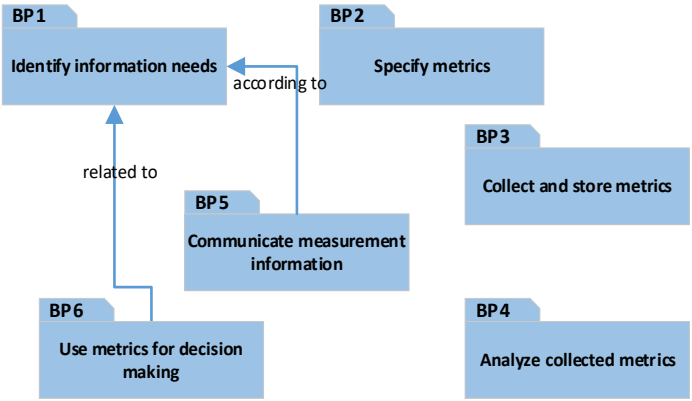
The information needs should be aligned with the stakeholders and decision-makers.

For decision-making, a review of the analysis and interpretation of the collected metric is needed.

Rating rules:

[MAN.6.RL.2] If BP1 is downrated because there is no involvement of the decision-makers, the indicators BP5 and BP6 shall be downrated.

[MAN.6.RL.3] If the analysis and the interpretation of the metrics are not reviewed before decision-making, BP4 and BP6 shall be downrated.



3.30.3 Rating rules with other processes

None.

3.31 PIM.3 Process Improvement

The purpose is to continually improve the organization's effectiveness and efficiency through the processes used and ensure alignment of the processes with the business needs.

3.31.1 General information

As PIM.3 Process Improvement was not included in the former VDA scope the experience regarding typical pitfalls in applying this process in an assessment is very limited. Therefore, in this guideline the number of rating rules for this process is rather low.

3.31.2 Rating rules within the process

3.31.2.1 Process improvement application

Process improvement is established in the automotive industries and known, for instance, as "lessons learned processes" or "continuous improvement." In the context of Automotive SPICE®, it is important to act in a structured way.

The process improvement process is applicable, for example, to:

- achieving a target capability profile
- realizing internal process optimization
- removing particular weaknesses in process performance
- conducting activities to achieve target performance determined from MAN.6 Measurement

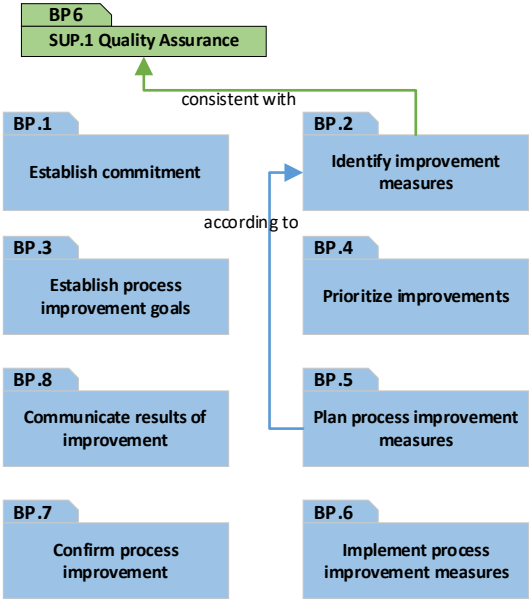
3.31.2.2 Improvement goals

A clear improvement goal shall be committed and defined. Improvement goals shall be communicated within the organization.

Rating rules:

[PIM.3.RL.1] If there is no commitment regarding the improvements, BP1 shall not be rated higher than P.

[PIM.3.RL.2] The rating of BP5 shall be in line to the rating of BP2.



3.31.3 Rating rules with other processes

Rating rules:

[PIM.3.RL.3] If the identification of process improvement measures does not consider aspects of improvements identified in SUP.1.BP6, PIM.3.BP2 shall be downrated.

3.32 REU.2 Reuse of Products

The purpose is to ensure that reused (work) products are analyzed, verified, and approved for their target context.

3.32.1 General information

The reuse of components in the automotive business is a common method to establish sustainable development and to save effort and time in development. It is important to reflect this aspect according to the context of a project.

Products for reuse can range from vehicle system components to software units. The process REU.2 is focused on the usage of previously developed products.

Common approaches for reuse are, for example:

- use of platform components (e.g., one ECU for multiple applications)
- use of standard components (e.g., operating system)
- use of legacy software components
- use of FOSS
- use of commercial off-the-shelf components (e.g., CAN driver)
- use of configurable components (e.g., park distance control with 8, 12 or 16 channels)

For a reused work product, it must be confirmed that it fits to the intended use of the target system.

For further aspects of reuse of components, please also refer to 2.5.3 .

3.32.1.1 Analysis of reused products

The analysis of reused products shall respect the current functional and non-functional requirements, environment, and stakeholder expectations.

As result of the analysis, constraints and needed qualification for the product shall be defined.

3.32.1.2 Ensure qualification of products for reuse

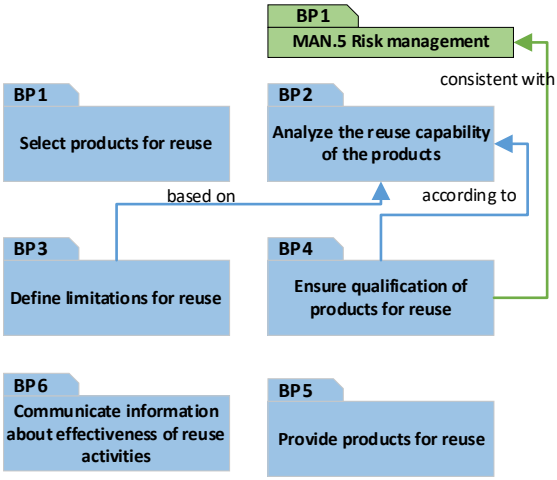
Until the qualifying is finished, reused products must be evaluated to determine if they are relevant for the risk management scope.

3.32.2 Rating rules within the process

Rating rules:

[REU.2.RL.1] If the analysis of the reuse capability of the product does not consider constraints of the target architecture, BP2 shall not be rated higher than P.

[REU.2.RL.2] If the constraints and defined qualification for the reused product is not based on the analysis (BP2), then BP3 and BP4 shall be downrated.



3.32.3 Rating rules with other processes

Rating rules:

[REU.2.RL.3] If MAN.5.BP1 is downrated due to not considering reuse products in defining risk sources, then REU.2.BP4 shall be downrated.

4 Rating guidelines on process capability level 2

The previously described performed process is now implemented in a managed fashion (planned, monitored and adjusted) and its work products are appropriately established, controlled and maintained.

On capability level one process-specific indicators are used to evaluate the extent to which the outcomes of the process are achieved. Assessors regularly use the base practices to assess a project's capability. These are activity-based indicators. In addition, there are information items which are result-oriented indicators. Guidance on possible content of the output work products is documented in Annex B of Automotive SPICE®.

At higher capability levels, generic practices and related information items are available as indicators. As the names imply, these indicators are not process-specific and have to be used for all processes. Hence, they must be interpreted for each single process individually.

On capability level 1, the intent is to achieve the purpose of the process. Therefore, the assessor judges whether the result of the process is appropriate with respect to the context of the project including achievement of all outcomes.

On capability level 2, all activities that contribute to the purpose of the process and to the achievement of capability level 2 itself (like, e.g., reviews) must be planned and controlled. All resulting work products will be considered regarding configuration management and quality assurance.

Additionally, on capability level 2, objectives (e.g., planning goals) and strategies for the activities that must be planned for the assessed process have to be specified and documented. Also, requirements for all relevant work products of each process must be defined. These requirements include such information as content and structure (e.g., as table of contents), history, layout, etc. Very often, the requirements for a work product are documented as a work product template, including instructions for using the template.

If tools are used, it should be documented how the tools must be used, such as which fields are mandatory, and which are optional.

There is a strong dependency between MAN 3. Project Management and process attribute PA 2.1 Process Performance Management. Regarding the process attribute PA 2.2 Work Product Management there is a strong dependency to SUP.1 Quality Assurance and SUP.8 Configuration Management. For details refer to the chapters on PA 2.1 and PA 2.2 below.

4.1 PA 2.1 Process Performance Management

The process performance management process attribute is a measure of the extent to which the performance of the process is managed.

4.1.1 General information

As a basis for a repeatable and sustainable process performance management, first the objectives for the performance of the process need to be identified. A process performance objective provides the starting point for detailed process-specific planning. This in turn is the basis for tracking and adjusting the process performance.

The process performance strategy defines the operational approach supporting the achievement of the process performance objectives and considering the scope of the process application. In the PAM, the process performance strategy is defined as an information item as part of process attribute PA 2.1.

PA 2.1 relates to the specific process. In contrast, the ability of a project to plan, monitor, and adjust for the overall project (i.e., across processes and domain) to reach the project's goals is the purpose of the Project Management process (MAN.3). However, there is a relationship between process performance management attribute PA 2.1 of the different processes and MAN.3. Therefore, the guidelines provided for Project Management (MAN.3) should be considered correspondingly for PA 2.1 (e.g., granularity of activities, frequency of monitoring activities).

In contrast to capability level 3, explicit process descriptions are not required for fulfilling PA 2.1 and PA 2.2, providing the PA achievements are met.

4.1.2 Rating rules within the process attribute

4.1.2.1 Identify the objectives for the performance of the process (GP 2.1.1)

Process performance objectives need to be defined as a basis for the detailed planning (see GP 2.1.2). They are not a repetition of process outcomes at capability level 1. In this respect, performance objectives target how to organize the establishment of the process outcomes.

The definition of process performance objectives can be done based on the SMART principles (specific, measurable, achievable, relevant and time-bound).

Process performance objectives can either be quantitative (e.g., requirements to be implemented for specific releases, maximum/minimum efforts to be spent) or qualitative (e.g., adherence to an Automotive SPICE® capability level). However, the defined quantitative process performance objectives do not require process measurements as defined in MAN.6 or Automotive SPICE® capability level 4.

Examples are:

- effort, costs, or budget targets (e.g., min & max limits)
- process-specific deadlines for work packages, or frequency of activities (e.g., dates for process-specific work packages)
- metric-oriented objectives, e.g.,
  - for SUP.10: max. number of open change requests 6 weeks before the next product release;
  - for SUP.8: not more than 60% of configuration items in status “in work” two months before next delivery baseline.

Based on, but also complementing, the identified process performance objectives, a strategy is defined. A strategy addresses:

- the operational approach, e.g.,
  - proceedings, including the monitoring of the performance of the process;
  - methodology.

- the strategy scope within the process, e.g.,
  - development sites;
  - application domain-specific differences (e.g., software drivers vs. powertrain software);
  - disciplines (such as SUP.8 differently for software and hardware, or a combined approach);
  - options due to socio-cultural differences.

This leads to the following rating rules for the indicator GP 2.1.1 in relation to process performance objectives:

Rating rules:

> **[PA2.1.RL.1]** If a standard process does not include standard or generic objectives, but process performance objectives have been defined, then GP 2.1.1 shall not be downrated.

4.1.2.2 Define a strategy for the performance of the process (GP 2.1.1)

The process performance strategy defines the operational approach in terms of proceedings and methodologies supporting the achievement of the process performance objectives considering the scope of the process application.

A strategy can be represented by:

a) presentation slides of an organizational unit describing the purpose and objectives of their individual processes and providing sufficient explanation of corresponding proceedings
b) existing tools that enforce certain workflows (e.g., including user interfaces with mandatory or restricted edit fields, attributes in a document management, configuration or change request management system)
c) automated (or partially automated) workflows implemented by tools and scripts, e.g.,
   - automatically generated test result report frame with traceability links to the test case specification
   - build tools including mandatory static software verification
   - continuous integration approaches
   - continuous delivery approaches

d) an appropriate media source, such as a photo of a whiteboard drawing, video or podcast explaining key elements of the process performance

A process performance strategy does not necessarily have to be documented for each process separately. The strategy does not have to be described in a specific document either. Common and useful practices are to document strategy elements applicable to multiple processes in joint documents, e.g.:

- joint test strategy for system testing related processes
- joint change request and configuration management strategies (as change requests are to be placed against concrete versions of artifacts or entire product baselines)
- communication strategies of several processes in a common project manual

Adherence to a given strategy and the effectiveness of the strategy are essential. However, the existence of a documented strategy does not necessarily ensure that the strategy is established and effective.

Therefore,

- the necessary comprehensiveness and information detail indicated by the examples above is always context-dependent, and
- people interviewed (besides the author of the strategy) shall be independently able to demonstrate the knowledge and awareness of the strategy.

It is the responsibility of the assessors to check whether an existing strategy is also effective.

Note that the definition and existence of documented information related to a strategy is not relevant for the rating of PA 1.1 of the process.

