

Quality Management for Agile Software Development

Pocket Guide
Version 1.0

Preface

The automotive industry is undergoing a transformation towards greater digitalization of its products and services. This change affects the type of product development and maintenance: While previously phase-oriented product development processes dominated the automotive industry, agile approaches are becoming more important. This increasing agility also calls into question the previous understanding of quality management and requires new answers from quality management in the transition to agile development methods in the automotive industry.

This pocket guide describes the challenges for quality management in an agile environment and gives recommendations for quality management of products developed in an agile manner. The entire spectrum of possible digital products in general, as well as developments typical for the automotive industry, such as safety-critical electronic control units in particular, are considered. It covers the familiar tasks of quality management from defining quality requirements to deriving lessons learned after product release and field observation.

Overview of quality management tasks

Quality management in the automotive industry is facing many challenges in the introduction of agile development methods for digital products. The familiar tasks of quality management are the starting point for discussing these challenges and their solutions:



1. Define and introduce quality requirements

Capture and formulate quality requirements regarding products and methods and steer them into the development process



9



2. Evaluate suppliers

Audit suppliers and check their ability to implement requirements



27



3. Assure maturity level

Verify and validate product and process quality

→ 33



4. Release products

Final check of product and release for delivery

→ 45



5. Observe the field

Record product usage by the customer and complaints

→ 51



6. Derive lessons learned

Record findings from field observation and the development and production process and integrate them into the developments

→ 51



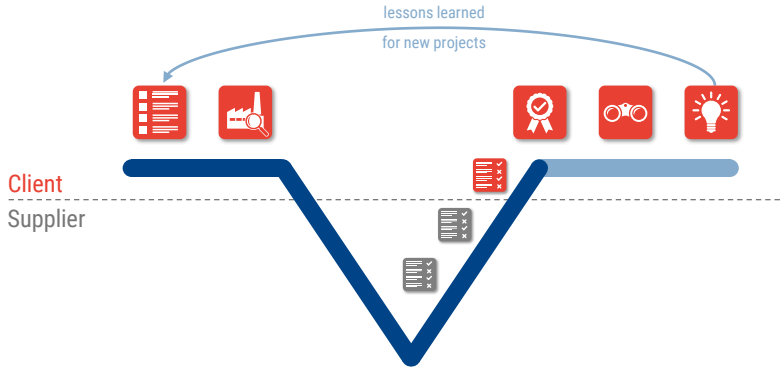
7. Live agile principles

Cultivate agile mindset and agile culture

→ 55

Quality management process transformation

Phase-oriented development
& release process



Agile development & release process





Define and introduce Q-requirements

Product classification and Q-requirement	10
Decision for or against agility	18
Organization	23



Product classification and Q-requirements

Depending on the type of product, different subject-specific quality requirements have to be implemented and organizational and legal constraints and industry standards have to be taken into account.

The first step therefore is to clarify the class to which the product to be developed belongs and the requirements associated with the product class before development begins.



Example **Parking assistance system**

A new parking assistance system that enables automated parking even in more complex parking situations (multi-story parking garage, etc.) is to be developed. This results in a number of quality requirements, such as with regard to the maintainability of such a system by workshops and reliability. Many of these requirements are already known from previous developments of parking assistance systems and can be used again for the new development.

The parking assistance system is an embedded system that communicates with other ECUs in the vehicle via the CAN bus and is based on the AUTOSAR platform and its architecture.

The system must be implemented according to the requirements of Automotive SPICE.

It is also a safety-critical system that must therefore be developed in accordance with ISO 26262. It is assumed here that the hazard and risk analysis ASIL D, that is, the highest safety-critical level, applies.





Example **Charging station search**

An app shall be developed for drivers to find charging poles and obtain further information about the respective charging poles (supplier, costs, availability, etc.).

As this requires neither development, modification nor activation of vehicle ECUs, nor implementation nor adaptation of safety-critical functions, Automotive SPICE and ISO 26262 do not need to be considered.

There are cybersecurity requirements (GPS data of the app user should be transmitted in encrypted form), but these do not exceed the security requirements of similar apps.

The quality management itself has no specific requirements for the app but must ensure the quality of the implemented functional requirements and the maintainability of the app.

The determination of the product type and quality requirements in an agile environment is basically no different from the conventional procedure. However, typically the type of documentation and the transmission of requirements to the development differs. Instead of a conventional requirement description in the form of a requirement specification, the requirements for agile development are included in the product backlog and usually written in the form of user stories.