If process performance objectives and aspects of the strategy are derived from an existing standard process, the suitability of this standard process's objectives and strategy for the specific project context needs to be considered. Still, the definition of standard

objectives and standard strategies in a standard process according to capability level 3 is not required for GP 2.1.1.

This leads to the following rating rules for the indicator GP 2.1.1:

Rating rules:

> **[PA2.1.RL.2]** If a standard process including a standard or generic strategy does not exist, but a strategy is effectively adhered to, then GP 2.1.1 shall not be downrated.
>
> **[PA2.1.RL.3]** If a strategy is not documented in a specific document, but there is evidence of the strategy adhered to by all relevant parties, then GP 2.1.1 shall not be downrated.
>
> **[PA2.1.RL.4]** If the strategy is not described in a single process-specific document, but strategies of different processes are combined in one document, then GP 2.1.1 shall not be downrated.
>
> **[PA2.1.RL.5]** If a documented process performance strategy does not exist, but there is evidence that an effective strategy is followed, then PA 2.1.1 shall not be downrated.

4.1.2.3 Plan the performance of the process (GP 2.1.2)

To ensure a proper planning for the process performance, the following aspects can be considered for the detailed planning based on the performance objectives, and consistent with the strategy:

- A schedule with activities for the considered deadlines and their start date, due date, effort, assigned resources (typically, e.g., for engineering activities), sequence, and dependencies.
- Alternatively, some activities represent “background tasks” or are event-triggered (e.g., check-in/check-out activities in configuration management, addressing change requests in change management, progress tracking in project management). These are usually not planned to be performed at particularly foreseeable points in time. For such activities, defining an effort percentage for, e.g., allocated human resources can be sufficient.

- Estimates for, e.g., effort, size of work products are to be reasonable and reproducible. Reasonable buffer time (e.g., for bug fixing, vacation) needs to be planned.

Planning must include at least the activities related to process performance (PA 1.1), process performance management (PA 2.1, e.g., planning and monitoring of the process) and work product management (PA 2.2., e.g., review of work products).

Evidence of the planning can be visible in, but not restricted to, for example:

- schedule(s)
- ALM or PLM tools
- open-item list or backlogs
- task board (such as Kanban board)
- separate documents (e.g., in a meeting plan as part of a general stakeholder communication matrix)

Planning may not necessarily be documented specifically for each process in distinct documents. For example:

- A project-wide schedule or work breakdown structure (according to MAN.3) may contain the activities of the individual processes.
- A project-wide effort estimation (according to MAN.3) may include estimates for the individual processes.
- A central department for system testing (according to SYS.5) plans the test engineering/execution activities in one document, but allocation of the infrastructure and equipment is planned in another document or tool.

However, process-specific planning may also be documented in one particular/single document for the process. Achieving PA 2.1 does not require a process description. GP 2.1.2 only requires the existence of a detailed planning based on the performance objectives and the strategy.

Organizations do not need to structure the activities to be planned and monitored in a way that resembles the PAM structure or assessment indicator structure. It is therefore the responsibility of

the assessors to map evidence to the assessment indicators of the PAM.

This leads to the following rating rules for the indicator GP 2.1.2 in relation to a process performance strategy:

Rating rules:

> **[PA2.1.RL.6]** If no process description is available, but all expected planning information based on the performance objectives exists, then GP 2.1.2 shall not be downrated.
>
> **[PA2.1.RL.7]** If the determination of critical dependencies of activities and work packages is not considered in the process performance planning, then GP 2.1.2 shall be downrated.
>
> **[PA2.1.RL.8]** If supporting activities are not planned as explicit activities but are planned as percentage or absolute number of hours over a certain period of time, then GP 2.1.2 shall not be downrated.
>
> **[PA2.1.RL.9]** If the objectives or the strategy for the performance of the process (GP 2.1.1) is downrated due to missing suitability to achieve the process outcomes, then GP 2.1.2 shall be downrated.
>
> **[PA2.1.RL.10]** If planning and monitoring only includes the activities for capability level 1, but not the reviewing of work products according to GP 2.2.4, then GP 2.1.2 shall be downrated.

4.1.2.4 Determine resource needs (GP 2.1.3)

The following aspects need to be covered in the project and adequately documented:

a) The need of human resources, as well as physical and material resources, to perform the process-specific work packages is determined based on the planned activities.
b) Responsibilities, commitments, and authorities (e.g., access rights, release of work products) to perform the process activities of the project need to be defined, assigned, communicated, and agreed.

c) Responsibilities and authorities to verify process-specific work products need to be defined, assigned, communicated, and agreed upon.
The required experience, knowledge, and skills are defined; needs can either be process- or product-specific, or both (e.g., methods, tooling, needed algorithms).

Distinct from GP 3.1.2, all definitions for responsibilities and authorities can be made specifically for the process performance without considering a standard process and standard role definitions.

This leads to the following rating rules for the indicator GP 2.1.3:

Rating rules:

[PA2.1.RL.11] If the determination of resource needs only relates to human resources, but does not include relevant and necessary physical or material resources, then GP 2.1.3 shall not be rated higher than P.

[PA2.1.RL.12] If resource needs are adequately covered but role definitions provided by a standard process are not available, then GP 2.1.3 shall not be downrated.

[PA2.1.RL.13] If the planning of the process performance (GP 2.1.2) is downrated because of not covering the process outcomes or the reviewing of process-specific work products (PA 2.2), then GP 2.1.3 shall be downrated.

4.1.2.5 Identify and make available resources (GP 2.1.4)

The identification and provision of resources shall cover the following aspects:

a) The individuals/people required for the process performance, process performance management and work product management are identified by name, and made available. Resource planning must be reproducible (e.g., rate of utilization is transparent, absences such as vacation and trainings are considered, procedures for planning in a matrix organization or distributed development are defined). A comparison of needed

human resources vs. allocated resources is available and during process performance (see also GP 2.1.5).

b) The physical and material resources required for the process performance, process performance management and work product management (e.g., tool licenses, samples, test equipment) are made available and used. A comparison of physical and material resources vs. allocated resources is available and maintained during project lifecycle (see also GP 2.1.5).

c) The individuals performing and managing the process and work products are qualified by, e.g., training, mentoring, or coaching to be able to execute their responsibilities. A qualification fit/gap analysis is performed. Necessary qualification measures are planned and performed in time, according to the needs of the process.

d) The information necessary to perform the process is made available to all affected individuals (e.g., manuals, project wiki).

This leads to the following rating rules for the indicator GP 2.1.4:

Rating rules:

[PA2.1.RL.14] If the identification and provision of resources relates to individuals only (while also physical or material resources need to be considered), then GP 2.1.4 shall not be rated higher than P.

[PA2.1.RL.15] If the indicator for determination of resource needs (GP 2.1.3) is downrated due to inadequately or incompletely defined resource needs, then GP 2.1.4 shall be downrated.

4.1.2.6 Monitor and adjust the performance of the process (GP 2.1.5)

In order to monitor whether the process is performed according to the strategy and the planning (see GP 2.1.1 and GP 2.1.2) or deviates, the following aspects are considered:

a) Actual data is continually compared with planned values (this means also that the granularities of planned and actual data match), e.g.,
   - dates against deadlines or durations,
   - booked effort against planned effort,
   - estimates for human resources, materials, and physical resource needs as well as needs for skills, knowledge, and experience match the process needs.
b) The comparison between planned and actual data should:
   - show the current state of progress, and
   - be performed at an adequate frequency.
c) There is documentation of monitoring activities, e.g., as
   - status report,
   - status meeting minutes, or
   - information in tools or repositories (including ALM or PLM tools).

In the case monitoring identifies deviations, then
- these are analyzed and root causes are determined, and
- either corrective measures to align performance with plans must be taken or
- plans are adapted.

This leads to the following rating rules for the indicator GP 2.1.5:

Rating rules:

[PA2.1.RL.16] If the levels of granularity of planning and monitoring do not match in absence of a consistent mapping in between, then GP 2.1.5 shall be downrated.

[PA2.1.RL.17] If the chosen frequency of monitoring activities is incapable of identifying deviations versus plan in time (e.g., not very early before the closure of the activity), then GP 2.1.5 shall be downrated.

[PA2.1.RL.18] If the indicator for planning the performance of the process (GP 2.1.2) is downrated due to incomplete planning of activities and work packages or missing determination of resource needs, then GP 2.1.5 shall be downrated.

4.1.2.7 Manage the interfaces between involved parties (GP 2.1.6)

The individuals and groups external to the process are determined.

Managing the interfaces should cover the exchange of information and work products and should include the following aspects:

a) responsibilities are defined, e.g.,
   - who delivers or communicates information
   - what is the purpose of information delivery
   - who are the receivers;

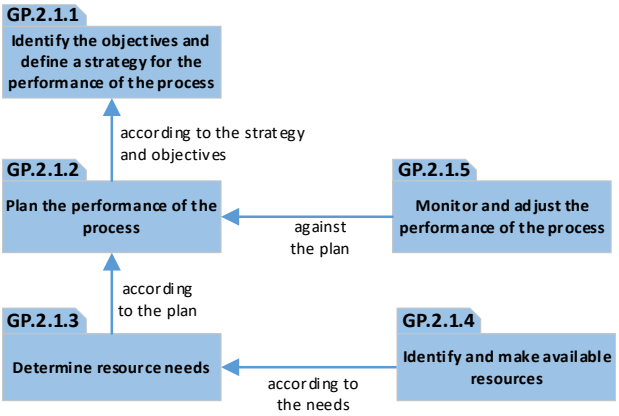
b) the technical interfaces and media type, e.g.,
   - email notifications with direct information or links/references
   - meetings with minutes sent to defined distribution lists
   - baseline reception
   - tasks in ALM or PLM tools
   - participants for the meetings are defined (depending on responsibilities, tasks, or processes)
   - the triggering mechanism is defined (push/pull);

c) communication between the involved parties is established, managed, maintained and effective in a repeatable fashion. This includes defining when and how frequent communication is done.

4.1.3 Rating consistency

4.1.3.1 Rating consistency within PA 2.1

The following figure shows relationships among GP 2.1.x generic practices:



4.1.3.2 Rating consistency with other processes and practices

During the assessment and in terms of the consistency of the assessment ratings, dependencies between PA 2.1 and assessment indicators of other processes are to be considered.

Accordingly, evidence obtained during the assessment needs to be analyzed regarding potential relevance for other dependent assessment indicators. Significant rating differences across dependent processes and process attributes (e.g., with more than one NPLF step) might be indicative of an inconsistent rating.

Dependencies of the PA 2.1 Process Performance Management attribute and its generic practices to other processes are:

- The ratings of Base Practices 4,5,6,7,8 of MAN.3 should be in line with the ratings of GP 2.1.2, GP 2.1.3, GP 2.1.4, GP 2.1.5, and GP 2.1.6 of the other processes.

4.2 PA 2.2 Work Product Management

This process attribute is a measure of the extent to which the work products produced by the process are appropriately managed.

4.2.1 General information

Relevant work products of the process are those that are required to fully achieve capability level 1, and additionally those work products required for successful implementation of the process attributes 2.1 and 2.2.

A work product may not only be a document but also, for example, a record or repository entry in a tool (e.g., change requests or problem reports implemented in a workflow tool are also work products).

Not included in the term “work product” are all standard process-related documents such as process descriptions, procedures, method descriptions, or role descriptions; these are applicable at capability level 3 only. Therefore, any weaknesses with handling these process assets that are not related to the content (e.g., improper versioning) must not be reflected in the process attribute 2.2 of the process under review. However, if organizational process documents are available, they can support the implementation of process attribute 2.2.

4.2.2 Rating rules within the process attribute

4.2.2.1 Define the requirements for the work products (GP 2.2.1)

Work product requirements include:

a) requirements defining content and structure, e.g.,

- layout, revision history, table of contents,
- technical content (e.g., for engineering work products, like requirement specifications, architectural descriptions, as well as management-oriented work products, such as schedules, minutes, open point lists),
- guidelines (e.g., programming or modelling guidelines), or

- standards.

b) appropriate review and approval criteria, e.g.,

- definition whether the work product needs to be explicitly reviewed, or is only implicitly reviewed by distributing it and implicitly approved in case there is no feedback (e.g., minutes, open-point-lists, reports etc.);
- review method, review coverage (including justification); review frequency (including justification), and review participants;

c) quality criteria – based on aspects a) or b).

Very often, the requirements for a work product are documented as a work product template or checklist. However, defining templates or checklists is not necessarily required by PA 2.2, as long as all aspects above are adequately documented.

This leads to the following rating rules for the indicator GP 2.2.1:

Rating rules:

[PA2.2.RL.1] If templates or checklists do not exist for the work product, but content, structure, review and approval and quality criteria are adequately covered and documented, then GP 2.2.1 shall not be downrated.

[PA2.2.RL.2] If standard work product templates provided by a standard process are available, but there is a defined specific solution that is effective although deviating from the standard process, then GP 2.2.1 shall not be downrated.

[PA2.2.RL.3] If standard work product templates provided by a standard process are available and used by the process, but do not fit the purpose of the process, then GP 2.2.1 shall be downrated.

4.2.2.2 Define the requirements for storage and control of the work products (GP 2.2.2)

Certain requirements regarding storage and control have to be defined for all relevant work products. These requirements must be

set-up for each identified work product (see also SUP.8.BP2 and subchapter 4.2.3.2).

The requirements for storage and control shall cover at least these minimal required aspects:

a) naming conventions
b) ownership
c) access rights (at least read and write permission)
d) if applicable: work product status model (states and transitions), workflow, approval and release proceedings
e) versioning rules (including baselining depending on the work product type)
f) storage media (e.g., configuration management tool, ALM/PLM tools)
g) distribution channels (communication mechanisms for releases and changes)

Some of this might apply to subitems of certain work products, e.g., requirements contained in a requirement specification may require naming conventions and status model.

The expectations for a status model and workflow definition (see aspect e) above), if applicable, cover the following aspects:

- criteria for status changes, and relevant stakeholders together with their responsibility and authorization, etc.
- work product status transitions follow the workflow to a final status and are tracked accordingly. There might be more than one final status (e.g., closed, rejected, cancelled), but it must be ensured that one of them is always reached (e.g., there is a status “resolved” but the status model defines an additional step “closed” that will usually not be reached).
- work products with a very simple status definition (e.g., living documents, meeting minutes, agendas) do not require a status model with transitions, workflow, approval, and release procedure.

This leads to the following rating rules for the indicator GP 2.2.2:

Rating rules:

[PA2.2.RL.4] If the requirements for storage and control do not cover the minimal required aspects, the indicator GP 2.2.2 shall be downrated.

[PA2.2.RL.5] If the requirements for storing and controlling work products do not cover versioning and storage requirements, then GP 2.2.2 shall not be rated higher than P.

[PA2.2.RL.6] If the definition of a status model for work products with a trivial status definition lacks definitions of workflow, criteria for status changes, stakeholder and their authorization, then GP 2.2.2 shall not be downrated.

4.2.2.3 Identify, store, and control the work products (GP 2.2.3)

All identified work products must be stored and controlled (indicator GP 2.2.3) according to their requirements (indicator GP 2.2.2). The following rule is defined because of this dependency.

Rating rules:

[PA2.2.RL.7] If the indicator for defining requirements for storage and control of the work products (GP 2.2.2) is downrated, then GP 2.2.3 shall not be rated higher than that.

4.2.2.4 Review and adjust work products (GP 2.2.4)

Work product reviews shall be performed against defined work product review criteria (see GP 2.2.1) in accordance with the planning (see PA 2.1). The execution of work product reviews does not necessarily require a formal review or inspection including a dedicated findings record but can also follow a less formal approach such as walk-throughs, or pair-programming, or implicit review by frequent use (e.g., living documents such as a schedule, or meeting minutes). With a more formal approach, the following aspects are to be demonstrated:

a) review information:
   - identification of the work product under review (including name and version)
   - date of the review
   - name(s) and roles of reviewer(s)

- review findings, if they are not immediately resolved in the review, e.g., in pair-programming
- review result (e.g., “Passed”, “Conditionally Passed”, “Failed/Re-review required”)
- used review method/approach
- if applicable, approval criteria

b) handling of review findings:
   - a procedure for handling of review findings has to be followed
   - review findings must be monitored and tracked until resolved.

This leads to the following rating rules for the indicator GP 2.2.4:

Rating rules:

[PA2.2.RL.8] If the proof of work product reviews does not cover all the required aspects to be demonstrated, then GP 2.2.4 shall be downrated.

[PA2.2.RL.9] If the proof of work product reviews for relevant capability level 1 related work products does not cover the name and version of the work product under review, review findings (unless immediately solved), and the used review and, if applicable, approval criteria, then GP 2.2.4 shall not be rated higher than P.

[PA2.2.RL.10] If for relevant capability level 1 related work products the review findings are not resolved, then GP 2.2.4 shall not be rated higher than P.

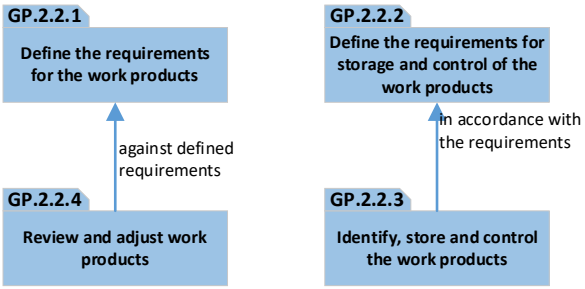
[PA2.2.RL.11] If work product reviews can be demonstrated according to all aspects above but are not explicitly documented in a formal review record, then GP 2.2.4 shall not be downrated.

[PA2.2.RL.12] If the indicator for defining requirements for the work products (GP 2.2.1) is downrated due to inappropriate review and, where applicable, approval criteria, then GP 2.2.4 shall be downrated.

4.2.3 Rating consistency

4.2.3.1 Rating consistency within PA 2.2

The following figure shows relationships among GP 2.2.x generic practices:



The generic practices of capability level 2 can be grouped into two main topics. The first one covers requirements, quality criteria, review, and adjustment of all relevant work products of the corresponding process (GP 2.2.1 & GP 2.2.4), whereas the second one covers the storage and control of those work products (GP 2.2.2 & GP 2.2.3).

4.2.3.2 Rating consistency with other processes and practices

During the assessment and in terms of the consistency of the assessment ratings, the dependencies between PA 2.2 and assessment indicators of other processes and practices need to be considered. Accordingly, evidence obtained during the assessment needs be analyzed regarding potential relevance for other dependent assessment indicators.

Dependencies of the work product management attribute PA 2.2 and its generic practices to other processes are:

- The rating of SUP.1.BP2 should be in line with the ratings of GP 2.2.1 of the other processes.

- The rating of SUP.1.BP3 should be in line with the ratings of GP 2.2.4 of the other processes.
- The rating of SUP.8.BP2 should be in line with the ratings of GP 2.2.2 of the other processes.
- The ratings of SUP.8.BP4 and SUP.8.BP5 should be in line with the ratings of GP 2.2.3 of the other processes.

Dependencies of the work product management attribute PA 2.2 and its generic practices to base practices of the same processes are:

- The rating of the base practice of ensuring consistency of work products for the same process should be in line with the rating of GP 2.2.4, if review is used as a primary means for establishing consistency (SYS.2.BP5, SYS.3.BP4, SYS.4.BP4, SYS.5.BP4, SWE.1.BP5, SWE.2.BP4, SWE.3.BP4, SWE.4.BP4, SWE.5.BP6, SWE.6.BP4, VAL.1.BP4, MLE.1.BP5, MLE.2.BP6, MLE.3.BP4, MLE.4.BP6, HWE.1.BP5, HWE.2.BP5, HWE.3.BP5, HWE.4.BP5, MAN.3.BP9, SUP.8.BP7).

5 Rating Guidelines on Process Capability Level 3

The previously described managed process is now implemented using a defined process that can achieve its process outcomes.

On capability level 2, all projects may use “their” own process if the requirements of Automotive SPICE® are fulfilled.

On capability level 3, the projects must use a standard process. A possibility to cover variations between projects is to describe tailoring guidelines. This derived process is the so-called “defined” process. The defined process has to cover all activities and work products of capability level 1 and 2 for the assessed project.

Large organizations would have problems with only one standard process. The organization may define several different standard processes (e.g., one standard process for each development site, or one standard for each business unit). The other possibility to cover variations between projects is the aforementioned description of tailoring guidelines. Based on predefined criteria, the process may be tailored to the needs of the project.

Exceptionally waivers for the standard process may be used (which should not be the rule). Assessors should check whether these exceptions have a rationale and are approved by appropriate organizational roles.

It must be kept in mind that the advantage of organizational processes is to standardize the approach to, for example,

- establish processes known by the stakeholders,
- establish interfaces to facilitate cooperation (also between different locations),
- facilitate introduction of new personnel or exchange personnel and infrastructural resources between projects,
- facilitate reuse of assets and work products,
- share good practices and lessons learned, or
- establish benchmarking.

The aim of establishing processes might get missed if there are too many variations of the processes. This should be reflected by the assessment result.

Note that capability level 3 only addresses the standard process description and its suitability (PA 3.1), and the effective adherence to a tailoring of a standard process (PA 3.2). Capability level 3 does explicitly not address

- the configuration management of the process descriptions as a work product,
- the storage of the process descriptions as a work product,
- handling of process description tools or repositories, or
- management of human resources for process improvement.

Such aspects would have to be addressed by capability level 1 and 2 of, e.g., ORG.1 Life Cycle Model Management in ISO/IEC 33060 and 33061, respectively, or Automotive SPICE® PIM.3 Process Improvement. Therefore, weaknesses regarding these aspects must not lead to any downrating on capability level 3.

5.1 PA 3.1 Process Definition

This process attribute is a measure of the extent to which a standard process is maintained to support the deployment of the defined process.

5.1.1 General information

The process defined is organization wide and no longer project specific. This includes, at least:

- a developed, established and maintained standard process including tailoring guidelines (GP 3.1.1)
- required competencies, skills, and experience for the identified roles performing the standard process (GP 3.1.2)
- required physical and material resources and process infrastructure needs for performing the standard process (GP 3.1.3)
- suitable methods and required activities for monitoring the standard process (GP 3.1.4)

Each of these aspects has to be rated only in the respective GP.

5.1.2 Rating rules within the process attribute

5.1.2.1 Establish and maintain the standard process (GP 3.1.1)

GP 3.1.1 covers the definition and maintenance of the standard process including tailoring guidelines.

Definition of the standard process

In order to define the standard process, its scope, purpose and intended use must be identified and correspondingly documented. The fundamental process elements will be covered by the standard process, such as process activities including detailed descriptions and required inputs/expected outputs including corresponding entry and exit criteria.

Rating rules:

> **[PA3.1.RL.1]** If scope, purpose and intended use are missing in the standard process definition, then GP 3.1.1 shall not be rated F.
>
> **[PA3.1.RL.2]** If process activities including descriptions are missing in the standard process definition, then GP 3.1.1 shall not be rated higher than P.
>
> **[PA3.1.RL.3]** If required inputs or expected outputs of process activities are missing in the standard process definition, then GP 3.1.1 shall not be rated higher than P.

Additionally, the sequence and interaction of process activities inside one process as well as the sequence and interaction of the process to other processes must be identifiable. This might also include parallel or iterative sequencing of activities, which are synchronized by, for instance, work product completion.

> **[PA3.1.RL.4]** If the sequence and interactions of process activities within the process or to other processes is not identifiable, then GP 3.1.1 shall not be rated higher than P.
>
> **[PA3.1.RL.5]** If the sequence and interactions of process activities within the process or to other processes is not explicitly documented as such, but is identifiable (e.g., by work product status and entry/exit criteria), then GP 3.1.1 shall not be downrated.

To support the execution of the process, guidance, procedures, method descriptions, and/or templates should be provided as needed.

> **[PA3.1.RL.6]** If explicit templates for the expected outputs are not provided, but corresponding detailed requirements regarding the expected content are existing, then GP 3.1.1 shall not be downrated.

Process performance roles must be identified and assigned to the standard process activities including their type of involvement (e.g., key responsibility roles by RASI/RASIC-matrix), responsibilities, and authorities.

[PA3.1.RL.7] If process performance roles are not identified and assigned to standard process activities, then GP 3.1.1 shall not be rated higher than P.

[PA3.1.RL.8] If the type of involvement of the process roles in the process activities is not defined, then GP 3.1.1 shall be downrated.

Here in GP 3.1.1, only the identification of the roles including their involvement is covered, whereas the detailed role descriptions are handled in GP 3.1.2 (see next subchapter 5.1.2.2).

[PA3.1.RL.9] If role definition details like competencies, skills, experience, or qualification methods are missing, then GP 3.1.1 shall not be downrated.

Maintenance of the standard process

The defined standard process needs to be continuously maintained according to corresponding feedback from monitoring the deployed process (see also GP 3.2.4 for corresponding input), and adapted process requirements (standards, regulations, laws, changed/new infrastructure, etc.). The maintenance should be documented in change requests and corresponding process version numbering. The validity of process versions needs to be defined, which includes:

- Date for obligatory use of the latest version of the standard process for all upcoming projects
- Handling of usage of new version of the standard processes or new process elements for running projects (e.g., not applicable for projects at a stage later than x)
- Mechanism to ensure availability of previous process versions

This leads to the following rating rules:

Rating rules:

[PA3.1.RL.10] If the standard process does not have a unique version identifier, then GP 3.1.1 shall not be rated F.