The requirements should be

- *Independent*
- *Negotiable*
- *Valuable*
- *Estimable*
- *Small*
- *Testable*



Example Parking assistance system

As a garage employee, I can update the software of the parking assistance system in order to keep the control unit up to date.



Requirement specification =



The user stories should be supplemented by acceptance criteria and made more specific in a later dialogue with the development team. It is the task of the Product Owner, in consultation with the requester, to break down the business requirements into fine granular business requirements so that the requirements can be implemented within an iteration (*ready for sprint*).

Quality requirements and boundary conditions can already have been formulated in previous projects and integrated into the development at that time. These requirements are usually already specified in detail. Even if such a documented detail specification of requirements is unusual in an agile environment, it still makes sense to use such existing documents for new agile developments.




Example **Parking assistance system**

Several quality requirements have already been recorded in the DOORS requirements management system in previous developments of parking assistance systems. The new development will use JIRA as a tool to manage the product backlog. The new functional requirements are gathered there.

A transfer to JIRA of the DOORS requirements known from previous projects has proven impracticable, so that a backlog entry in JIRA with a brief description and a link to the corresponding DOORS entry is created for each business requirement in DOORS. In order to ensure the bidirectional traceability required in Automotive SPICE, the IDs of the backlog entries are also maintained in DOORS.





In agile development, requirements can be introduced into the development process even after implementation has begun. This also applies to quality requirements. It is crucial that quality management is informed about new planned functional requirements and that the resulting quality requirements can be added to the backlog by the Product Owner.



Example **Charging station search**

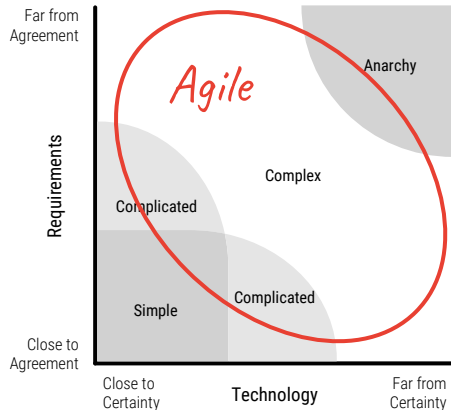
The app shall be expanded to include functions for customization (e.g., charging station search near the stored residential and workplace addresses). This results in legal boundary conditions regarding data privacy, such as compliance with the GDPR in Europe. These boundary conditions must also be taken into account when the development is carried out.



Decision for or against agility

Simple or complex project?

The crucial argument for agility is uncertainty regarding business requirements and technical implementation of complex products. Whether this complexity is present in the development of a system cannot be answered in general terms. If requirements can be described completely, correctly and with enough precision, it would be more appropriate to develop according to a classical approach.

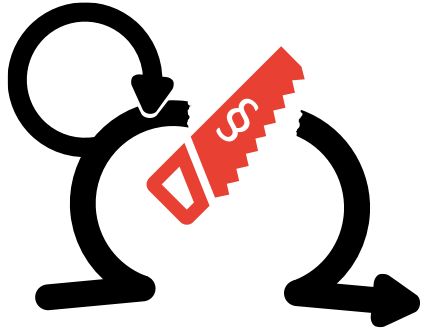


Low or high change costs?

Low costs for necessary revisions of the increments implemented to date are the prerequisite for an economical, agile approach. These costs seem to be very high for hardware: New features may require revision of the hardware design and thus creation of new production tools. The idea of continuously rolling out new versions of embedded systems, combined with recall actions after each development iteration, shows that agility that includes continuous delivery would be economically unreasonable.

If, however, one only considers pure development and excludes production and delivery, in many cases hardware can be developed economically in an agile way. To this end, only the term “deliverable” needs to be expanded: If “deliverable” in pure software development means delivery to the customer, “deliverable” here should be interpreted as “ready for volume production.” Production itself then takes place in a conventional phase-oriented manner and according to well-defined process steps.





Boundary conditions compatible with agility?

Legal regulations, industry-specific guidelines and in-house processes can apply to the development of products and must be respected. It is necessary to check whether the agile procedure is in principle compatible with these standards and to what extent the agile procedure may have to be modified.