[PA3.1.RL.11] If changes between standard process versions are not documented and identifiable, then GP 3.1.1 shall be downrated.

[PA3.1.RL.12] If it is unclear when a new version of the standard process becomes mandatory for new projects, or by when running projects will have to switch to the new standard process, then GP 3.1.1 shall be downrated.

Nevertheless, for a project running in a late project phase, it could make sense to not switch to the latest version of an updated standard process. Therefore, a record of standard process versions must be kept. This also contributes to evidence to be demonstrated in case of product liability lawsuits.

[PA3.1.RL.13] If a project uses a former standard process version, the documentation of which is no longer available, then GP 3.1.1. shall be downrated.

Tailoring of the standard process

Deployment can be done with or without tailoring of the standard process, which is supported by corresponding tailoring guidelines (see also GP 3.2.1). Tailoring can be performed through different proceedings such as deleting, adding or selection between different elements of the process based on predefined criteria. Additionally, the responsibility for tailoring and corresponding approval must be defined.

[PA3.1.RL.14] If the tailoring guidelines do not include the responsibility for tailoring and corresponding approval, then GP 3.1.1 shall be downrated.

In case that neither the standard process as-is nor the existing tailoring guidelines cover the needs of a project, a corresponding deviation can be approved as an exception (e.g., by a waiver including rationale). In case of several approved similar deviations from the standard process, the process either needs to be reworked, or the tailoring guidelines must be updated.

[PA3.1.RL.15] If the same or similar deviations from the standard process are regularly approved (e.g., by waivers) without updating the standard process and/or tailoring guideline, then GP 3.1.1 shall be downrated.

5.1.2.2 Determine the required competencies (GP 3.1.2)

The standard process identifies and assigns required process performance roles to process activities, including their type of involvement (e.g., key responsibility roles by RASI/RASIC-matrix), responsibilities, and authorities (see GP 3.1.1, including corresponding rating). Thus, if this is missing, it must be downrated in GP 3.1.1, and not in GP 3.1.2.

Rating rules:

> **[PA3.1.RL.16]** If the type of involvement of the process roles regarding responsibilities, or authorities in standard process activities is not defined, then GP 3.1.2 shall not be downrated.

Nevertheless, the process roles need to be described in more detail including required competencies, skills, and experience, which is covered by the indicator GP 3.1.2. Furthermore, appropriate qualification methods for human resources need to be determined, maintained, and made available.

> **[PA3.1.RL.17]** If required competencies or required skills are missing for the defined process roles, then GP 3.1.2 shall not be rated higher than P.

> **[PA3.1.RL.18]** If qualification methods for human resources are either not determined, or not maintained, or unavailable for the defined roles, then GP 3.1.2 shall not be rated higher than P.

5.1.2.3 Determine the required resources (GP 3.1.3)

Requirements for human resources are covered by GP 3.1.2.

> **[PA3.1.RL.19]** If requirements for human resources are not determined, then GP 3.1.3 shall not be downrated.

However, other non-human resources like physical and material resources as well as process infrastructure needs must be determined, which is covered by GP 3.1.3. This includes the definition and description of the tools to be used (including qualification, if relevant, e.g., for safety critical use) and infrastructure, methods and responsibilities to ensure that the

needed work environment is available for the projects (e.g., licenses), or also samples.

> **[PA3.1.RL.20]** If required tools are not defined, then GP 3.1.3 shall not be rated higher than P.

5.1.2.4 Determine suitable methods to monitor the standard process (GP 3.1.4)

In order to monitor the effectiveness of the actual process performance, and the adequacy of the standard process by evaluation, corresponding methods and activities need to be defined. Adequacy relates to meeting requirements including coverage of capability level 1 and 2 expectations, compliance with internal predefined provisions, and relevant industry standards (e.g., reflecting the state of the art, but not proprietary customer-specific standards).

Successful process compliance checks or internal audits/assessments can show the effectiveness of a process, whereas a lack in process compliance for most projects can demonstrate that the process is not effective. However, process compliance checks cannot determine the adequacy of the standard process (e.g., relevant industry standards are not addressed).

Lessons learned or retrospective meetings can be used to get process feedback. Feedback should be documented in a defined way, analyzed, and taken into account for process development. In addition, there should be a defined and well-known way for project staff to give process feedback to the responsible process development organization. However, lessons learned, and project feedback may or may not include a reflection on relevant industry standards.

Automotive SPICE, being a PAM, is at the WHAT level (see Automotive SPICE® PAM subchapter 3.4). This means it cannot predefine any method which would be a HOW level decision. The only directive is: whatever methods are chosen, the process purpose of capability level 1 must be effectively met.

[PA3.1.RL.21] If the selected methods for monitoring the adequacy of the standard process comply only with internal provisions and do not reflect relevant industry standards or comparable methods capable of achieving capability level 1, then GP 3.1.4 shall be downrated.

[PA3.1.RL.22] If continual effectiveness and adequacy monitoring is performed, but no need for process improvements is reasonably identified, then GP 3.1.4 shall not be downrated.

[PA3.1.RL.23] If the standard process does not follow an international standard in terms of naming and structuring of work products, templates, activities, and roles then GP 3.1.1, GP 3.1.2, GP 3.1.3, and GP 3.1.4 shall not be downrated.

Furthermore, quantitative methods may be used to monitor effectiveness and/or adequacy of the standard process (e.g., efficiency, cost). However, quantitative methods are not mandatory on capability level 3.

[PA3.1.RL.24] If the determined standard process monitoring methods are only of qualitative nature, but still appropriate regarding effectiveness and adequacy, then GP 3.1.4 shall not be downrated.

5.1.3 Rating consistency within the process attribute

No explicit consistency dependencies were identified within this process attribute, and therefore, no corresponding rating rules were defined.

5.2 PA 3.2 Process Deployment

This process attribute is a measure of the extent to which the standard process is deployed as a defined process to achieve its process outcomes.

5.2.1 General information

The rating of process attribute 3.2 should reflect the degree to which the process is using the standard process considering the tailoring guidelines.

5.2.2 Rating rules within the process attribute

5.2.2.1 Deploy a defined process that satisfies the context specific requirements of the use of the standard process (GP 3.2.1)

The deployment of a defined process should include

- the project-specific selection and/or tailoring from the standard process using the defined tailoring guideline and criteria. The decisions made and the rationale for the decisions need to be documented;
- the verification that the defined process conforms with standard process and its tailoring guidelines and accordingly applied in the project. This must be done by an authorized role, e.g., process owner, process group, quality management or quality assurance. Evidence of the verification or a final release of the defined process needs to be documented.

Rating rules:

> **[PA3.2.RL.1]** If the defined process is not selected, documented and verified according to the tailoring guidelines, then GP 3.2.1 shall be downrated.

In case that the standard process – including existing tailoring guidelines – is not suitable for a project as defined process, this can be correspondingly approved as an exceptional case (e.g., by a

waiver including rationale). Therefore, the following rule must be considered:

> **[PA3.2.RL.2]** If deviations from the standard process are independently approved based on reasonable arguments (e.g., by a waiver), but not reflected in the tailoring guideline, then GP 3.2.1 shall not be downrated.

5.2.2.2 Ensure required competencies for the defined roles (GP 3.2.2)

Roles, responsibilities, and authorities for performing the defined process are assigned and communicated.

Ensuring required competencies includes:

- The assurance of appropriate skills and competencies for assigned personnel. Evidence that the assigned persons have the required qualifications (e.g., qualification records) should be available. The qualification must be in line with the skills and competencies defined in GP 3.1.2 for performing the standard process.
- If gaps in skills and competencies are identified, adequate qualification measures should be defined and monitored.
- The availability of suitable qualification measures for those performing the defined process. Availability guarantees that project members are qualified in time to perform the defined processes in the project.

Rating rules:

> **[PA3.2.RL.3]** If no evidence that the assigned persons have the required qualification is available, then GP 3.2.2 shall not be rated higher than P.
>
> **[PA3.2.RL.4]** If the necessary skills and competencies are not available in time, then GP 3.2.2 shall not be rated F.

Rationale: If a qualification measure is only planned for the future but the qualification is required today, the qualification is still missing.

Ensuring the availability, allocation and use of the project staff and related information includes that:

- In addition to GP 2.1.4, the resources need to be available according to the roles and qualifications defined in the defined process roles.
- The availability of the resources needs to be secured, taking into account that resources may be also used by other projects of the organization.
- Related information about human resources should be available and include expert knowledge from previous projects and/or training materials.

[PA3.2.RL.5] If the availability and usage of the human resources is not monitored, then GP 3.2.2 shall not be rated F.

[PA3.2.RL.6] If the availability and usage of the human resources for process improvement is not measured and monitored, then GP 3.2.2 shall not be downrated.

5.2.2.3 Ensure required resources to support the performance of the defined process (GP 3.2.3)

The required physical and material resources, infrastructure and work environment according to the standard process and the project specific definition shall be available.

Organizational support to effectively manage and maintain the resources, infrastructure, and work environment for performing the defined process shall be available and known by the project members, such as:

- Resources for the support are planned by the organization.
- Availability of licenses is checked regularly.
- Information about anticipated or planned infrastructure changes, e.g., new tool chains, shall be made available to the projects.

Infrastructure and work environment shall be used and maintained. If updates or new versions of the work environment are available, the handling must be planned in coordination with the project.

Rating rules:

> **[PA3.2.RL.7]** If the organizational support is not adequate to effectively manage and maintain the resources, then GP 3.2.3 shall not be rated F.

The availability and usage of the resources are measured and monitored.

> **[PA3.2.RL.8]** If the availability and usage of the resources is not monitored, then GP 3.2.3 shall not be rated F.

5.2.2.4 Monitor the performance of the defined process (GP 3.2.4)

The performance of the defined process should ensure that:

- Information required to understand the behavior and evaluate the effectiveness and adequacy of the defined process are identified based on the definitions in GP 3.1.4. Information about process performance may be qualitative or quantitative.
- Information is collected and analyzed to understand the behavior of the process, and to evaluate the effectiveness and adequacy of the defined process. The frequency and approach for collecting and analyzing are defined.
- Results of the evaluation are used to identify where continual improvement of the standard and/or defined process can be made. Results should be documented and made available to all affected parties.

Rating rules:

> **[PA3.2.RL.9]** If the evaluation of the effectiveness and adequacy of the defined process is not made available to all affected parties, then GP 3.2.4 shall not be rated F.

5.2.3 Rating consistency within the process attribute

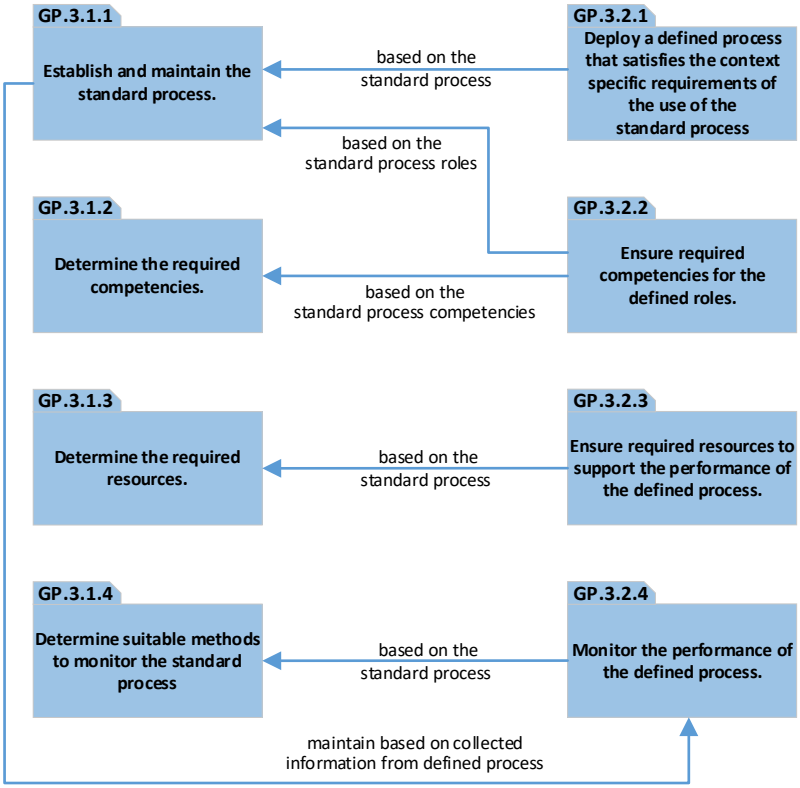
No explicit consistency dependencies were identified within this process attribute, and thus no corresponding rating rules were defined.

5.3 Rating consistency

5.3.1 Rating rules within capability level 3

PA 3.1 defines the context of standard processes. PA 3.2 is about successfully deploying what has been defined within PA 3.1. Based on this, the following philosophy is followed in this document: processes can only be deployed to the extent that they are defined. This means the rating of GP 3.2.x cannot be higher than the rating of GP 3.1.x.

The following figure shows relationships between generic level 3 practices:



This also implies the following general scenario:

PA 3.1 is not rated high because there are gaps. The individual project realizes this and fills in these gaps in a project-specific way. In such a case, still PA 3.2 cannot be rated higher than PA 3.1. However, PA 2.1 or PA 2.2, respectively, may be rated high in that project. Recall that project-specific solutions are addressed at capability level 2 while capability level 3 addresses the organization across projects.

GP 3.1.1 Establish and maintain the standard process

> **[PA3.1.RL.25]** If the indicator 3.2.4 is downrated due to missing or inadequate information from monitoring the performance of the process, then GP 3.1.1 shall not be rated F.

GP 3.2.1 Deploy a defined process that satisfies the context specific requirements of the use of the standard process.

> **[PA.3.2.RL.10]** If the indicator GP 3.1.1 is downrated due to missing or inadequate definition of the standard process, the indicator GP 3.2.1 shall be downrated.

GP 3.2.2 Ensure required competencies for the defined roles.

> **[PA3.2.RL.11]** If the indicator GP 3.1.1 is downrated due to missing or inadequate definitions of roles, responsibilities and authorities, the indicator GP 3.2.2 shall be downrated.
>
> **[PA3.2.RL.12]** If the indicator GP 3.1.2 is downrated due to missing or inadequate definitions of competencies, skills or experience, the indicator GP 3.2.2 shall be downrated.

GP 3.2.3 Ensure required resources to support the performance of the defined process.

> **[PA3.2.RL.13]** If the indicator GP 3.1.3 is downrated due to missing or inadequate definitions of resources, the indicator GP 3.2.3 shall be downrated.

GP 3.2.4 Monitor the performance of the defined process.

> **[PA3.2.RL.14]** If collecting and analyzing the required information is not performed according to the defined methods

and activities (GP 3.1.4), the indicator GP 3.2.4 shall be downrated.

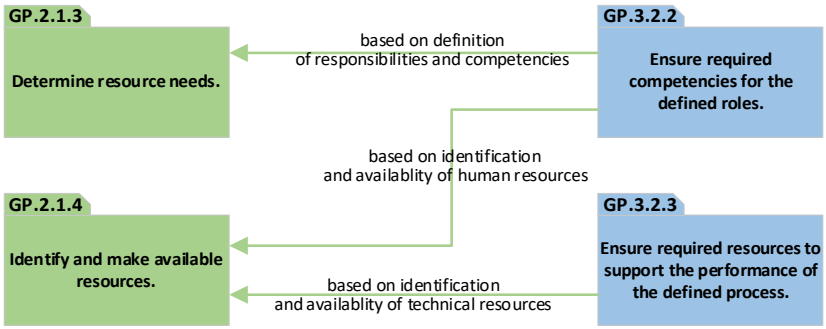
5.3.2 Rating rules between capability level 2 and 3

Process attribute 3.1 Process Definition is one of the few process attributes that does not depend on the lower process attributes.

The rationale is that whether the lower process attributes are performed well or poorly may or may not affect the definition of the standard process.

However, for a capability level 3, the standard process must cover all aspects of capability level 1 and 2 and a feedback mechanism to regularly check and improve the standard process itself. Therefore, the rating of the process attribute PA 3.1 is independent of the project.

The following picture shows the dependencies of PA 3.2 to level 2:



GP 3.2.2 Ensure required competencies for the defined roles

[PA3.2.RL.15] If GP 2.1.3 is downrated due to missing or inadequate determination of responsibilities, authorities, knowledge or skills, the indicator GP 3.2.2 shall be downrated.

Rationale: If there is a weakness on GP 2.1.3 regarding definition of the responsibilities and authorities, this weakness could be evident in two possible scenarios on level 3:

- The weakness is also found in the GP 3.1.2, the identification of roles, competencies, etc., which in turn would lead to a weakness in the project using this standard process (GP 3.2.2.).
- The standard process is rated F regarding GP 3.1.2, but the project does not use the process properly otherwise, the GP 2.1.3 would not be downrated.

[PA3.2.RL.16] If the identification, allocation and availability of resources (GP 2.1.4) is downrated due to human resources issues, the indicator GP 3.2.2 shall be downrated.

[PA3.2.RL.17] If GP 2.1.4 is downrated due to missing or inadequate qualification of individuals, then GP 3.2.2 shall be downrated.

GP 3.2.3 Ensure required resources to support the performance of the defined process.

[PA3.2.RL.18] If the identification, allocation and availability of resources (GP 2.1.4) is downrated due to physical or material resource issues, then GP 3.2.3 shall be downrated.

Rationale: If there is a weakness on GP 2.1.4 regarding identification and availability of the resources, especially technical resources, this weakness could be evident in two possible scenarios on level 3:

- The weakness is also found in the GP 3.1.3, the determination of resources (human, material, process infrastructure), which in turn would lead to a weakness in the project using this standard process (GP 3.2.3.).
- The standard process is rated F regarding GP 3.1.3, but the project does not use the process properly otherwise, the GP 2.1.4 would not be downrated.

6 Understanding Capability Level 4

Based on the enterprise's business goals, the management derives information needs (GP 4.1.1 and GP 4.1.2). For those information needs metrics, or a set of combined metrics, are defined (GP 4.1.4). Furthermore, target thresholds (control limits, see GP 4.1.5) are determined proactively, indicating which quantitative data is considered satisfactory and which is not.

Now, for each process instance quantitative data is measured according to those metrics and recorded (GP 4.1.6). The evolving data history is analyzed using reasonable statistical methods to see if the target thresholds (control limits) are exceeded (GP 4.1.1); note that, in practice, the data history can also be used as the basis for determining the control limits (i.e., observing and understanding "what is normal").

If a particular process instance is identified as violating the control limits (GP 4.2.1) then a causal analysis is performed for that process instance to understand the "special cause of variation" (GP 4.2.2). Such special causes of variation are, by definition, specific to one individual process instance. Once understood, process instance-specific measures are taken to meet the control limits again (GP 4.2.3).

In practice, it is usually considered unfeasible to manually collect and combine quantitative data. Therefore, such data should be extractable from repositories, tools, or given reports. Also, it must be specified who takes the responsibility for analyzing and delivering the data to whom.

Note that capability level 4 requires a full capability level 3 as its basis. In other words, achieving capability level 3 makes it possible to compare quantitative process performance data across process instances (e.g., projects, organizational units) for the Automotive SPICE® process under consideration. Otherwise, such data would not be comparable per se.

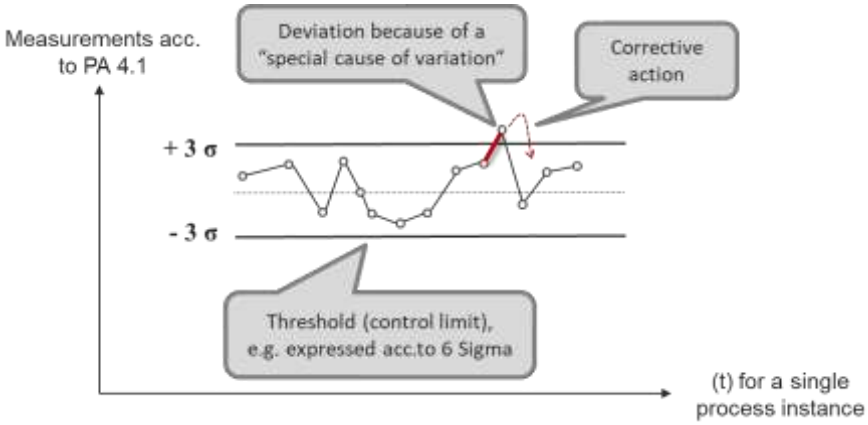


Figure 6-1: Understanding of capability level 4 for one particular process instance

Capability level 4 nor 5 do not predefine which statistical analyses or methods should be used, or whether a process mean is to be reduced or raised. The reason is that Automotive SPICE® is at the WHAT level (see Automotive SPICE® PAM subchapter 3.3).

7 Understanding Capability Level 5

Capability level 4 identifies “special causes of variations” from control limits (thresholds) in individual process instances. In addition, capability level 5 identifies “common causes of variation” across all process instances. At the same time, the control limits (thresholds), within which quantitative process performance data is expected to remain, are made narrower.

When common causes of variation are identified, the standard processes (since capability level 3) aim to reduce common causes of variation. in other words, the standard processes are improved, and the quantitative process performance data gathered by capability level 4 is observed to see if data variance across process instances has actually been reduced. If not, the approach is iterated. The exact statistical methods to use, and the desired thresholds (control limits) optimization will depend on the process information needs (see GP 4.1.2), and (combination of) metric(s) chosen (see GP 4.1.4).

Additionally, capability level 5 considers adopting meaningful state-of-the-art approaches and industry best practices regarding both product and process development.

Capability level 4 nor 5 do not predefine which statistical analyses or methods are to be used, or whether a process mean is to be reduced or raised. The reason is that Automotive SPICE® is at the WHAT level (see Automotive SPICE® PAM subchapter 3.3).

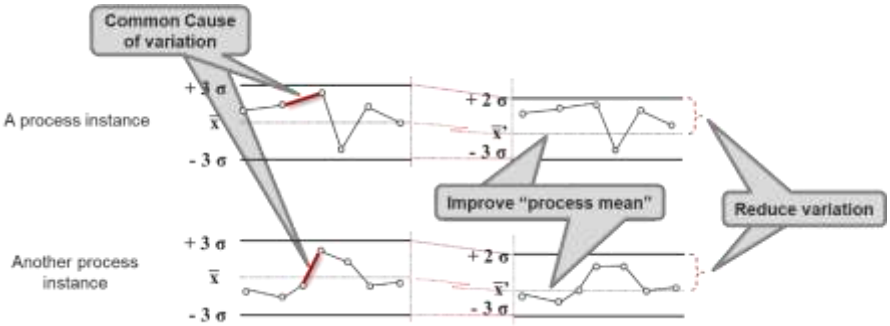
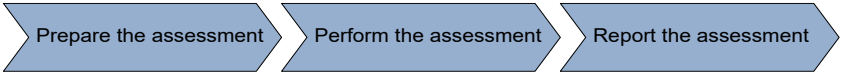


Figure 7-1: Understanding of capability level 5 on reducing variation across process instances based on common causes

Part 2: Guidelines for Performing the Assessment

The purpose of part two of the current publication is to support the assessors in performing an assessment based on the Automotive SPICE® process reference and assessment model, considering the requirements of ISO/IEC 33002 *[ISO33002]*.

Chapter 8 *("Documented assessment process"*) provides necessary input for performing the assessment defined in ISO/IEC 33002 *[ISO33002]*. It provides the tasks and activities of the so-called evaluation phase, in which the assessment is planned, prepared, performed and documented.



In Chapter 9 (*"Improvement process"*), an overview of the tasks and activities are given, for the case that the assessment results are to serve as an input for subsequent improvement measures. In this so-called improvement phase, the assessment results of the evaluation phase are used to plan, execute and track the process improvement actions.

Chapter 10 (*"Recommendations for performing an assessment")* provides additional requirements when applying the documented assessment process.

In Chapter 11 (*"Requirements relating to assessor qualification")* the requirements for assessors to demonstrate the competencies to conduct an assessment and to monitor and verify the conformance of a process assessment are given.

8 Documented Assessment Process

8.1 Introduction

This chapter provides a documented assessment process (DAP) according to ISO/IEC 33002 *[ISO33002]*, clause 4.1:

> *The assessment shall be conducted according to a documented assessment process. The documented assessment process shall be capable of meeting the assessment purpose and be structured in a manner that ensures that the purpose for performing the assessment is satisfied, in terms of the rigor and independence of the assessment and its suitability for the intended use.*

The documented assessment process provided was set up to serve most assessments within the automotive domain. It fulfills the requirements of ISO/IEC 33002 *[ISO33002]* under the following preconditions:

- The assessment is using the PRM and PAM Automotive SPICE® 4.1 and subsequent versions.
- The assessment is using the process measurement framework defined in Automotive SPICE® 4.1 which is an adaptation of ISO/IEC 33020:2019 "*Process measurement framework for assessment of process capability" [ISO33020].*
- A defined rating and aggregation method according to ISO/IEC 33020:2019 is used *[ISO33020]*.
- The assessment is classified as "Class 3" according to ISO/IEC 33002 *[ISO33002],* clause 4.6.
- The category of independence of the body performing the assessment, the lead assessor and the other members of the assessment team is A, B or C according to ISO/IEC 33002, Annex A *[ISO33002].*
- The assessment is not intended to evaluate organizational maturity.

It is the responsibility of the lead assessor to evaluate whether the assessment provides the given preconditions. In case of deviations, the lead assessor shall take appropriate steps to modify this given

DAP or select another suitable one. In this case the lead assessor takes responsibility for the conformity of the DAP to ISO/IEC 33002 *[ISO33002]*.