To this end, the following process should be carried out once in a company when introducing agile methods and, if necessary, again when changes are made to relevant standards:

- 1. Identify conflicts between standard and agility:** For each standard, it must be checked whether agile development is to be brought into line with the standard. Process groups, process steps, process results, artefacts or other rules must be examined regarding compatibility with the agile approach.
- 2. Technical evaluation of rules:** If a rule contradicts agile principles, practices or the intended agile framework, it should be checked whether the rule makes sense from a business point of view. If a rule is judged to be so important that loss of agility should be accepted, this rule should remain unchanged.
- 3. Change rules in line with the standard if necessary:** If the retention of a rule does not outweigh the loss of agility, it should be checked whether the rule can be adapted in such a way that the rule conforms to agile development.
- 4. Negotiate rules if necessary:** If reinterpretation or tailoring is not possible within reason, the suspension of the rule or its amendment may be negotiated. In particular, the rules of standards that have been defined in-house come into question for this.





Example Parking assistance system

Automotive SPICE and ISO 26262 requirements for ASIL D must be considered in the development of the parking assistance system. The degree of independence I3 required by ISO 26262 for confirmation measures for ASIL D is ensured by the fact that the auditors work closely and continuously with the development team on behalf of the Product Owner but belong to different departments and thus report to different superiors than the members of the agile team.

In addition, the company generally demands that the degree of maturity of all new parts be tested in accordance with the VDA standard. Due to poor compatibility of this standard with agile development, several changes are agreed for the development of the parking assistance system. For instance:

- Instead of a complete specification, only an initial product backlog is required before assignment takes place.
- Maturity levels 3 to 6 are replaced by burn-down charts.

Organization



The organization of the development project must be clarified in advance. In an agile environment, the following questions, among others, must be answered:

- Which agile framework should be used for the development?
- If applicable, what does the scaling model look like?
- Who takes on the role of Product Owner and Agile Team Master (or Scrum Master)?
To which organization or organizational unit do these roles belong?
- How should the development team be composed? To which organizations/organizational units should their members belong?
- How is the transparency of the product backlog, development status and other artefacts ensured? Who should be involved in the flow of information?
- Who can/should participate in the agile rituals?



Example **Parking assistance system**

Overall responsibility for the development of the parking assistance system remains with the technical development of an OEM. The sales department provides the Product Owner. A supplier should develop the software for the system. In addition, individual suppliers of system components (e.g., for sensors) are integrated partly via the OEM and partly via the tier 1 supplier. The development at the supplier is carried out according to the Extreme Programming Framework.

The supplier has set up three agile teams that are loosely coupled and essentially coordinate their activities according to the SAFe approach. Each of these teams has its own agile team master and its own team Product Owner. On the OEM side, there is a Quality Management department which, in coordination with the Product Owner of the sales department, will incorporate requirements and formally approve the entire system developed before each delivery.



Example Charging station search

A single team is to develop the charging station search app on behalf of an OEM. The development is to take place first according to Scrum, later according to Kanban if necessary. The OEM provides the Product Owner; the development team and the Scrum Master are provided by the contractor. The OEM's quality management department communicates with the Product Owner throughout the entire development process, sets requirements and participates in backlog definitions and sprint reviews as required.





Evaluate suppliers




Supplier audit

28

Contract arrangement and trial period

30

Supplier audit



Many companies in the automotive industry require a supplier audit to be carried out before the order is placed. Depending on the type of product to be developed, different standards must be adhered to. For agile development, it must be checked whether and to what extent these standards are compatible with an agile approach.

In contrast to supplier audits in conventional development projects, the following criteria should be considered for agile development:

- The supplier should demonstrate expertise in development according to agile values and principles and, depending on the agile framework, appropriate qualifications or certificates for the agile roles assumed by them.
- The supplier should demonstrate expertise, tools and methods for quality-assured continuous integration and continuous delivery.



Contract arrangement and trial period

The formulation of contracts with suppliers is not the task of quality management. Nevertheless, the type of contract plays a major role in quality management, since the contract should include requirements for the evaluation of suppliers. In contrast to the classic waterfall model, agile development does not have a complete specification that could serve as the basis for a contract for work and services. Instead, alternatives to the conventional fixed-price agreement with binding requirement specification must be found.

For example, the principal could be given the option of terminating the contract and assigning another supplier. Evaluation of the supplier then takes place during the collaboration and less in advance through a supplier audit.