8.2 Assessment input and output

8.2.1 Assessment plan

According to ISO/IEC 33002 *[ISO33002]*, an assessment plan shall be set up. According to this DAP, the assessment plan shall contain the following elements:

- required inputs specified in this standard → 8.2.2
- definition of the class of assessment and the category of independence of the body performing the assessment, the lead assessor and the other members of the assessment team → 8.1
- communications to the personnel involved in the assessment → 8.3
- identification of the documented assessment process including:
  - the strategy and techniques for the selection, identification, collection and analysis of objective evidence and data, to satisfy any requirements for coverage of the process scope of the assessment as defined for class 3 assessments → 8.4.1
  - the approach to derive an agreed process attribute rating, where relevant → 8.4.1 and Part 1
  - activities to be performed in the assessment → 8.4
  - resources and schedule assigned to these activities → 8.4
  - identification and definition of roles and responsibilities of the participants in the assessment → 8.3
  - criteria to verify that the requirements of ISO/IEC 33002 *[ISO33002]* are met → 8.1
  - description of the planned assessment outputs → 10.4

8.2.2 Assessment inputs

According to ISO/IEC 33002 *[ISO33002]*, the necessary assessment input shall be identified. According to this DAP the necessary input shall contain as a minimum the following elements:

- identity of the sponsor and the sponsor's relationship to the organizational unit(s) being assessed
- business context, including the organization's/sponsors business goals and circumstances of the assessment
- purpose of the assessment, e.g., process improvement or evaluation of the process capability assigned to a specific product delivery
- the application domain (e.g., system development, software development, hardware development) of the products or services of each organizational unit
- assessment scope as it applies to the business comprising a defined and declared organization scope, including:
  - the processes to be examined within the assessment
  - the process quality characteristic to be studied, including the highest process capability level for each individual process within the assessment scope
  - the organizational unit(s) that deploy the process
  - the boundaries of the assessed organization, including
    - the size of each organizational unit, e.g., number of personnel
    - key characteristics (e.g., size, criticality, quality) of the products or services of each organizational unit
  - the process context including the set of stakeholder requirements and changes which are under investigation
  - the process instances, which have been selected, if applicable
- identity of the model(s) and process measurement framework used:
  - Automotive SPICE® 4.1 or higher
  - Automotive SPICE® 4.1 process measurement framework

- assessment requirements, including:
  - reference to this documented assessment process
  - definition of the class of assessment and the category of independence of the body performing the assessment the lead assessor and the other members of the assessment team
  - rating method(s) to be employed
  - aggregation method(s) to be employed
- assessment constraints considering, at minimum:
  - availability of key resources
  - maximum duration of the assessment
  - specific processes or organizational units to be excluded from the assessment
- ownership of the assessment outputs and any restrictions on their use
- controls for handling confidential information and non-disclosure
- participants and their roles, the assessment team and assessment support staff with specific responsibilities for the assessment
- criteria for competence of the lead assessor

8.2.3 Assessment report

The assessment report shall give appropriate guidance to improve the process and to remove the process-related product risks. The requirements and recommendations for the assessment report are defined in detail in subchapter 10.4.

8.2.4 Objective evidence gathered

For evaluating the processes within the assessment scope objective evidence and additional information shall be collected. Each evidence shall be traceable to associated assessment indicators (base practices, information items, generic practices, etc.).

8.3 Parties and roles involved in the assessment

The main parties involved in the assessment are the sponsor, the assessing organization, and the assessed organization. The following roles shall be identified:

LAC: Local Assessment Coordinator

Individual, who takes responsibility for the organization of the assessment within the organizational unit assessed.

SP: Sponsor

Individual or entity, internal or external to the organizational unit to be assessed, who requires the assessment to be performed, and provides financial or other resources to carry it out (see ISO/IEC 33001 *[ISO33001],* clause 3.2.9).

AS: Co-Assessor

Individual who participates in the rating of process attributes (see ISO/IEC 33001 *[ISO33001],* clause 3.2.11). Assessors have appropriate education, training and both capability assessment experience and domain experience to perform the required class of assessment and make professional judgments (see ISO/IEC 33001 *[ISO33001]*, clause 3.2.11).

LA: Lead Assessor or Assessment Team Leader

Assessor who has demonstrated the competencies to conduct an assessment and to monitor and verify the conformity of a process assessment (see ISO/IEC 33001 *[ISO33001]*, clause 3.2.12).

PP: Participant

Individual from the organizational unit to be assessed, who takes part in the assessment.

Note: While the role definitions provided above are considered to represent the standard approach to responsibility distribution, it is possible that individual assessments may extend or reduce these role definitions as is appropriate for a given assessment. For example, the SP may be knowledgeable of process assessment and

may therefore participate in the detailed aspects of the assessment. The LAC may also be capable of performing a greater role in the process assessment, depending on their knowledge and training with respect to process assessment.

For the description of the responsibilities the following abbreviations are used:

R: Responsible

Those who do the work to execute the task. There is at least one role with a participation type of responsible, although others can be delegated to assist in the work required (see also RACI below for separately identifying those who participate in a consulting role).

A: Accountable (also approver or final approving authority)

The one ultimately answerable for the correct and thorough completion of the deliverable or task, and the one who delegates the work to those responsible. In other words, an accountable must sign off (approve) work that responsible provides. There must be only one accountable specified for each task or deliverable.

C: Consulted (sometimes counsel)

Those whose opinions are sought, typically subject matter experts, and with whom there is two-way communication.

I: Informed

Those who are kept up to date on progress, often only on completion of the task or deliverable, and with whom there is just one-way communication.

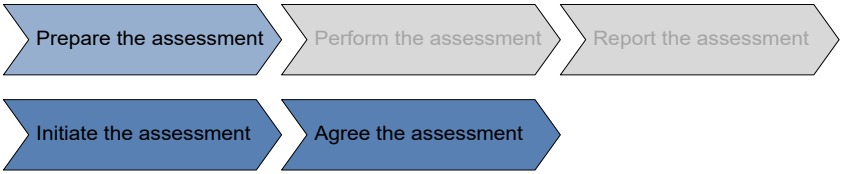
8.4 Assessment activities

The assessment process consists of three tasks:

- prepare the assessment
- perform the assessment
- report the assessment

8.4.1 Prepare the assessment

The preparation for an assessment is split into two sub-tasks:



8.4.1.1 Initiate the assessment

In the initialization phase the sponsor determines the need for an assessment and determines the framework conditions (scope, time period, team, etc.). All necessary information on the assessed organization is collected.

Brief description	The need for an assessment is determined and the framework conditions for its execution are established.
Process inputs	• formal or informal assessment inquiry • information about the organization assessed • previous audit reports and assessment reports
Process outputs	• assessment purpose • assessment agreement • assessment scope • time frame • contact persons in both organizations • assessment team list • assessment plan

Activities / Responsibilities	*LA*	*AS*	*SP*	*LAC*	*PP*
Determine the need for an assessment.	-	-	A, R	-	-
Establish the assessment agreement.	C	C	A, R	C	-
Define the assessment scope.	C, R	I	A	C	-
Collect and evaluate information on the organization assessed.	A, R	C	-	C	-
Define the assessment team.	A, R	C	-	C	-

Determine the need for an assessment

The need for an assessment must be determined by the sponsor. This may be derived based on different use cases and defines the purpose of the assessment. The purpose of the assessment is the base input for setting up the assessment scope.

Establish the assessment agreement

The assessment agreement is established based on the assessment purpose by

- defining the main focus of the assessment. This may be, for example, project management, engineering aspects or other areas of risk. If appropriate, a pre-selection should be made of the processes to be checked;
- determining the assessing organization responsible for performing the assessment;
- selecting the lead assessor and the assessment team members;
- defining the timeframe, within which the assessment should be carried out, and
- identifying the business divisions or departments and personnel in the organization assessed that are to be involved.

Define the assessment scope

The boundaries of the assessment, provided as part of the assessment input, encompassing

- the boundaries of the organizational unit for the assessment,

- the processes to be included,
- the capability level for each process to be assessed, and
- the context within which the processes operate,

are defined.

Collecting and evaluating information on the organization assessed

The information on the organization assessed relevant to the assessment must be collected and evaluated. This may include:

- organizational structure of all those involved in the project, such as
  - sponsor,
  - project team,
  - core/platform development,
  - (independent) quality assurance department,
  - (independent) test department or
  - sub-suppliers.
- standard software components/off the shelf items.
- if appropriate, the department responsible for the selection, release and maintenance of tools or the IT department, for example for configuration management.
- results of other audits and assessments.

  Note: Results from previous audits and assessments can be used for determining the assessment scope. Here, the time that has passed since the audit or assessment and whether the results are applicable for the project (assessment method, assessed department, personnel involved) must be considered.

Define the assessment team

The assessment team is determined and appointed.

8.4.1.2 Agree on the assessment

The exact terms of the assessment are agreed upon between the involved parties.

Brief description	The assessment and its framework conditions are agreed to.
Process inputs	• assessment scope • time frame • assessment team list • assessment plan
Process outputs	• non-disclosure agreement (NDA) • assessment schedule • list of documents to be exchanged in advance • requirements for the evidence repository • distribution list for the report • optional: minutes of the pre-assessment meeting

Activities\Responsibilities	***LA***	***AS***	***SP***	***LAC***	***PP***
Agree on the details of how the assessment shall be performed.	R	I	A	C	-
Conduct pre-assessment meeting (optional).	A, R	C	-	C	I

Agree on the details of how the assessment shall be performed

With the assessment agreement, a consensus regarding the assessment should be achieved by defining details of how the assessment shall be performed and agreed to between the parties.

It is essential that the sponsor, the assessing organization, and the organization assessed agree on the modalities of the assessment. The agreement can be reached formally by means of a contract and acknowledgement, or in an informal manner. Furthermore, the assessment agreement must consider and specify the following points:

- A non-disclosure agreement (NDA) should be agreed upon by all parties involved (assessing organization, organization assessed and assessors) and signed (if not already done in the project).
- The final schedule is agreed to.

- Contact persons are appointed on both sides for coordination.
- The distribution list for the report is established.
- Requirements relating to the evidence repository for the assessment are established.
- Requirements relating to the infrastructure, e.g., meeting rooms, beamers, printers, flipcharts etc. are established.
- Constraints for the scheduling, e.g., availability due to bank holidays, breaks, local conventions etc. are identified.

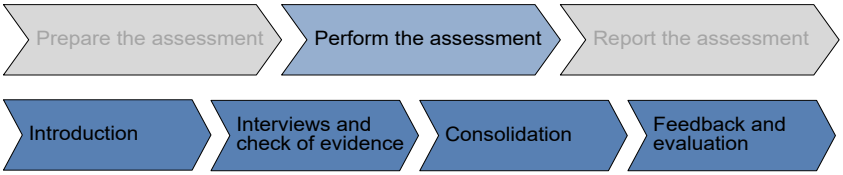
Conduct pre-assessment meeting (optional)

If necessary, a pre-assessment meeting can be carried out (on-site, by email or by a telecommunications conference). The purpose is to

- explain the framework and process of the assessment to the personnel involved;
- specify the set of documents to be handed out to the assessment team in advance for study;
- to understand and confirm the assessment context, and
- to perform preliminary document analysis.

8.4.2 Perform the assessment

The execution of the assessment is split into four tasks:



In the introduction task the assessment scope, the project to be assessed and the assessment method are presented. This is followed by the interviews and document reviews, where the actual collection of evidence is done which is the crucial part of the assessment. Once the collection of evidence has been completed, the consolidation task starts, and the first evaluation of the results (findings) takes place. Finally, in the feedback and evaluation task, the collected results are stored in the evidence repository, the

preliminary process attribute rating results are presented, and possible immediate actions are recommended.

8.4.2.1 Introduction

Brief description	The organization to be assessed, the project, the evaluation methodology and the activities of the assessment are presented.
Process inputs	• information on the organization assessed and the project • assessment scope • assessment schedule • assessment plan
Process outputs	• information of the organization assessed and project • information on Automotive SPICE®, the assessment scope, and the assessment schedule

Activities \ Responsibilities	***LA***	***AS***	***SP***	***LAC***	***PP***
Present the organization assessed and the project.	I	I	-	A, R	C
Present the assessment activities.	A, R	C	I	I	I

The introduction should give all those involved an overview of the organization assessed, the project, the assessment methodology and sequence.

Present the organization assessed and the project

The organization presents itself and the project in the scope to be assessed to the assessment team. The purpose of this activity is to provide the assessment team with an introduction to the project-specific conditions and circumstances.

Present the assessment activities

The assessment team presents the concrete activities of the Automotive SPICE® assessment. The purpose of this activity is to inform the organization assessed and the interviewees about the detailed procedure which will be followed during the assessment (for example, the evidence repository).

8.4.2.2 Interviews and checks of evidence

Brief description	The project-related information regarding the selected processes is collected and documented in accordance with the assessment model.
Process inputs	• assessment schedule • project-related documents and work products • process-related documents (for CL>2)
Process outputs	• assessment notes regarding results of interviews, documents or work products which have been examined and results of the inspection of the work environment • list of documents which have been examined

Activities \ Responsibilities	***LA***	***AS***	***SP***	***LAC***	***PP***
Perform interviews, document checks and inspections of the work environment, if appropriate.	A, R	C	-	C	C
Collect evidence for rating the processes.	A, R	C	-	C	C

Evidence relevant to the project in terms of the selected processes is collected and documented.

Perform interviews, document checks and inspections of the work environment, if appropriate

Based on the assessment schedule, interviews on the individual processes with the key personnel of the organization assessed are carried out and the associated documents, work products and evidence are examined (using the two-sources-of-evidence-principle here can be meaningful). If necessary, the conditions under which the process is performed can be checked at the workplace.

The results of the interviews are documented in the assessment notes.

Collect evidence for rating the processes

The assessment team collects the evidence to justify and document the findings for the individual processes related to the developed product (e.g., regarding process compliance, the tools used in the project and the quality of existing documents or work products).

8.4.2.3 Consolidation

Brief description	The selected processes are rated by the assessors based on the available evidence.
Process inputs	• assessment notes
Process outputs	• consolidated assessment notes • provisional process capability profiles

Activities \ Responsibilities	***LA***	***AS***	***SP***	***LAC***	***PP***
Evaluate the collected evidence.	A, R	C	-	-	-
Provide a provisional rating.	A, R	C	-	-	-
Document strengths and potential improvements.	A, R	C	-	I	I
Establish the traceability of process attribute rating to evidence.	A, R	C	-	-	-
Document the deviation of rating rules.	A, R	C	I	-	-

The evidence collected from interviews and document reviews is consolidated by the assessors.

Note: The consolidation might also be done incrementally after each interview session, see subchapter 8.4.2.2.

Evaluate the collected evidence

Following the interviews and the document reviews the assessment team consolidates and documents the analysis results and reaches consensus on the identified strengths and potential improvements of the processes which have been assessed.

Provide a provisional rating

Based on the findings the process attributes are rated, and a provisional set of process capability profiles is determined for the assessed processes. The rating is evaluated to ensure the consistency with the rules given in part one of this publication. The rating shall consider the rating rules given in Part 1 of this document.

Document strengths and potential improvements

The findings are evaluated in terms of strengths and potential improvements.

Establish the traceability of process attribute rating to evidence

For each process attribute rating the traceability to the collected evidence used in determining that rating is established. The relationship between the assessment indicators for each process attribute rated and the objective evidence is documented.

Document the deviation of rating rules

The rules not obeyed by the lead assessor are identified. A justification as to why the rule is not applicable or has no significant impact on the process attribute rating is provided.

Note: The purpose of the justification is to briefly document the lead assessor's decision to not follow a specific rule. It is the clear intention of the authors of this publication not to generate additional effort due to extensive documentation of rule deviations. The provision of a list of all rules, no matter whether they are obeyed or not might make sense for inexperienced assessors and might give an overview but is not required or intended by the authors of this publication.

8.4.2.4 Feedback and evaluation

Brief description	A provisional evaluation of the organization assessed is presented and immediate actions are identified.
Process inputs	• provisional process capability profiles • list of documents which have been examined • consolidated assessment notes
Process outputs	• provisional process attribute ratings and the process capability profiles • list of the most important findings (strengths and potential improvements) • document archive related to the assessment • list of immediate actions, if applicable

Activities \ Responsibilities	***LA***	***AS***	***SP***	***LAC***	***PP***
Present the results.	A, R	C	I	I	I
Identify immediate actions. (optional)	C	C	-	A, R	C
Store the evidence in the repository.	I	-	-	A, R	I

The purpose of feedback is to provide information on the assessment results and to reach a common understanding of the rating.

The feedback shall contain at least the following:

- the provisional process attribute ratings
- the provisional process capability profiles
- the major strengths and potential improvements (for each process assessed)

The feedback should be provided directly following the conclusion of all interviews. The contents of the feedback should be documented in writing as a feedback presentation and afterwards made available as a copy to the assessed party.

Present the results

If possible, immediately after the assessment the provisional process attribute ratings and the capability profiles are prepared and presented to the organization assessed. The most important findings (strengths and potential improvements) are presented.

Identify immediate actions (optional)

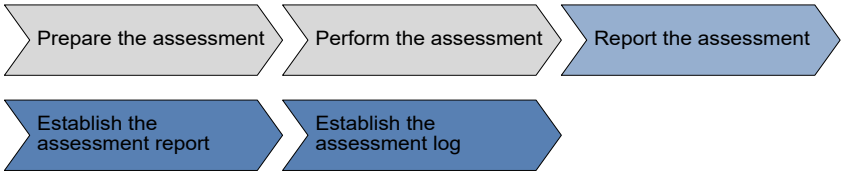
Based on the presented identified potential improvements, immediate actions are recommended to eliminate critical weaknesses.

Store the evidence in the repository

The organization assessed stores the evidence repository including references to the documents that have been analyzed.

8.4.3 Report the assessment

The elaboration and distribution of the report following an assessment is split into two tasks:



The detailed assessment report is drawn up to document the results of the assessment. The assessment log is prepared and submitted to the certification body.

8.4.3.1 Establish the assessment report

Brief description	The assessment team compiles the assessment report to be distributed within four calendar weeks in the assessed organization.
Process inputs	• consolidated assessment notes • provisional process capability profiles • list of the most important findings.

Process outputs	• assessment report with the process attribute ratings and the final process capability profiles • an explanation of deviations at the practice level (optional)

Activities \ Responsibilities	***LA***	***AS***	***SP***	***LAC***	***PP***
Consolidate the final process attribute ratings and the final process capability profiles.	A, R	C	-	-	-
Compile the assessment report.	A, R	C	I	I	I
Distribute the assessment report.	-	-	A, R	C	I

Consolidate the final process attribute ratings and the final process capability profiles

The set of final process capability profiles is drawn up. The consolidated findings and observations are documented in detail based on the assessment notes.

Compile the assessment report

The assessment report must be compiled, checked and released by the assessment team. The lead assessor is responsible for drawing up and releasing the assessment report. Deviations from rating rules given in Part 1 of this publication shall be documented in the assessment report. Normally, the assessment report is provided within four calendar weeks to the assessment sponsor for distribution in the assessed organization. Please refer to subchapter 10.4 for detailed requirements on the assessment report.

Distribute the assessment report

The released version is distributed within the assessed organization.

8.4.3.2 Establish the assessment log

Brief description	The assessment team draws up the assessment log.
Process inputs	• template for the assessment log
Process outputs	• the assessment log

Activities \ Responsibilities	***LA***	***AS***	***SP***	***LAC***	***PP***
Issue the assessment log.	R	C	A	C	-

Issue the assessment log

The assessment log serves as the official confirmation from the sponsor, the LAC, and the assessment team that the assessment was conducted according to the defined assessment process.

The assessment log shall be signed by the lead assessor and the assessment team members. The log shall be approved by the sponsor.

The assessment log shall be drawn up based on the template provided by the certification scheme (see Chapter 11, *"Requirements relating to assessor qualification"*).

9 Improvement Process

9.1 Introduction

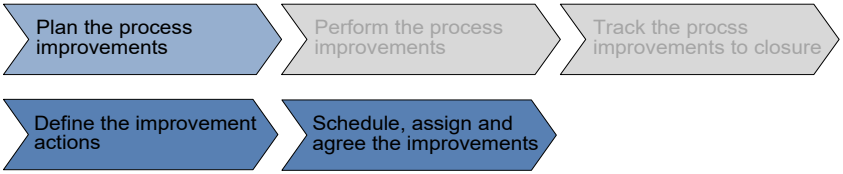
The process improvement phase may follow the evaluation phase and is split into the planning on the process improvement actions, and performing and tracking these actions.

Since the improvement actions will in general not be assigned to the roles involved in the evaluation phase, no detailed assignment of responsibilities is given in this chapter, albeit the responsibility for planning, performing and tracking all improvement actions lies with the assessed organization.

9.2 Improvement activities

9.2.1 Plan the improvements

The process and product improvement actions are established, together with the monitoring criteria, responsibilities and the time schedule.



9.2.1.1 Define the improvement actions

Brief description	The process improvement actions to be carried out are selected and prioritized.
Process inputs	• assessment report • list of immediate actions, if applicable
Process outputs	• list of process improvement actions • monitoring criteria for process improvement actions

Activities
Specify the process improvement actions.
Prioritize the process improvement actions.
Define the monitoring criteria.

Specify the improvement actions

A list of process improvement actions is established including the desired improvement result based on the assessment report. A traceability to the identified assessment findings is provided, if applicable.

Prioritize the improvement actions

Prioritization is performed based on an evaluation of the effectiveness of the improvement actions.

Define the monitoring criteria

Based on the list of process improvement actions monitoring criteria are defined which allow checking whether the implementation of the actions have the desired effects.

9.2.1.2 Schedule, assign and agree the improvements

Brief description	The improvements are scheduled, assigned and a commitment on the improvements is achieved.
Process inputs	• list of process improvement actions
Process outputs	• responsibilities for process improvement actions • time schedule for process improvement actions
Activities	
Define the responsibilities.	
Define the time schedule for implementation.	
Agree on the improvement actions.	

Define the responsibilities

The improvement actions are assigned to persons who are responsible for their implementation.

Define the time schedule for implementation

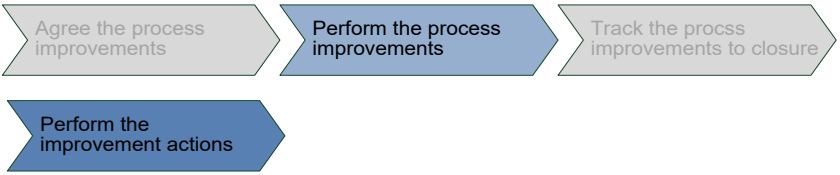
Dates and priorities are assigned to the individual process improvement actions. Based on a risk assessment, the actions from the list that are to be implemented in the project and/or in the assessed organization are identified.

Agree on the improvement actions

An agreement on the improvements is achieved from all affected parties.

9.2.2 Perform the improvements

Immediate actions should be carried out directly after the assessment. Other process improvement actions are implemented according to the defined schedule.



9.2.2.1 Performing improvement actions

Brief description	The process improvement actions are carried out.
Process inputs	• list of process improvement actions • responsibilities for process improvement actions • time schedule for process improvement actions.

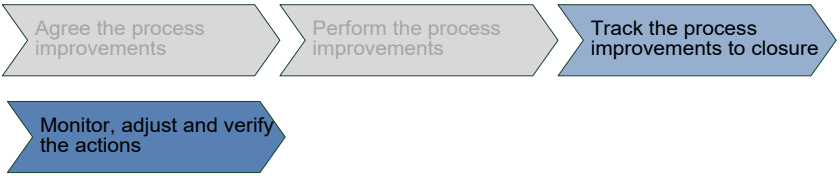
Process outputs	• documentation of the improvements which have been carried out
Activities	
Execute the process improvement actions.	

Execute the improvement actions

The process improvement actions should be carried out in due time by those responsible and according to priority.

9.2.3 Track the improvement to closure

Tracking the process improvement actions represents the completion of the improvement process:



The process improvement actions are monitored, and any necessary adjustments are made taking risks into account.

9.2.3.1 Monitor, adjust and verify the actions

Brief description	The actions are monitored and adjusted if necessary.
Process inputs	• list of process improvement actions • monitoring criteria for process improvement actions • documentation of the improvements which have been carried out
Process outputs	• status report of the process improvement actions • road map for long term actions exceeding the project scope

Activities
Monitor the process improvement actions.
Modify improvement actions if deficiencies are detected.
Verify and close improvement actions.
Plan long term actions exceeding the project scope.

Monitor the improvement actions

Based on the defined monitoring criteria the process improvement actions are checked regularly regarding their implementation and effectiveness.

Modify improvement actions if deficiencies are detected

If the actions do not achieve the desired effect, modified or new actions are specified.

Verify and close improvement actions

The improvement actions are closed if they achieved their purpose.

Plan long term actions exceeding the project scope

Long-term actions exceeding the project scope should be addressed within a road map.

10 Recommendations for Performing an Assessment

In the current chapter recommendations are provided, which should be considered when following the documented assessment process specified in Chapter 8.

10.1 Assessment results

10.1.1 Confidentiality of information

As a fundamental rule, assessment results and the information obtained during an assessment must be treated as confidential by all persons and organizations involved.

10.1.2 Handling the assessment results

The ownership of the assessment results is defined in the initial assessment agreement (see 8.4.1.1); by default, the Sponsor is the owner of the results.

If the assessment results are issued to third parties, an additional non-disclosure agreement should be signed where appropriate to maintain the confidentiality.

The assessment results and any relevant part of them should be made available within normally four calendar weeks after the assessment to all individuals involved in the assessed project and those involved in the performance and monitoring of the improvement actions. The criterion here is their involvement in the project or process development.

The assessment results should be documented and archived by the assessing organization.

10.2 Validity of assessments

10.2.1 Area of validity of the assessment results

Automotive SPICE® is predominantly used to assess single projects based on a given scope. In these assessments the focus is always on one particular project. Neither the complete set of all projects in an organization nor a statistically significant selection is scrutinized. It follows therefore that assessment results are a representative sample of the process capability within the scope of the assessment, but not applicable in general to the assessed organization as whole, the development location or the entire company.

The assessment results may be considered to reflect potential capability of another project with identical characteristics. Here the following criteria should be considered:

- Development locations: As a rule, assessment results are not transferable from one location to another.
- ECU domains: If at a large development location ECUs are developed for various domains, such as powertrain, chassis or body, assessment results are transferable only to a limited degree, given the different development environments.
- Distributed development: Where the development work on ECUs is distributed over several departments or several locations, the assessment results apply only to those locations or departments which have been assessed.

The extent to which assessment results can be transferred will depend on various factors – including the process capability level – and must be evaluated on a case-by-case basis.

10.2.2 Period of validity of assessment results

Assessment results have only a limited validity in terms of time. Experience has shown that they allow reliable conclusions to be drawn for 12 months regarding the project which has been assessed.

Changes within the project, such as, for example,

- the transfer of the development work to a different location,
- a re-organization in the organization which has been assessed or
- changes to the development processes

can, however, significantly affect the relevance of the assessment results to individual processes even within 12 months. Such changes may cause the actual capability of the development process to be better or worse than indicated by the last assessment result.

On the other hand, where there is a high degree of project stability, the assessment results may permit reliable conclusions regarding the project to be drawn for longer than 12 months. For these reasons, the period of validity must always be considered relative to the specific project circumstances.

10.3 Performing an assessment

The following recommendations should be observed when performing assessments:

10.3.1 General

The assessment team leader has the authority, and the responsibility, to take any necessary precautions and actions to ensure that the assessment is conducted in compliance with the relevant ISO/IEC 330xx parts, the Automotive SPICE® 4.0 measurement framework and this document. This includes the right to dismiss individuals (assessment team or interviewees), or to cancel interviews.

10.3.2 Assessment scheduling

When planning the assessment, at a minimum the following conditions should be considered:

- the scope of the assessment, specifically the number of assessed processes, the number of process instances and the highest assessed level
- the process context as defined in subchapter 1.3.4
- the complexity of the assessed project, e.g., in terms of distributed developments, size of the assessment scope, complexity of the developed product
- results and experiences from previous assessments
- assessment experience of the assessed party
- problems associated with different cultures and languages (e.g., organizing cultural trainings or a translator)

Based on this, sufficient interview and consolidation time frames should be planned.

There should be at least four weeks between agreement on an assessment and its execution.

In some cases, it could be appropriate to perform interviews for data collection only using phone and/or video conferences.

10.3.3 Individuals involved in the assessment

The assessing organization performing the assessment decides on the composition of the assessment team in consultation with the sponsor (see also 10.3.4).

Participation by observers or other guests in interviews:

- In principle, observers can be present at an interview – e.g., observers from the process development department.
- The number of people taking part in the interview should be kept as small as possible.
- The interviews must not be impaired by observers, whether active or passive.
- The assessment team leader decides whether observers may be present at the interviews and can exclude observers (in general or particular individuals) even during the assessment.

10.3.4 Composition of the assessment team

The assessment shall be carried out by at least two assessors. The lead assessor must have at minimum a valid competent assessor certification.

Independence of the assessors should be ensured to avoid any conflict of interest.

The assessment team leader must ensure that there is sufficient competence on the team regarding the PAM used and technical domain knowledge (e.g., hardware development in the case of assessing the HWE.x processes, machine learning in the case of assessing MLE.x processes).

The assessment team leader has the final authority for the selection of the assessor(s) and to exclude assessors and participants from the assessment.

10.4 Assessment report

In the assessment report the organization which has been assessed is given more detailed feedback of the strengths and potential improvements detected in the assessment. The assessment report should document especially those points that led to a downrating of the process attribute by referencing the individual indicator like base and generic practices or information item characteristics.

The assessment report should contain the following information:

10.4.1 General information

This subchapter contains general information on the assessment report.

Item	Required information
Unique identifier	• document/Version number or equal
Date of issue	• issue date of the report
Version	• version identification of the report
Issuer	• issuer of the report
Change history	• document change history

10.4.2 Formal information about the assessment

This subchapter contains formal information about the assessment.

Item	Required information
Assessment model	• assessment model and version that has been used (e.g., Automotive SPICE® PAM V4.x)
Assessment period	• the period during which the assessment was carried out
Sponsor	• name of the assessment sponsor

Local assessment coordinator	• name of the responsible coordinator in the assessed organization
Assessment class	• class of the assessment according to ISO/IEC 33002 *[ISO33002]*
Assessment category	• type A, B, C, or D according to ISO/IEC 33002 *[ISO33002],* Annex A

10.4.3 Purpose and scope of the assessment

This subchapter contains information about the assessment scope. Refer also to subchapter 1.3.2 (*"Defining the processes to be included"*).

Item	Required information
Assessment purpose	• see subchapter 1.2
Assessment scope	• see Chapter 1.3
Capability level	• target capability level for each process assessed
Assessed project	• project name / description
Organization	• company name • organizational / business unit • assessed sites • assessed departments

10.4.4 Participants of the assessment

This subchapter contains information about the assessment team, the interview persons and other participants of the assessment.

Item	Required information
Assessment team leader	• name of the assessment team leader • assessor grade (e.g., competent, principal) • license number of the assessment team leader

	• expiration date of the assessor license
Co-assessor(s)	• name of the co-assessor(s) • assessor(s) grade (e.g., provisional, competent, principal) • license number of assessor(s) license(s) • expiration date of the assessor(s) license(s)
Local assessment coordinator	• name of local assessment coordinator
Interviewed persons	• names of interviewed individuals incl. their role in the project or organizational unit • mapping to the processes for which they were interviewed for (project manager, e.g., could be interviewed for more than one process)
Guests (optional)	• names of persons passively attending the interviews without any rights, e.g., observers, assessor candidates… *Note: To gather experience assessor candidates may participate in the process attribute rating but should not be involved in the rating decision.*

10.4.5 Constraints

This subchapter contains information about constraints and possible impacts of these constraints that must be considered to understand the assessment results.

Item	**Required information**
Constraints (if applicable)	E.g., • somebody was not available (e.g., off, sick) • separated development areas have been included via Video/WebEx (no on-site assessment) • disclaimer (e.g., that the assessment results do not allow conclusions pertaining to the complete organization or other departments of the organization that have been not assessed) • confidentiality constraints, e.g., access to evidence or to infrastructure and sites may be subject to legal access rights.

10.4.6 Assessment results

The following table defines the minimum content of an assessment report and expected quality criteria.

Item	Required information
Process Profiles	• acc. to ISO/IEC 330xx
Capability Profile	• acc. to ISO/IEC 330xx
Weaknesses	• A weakness is a context-specific comprehensive and comprehensible explanation of the process risk. It must deliver sufficient understanding for the assessed parties to understand the context-specific process risk as a starting point for deriving improvements. • A weakness must be substantiated by traceable objective evidence gathered during the assessment. Referring to another assessment report, or making general statements, is not applicable. • Consequently, using the text of the respective Automotive SPICE® indicator in an inverted manner does not serve as an acceptable weakness. • Also, a rating rule cannot replace comprehensive weakness statements. Omitting, or neglecting, comprehensive and comprehensible weakness statements in favor of only referring to rating rules is not acceptable and therefore renders the entire Assessment Report invalid. A rating rule may only support a given comprehensive and comprehensible weakness statement and a given rating.
Objective Evidence references	• The objective evidence examined for each process. • Confidentiality policies may apply.
Rating Rules not followed	• A Rating Rule that was not followed in the given assessment context requires a rationale.

The following table describes optional (i.e. not mandatory) content in an assessment report.

Item	Required information
Strengths (optional)	• A strength is a rewarding statement that a particular practice, or process solution, is a particularly well established, efficient, or advanced solution. • Mentioning strengths is motivating and thus can improve an assessed parties' buy-in.
Improvement suggestions (optional)	• Directions or concrete proposals on how to solve weaknesses from the assessor's perspective.

11 Requirements relating to Assessor Qualification

It is essential that Automotive SPICE® assessments are conducted by appropriate and trained specialists. The lead assessor entrusted with the leadership of the assessment, who also accepts responsibility for the result of the evaluation, plays a special role.

The training of assessors shall be carried out by registered training organizations based on a published certification scheme.

The personal certification of assessors will be carried out by a certification body based on a published certification scheme. The certification scheme shall cover the guidance, the rules and the recommendations given within this publication.

Acceptance of valid qualification schemes for assessors is carried out by the quality management board of the VDA QMC. Currently, the intacs scheme is a valid and accepted qualification scheme *[intacs]*.

11.1 Requirements for assessors

According to the definitions provided in ISO/IEC 33001 *[ISO33001]*, clause 3.2.11, the term “assessor” is defined as:

> *An individual who participates in the rating of process attributes.*

A valid personal Automotive SPICE® Provisional, Competent or Principal SPICE Assessor license issued by the VDA QMC is required as evidence for the qualification and experience of any assessor who is member of the assessment team.

11.2 Requirements for lead assessors

According to the definitions provided in ISO/IEC 33001 *[ISO33001]*, clause 3.2.12, the term “lead assessor” is defined as:

> *An assessor who has demonstrated the competencies to conduct an assessment and to monitor and verify the conformance of a process assessment.*

A valid personal Automotive SPICE® competent or principal assessor license or a valid instructor license issued by the VDA QMC covering the entire assessment scope is required as evidence for the qualification and experience of the lead assessor.

11.3 Requirements for non-lead assessors

Pursuant to the definitions provided in ISO/IEC 33001 *[ISO33001]*, clause 3.2.11, the term “assessor” is defined as:

An individual who participates in the rating of process attributes.

A valid personal Automotive SPICE® provisional, competent or principal assessor license or a valid instructor license issued by the VDA QMC is required as proof of the qualification and experience of any other assessor who is member of the assessment team.

Bibliography

[AS41] Automotive SPICE® Process Reference Model, Process Assessment Model, Version 4.1, 2026-04

[INCOSE] INCOSE Guide for Writing Requirements **–** INCOSE International Council on Systems Engineering; https://www.incose.org/

[intacs] International Assessor Certification Scheme, www.intacs.info

[IntAgile] Frank Besemer, Timo Karasch, Dr. Pierre Metz, Joachim Pfeffer, Intacs white paper, Clarifying Myths with Process Maturity Models vs. Agile, Aug 6th, 2014, www.intacs.info

[IREB CPRE] IREB – International Requirements Engineering Board – Foundation Level CPRE, https://www.ireb.org/en/cpre/foundation/

[ISO19011] ISO 19011:2018, Guidelines for auditing management systems

[ISO24765] ISO/IEC/IEEE 24765:2017, Systems and software engineering — Vocabulary, 2017-09

[ISO26262] ISO 26262:2018, Road vehicles – Functional safety, 2018-12

[ISO29148] ISO/IEC/IEEE 29148:2018, Systems and software engineering — Life cycle processes — Requirements engineering, 2018-11

[ISO33001] ISO/IEC 33001:2015, Information technology — Process assessment — Concepts and terminology, 2015-03

[ISO33002] ISO/IEC 33002:2015, Information technology — Process assessment — Requirements for performing process assessment, 2015-03

[ISO33020] ISO/IEC 33020:2019, Information technology — Process assessment — Process measurement framework for assessment of process capability, 2019-11

[Metz2016] Dr. Pierre Metz, Automotive SPICE® - Capability level 2 und 3 in der Praxis, August 2016, dpunkt.verlag, ISBN 978-3-86490-360-1

Quality Management in the Automotive Industry

You can find the current versions of the published VDA volumes on Quality Management in the Automotive Industry (QAI) on the Internet at http://www.vda-qmc.de.

You can also place direct orders on this homepage.

Reference:

Verband der Automobilindustrie e.V. (VDA)

Qualitäts Management Center (QMC)

10117 Berlin, Behrenstr. 35
Phone: +49 (0) 30 89 78 42-235 ; Fax : +49 (0) 30 89 78 42-605
Email: info@vda-qmc.de; Internet: www.vda-qmc.de