Example **Charging station search**

Two companies have been short-listed to implement the charging station search app. The principal agrees a service agreement with both companies in which the companies undertake to develop increments that can be delivered every two weeks. Four months after the start of development, the principal evaluates the performance of the two companies, chooses one of them for further cooperation and terminates the contract with the other.

In order to ensure conformity with the company's internal guidelines on temporary employment, the contracts stipulate that the official contact person of the contractor is always present during the sprint planning phase and authorized to accept orders.





Assure maturity level

Verification of software quality	34
Validation of the business requirements	36
Automation of quality assurance	39
Continuity and cooperation	41



Verification of software quality

In an agile development team, it is the task of the entire team to ensure quality. In many cases, quality assurance experts can and should be part of the development team, regardless of whether they belong to the same organizational unit as the programmers or not. These quality assurance experts carry out a large part of the quality assurance tasks within the team. They are not solely responsible for quality but only as part of the development team.



Cross-functional
development team



Example Parking assistance system

Due to the high quality requirements for the parking assistance system, especially with regard to functional safety, the agile teams each have several quality assurance specialists with a focus on test automation, performance and functional safety who are part of the development team. These quality assurance experts in particular identify safety-relevant test cases, support the developers in implementing the corresponding unit tests, verify the integrated software and

configure the DevOps tools so that all critical tests are automatically tested as regression tests before a new version is delivered. The quality assurance experts belong to a "Quality Assurance" organizational unit that is independent of the development department.



Validation of the business requirements

In essence, it is the client's task to validate the implementation of the functional requirements, boundary conditions and quality requirements in purely business terms, while at the same time leaving it to the development team to decide how to implement these requirements technically.

How such quality management can be organized in practice depends strongly on the type of product and the associated requirements and cannot be defined in general terms.



Product Owner & stakeholders



Example **Parking assistance system**

When it comes to assuring the degree of maturity of the parking assistance system, the focus is on the quality criteria of functional suitability, particularly functional safety, as well as time behavior and reliability. The integrated system, whether in a virtual test bench or the physical component or physical vehicle, is tested according to these criteria.

The testing itself does not differ fundamentally from the testing of a conventionally developed system, but these tests are carried out more frequently and not only at the end of product development. In addition, quality management on this level focuses on explorative tests; mass testing of various test cases is carried out within the team and, if possible, on the unit level.





Example **Charging station search**

The requirements for maintainability, usability and portability are high. Precisely because the app is to be brought onto the market quickly and then successively expanded to include further functions, a high degree of modularity, reusability, analyzability, modifiability and testability must be guaranteed. To this end, quality management regularly records the speed at which functional requirements are implemented. Slowing down of the speed is an indication of a lack of maintainability of the software.

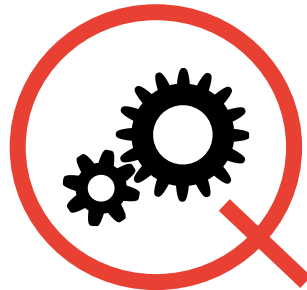
Quality management also focuses on the usability of the app and continuously coordinates this with sales and the Product Owner. Quality management conducts usability tests with potential users of the app and discusses the results with the agile team. The Product Owner records the resulting requirements in the backlog and prioritizes them.

In addition, quality management initiates performance and load tests by the supplier and evaluates the results. These tests also do not only take place at the end of the project but are carried out several times during the development period.

Automation of quality assurance

Due to the short iterations, test automation is of great importance in agile development. It is a prerequisite for enabling continuous integration and continuous delivery, if required, at a reasonable cost. Depending on the type of product, quality management should demand this from the supplier.

For quality management, this also means that bugs or other shortcomings discovered during or after development should be covered by an automated regression test in the future, if possible, and do not have to be repeated manually.



DevOps team





Example **Parking assistance system**

The various development teams of the suppliers operate their own continuous integration systems for their components to be developed. Using these systems, the tested software units are imported into the respective integration environments and automatically tested there on the component level.


In addition, the OEM as the client provides the development teams with a platform on which the developed hardware can be installed and to which the software can be uploaded online.

Changes are not applied to this acceptance instance until the components have successfully passed all tests at the respective supplier. The acceptance instance is used by quality assurance experts of the agile development teams as well as by the Product Owner, quality managers and quality experts of the customer. The OEM reports the defects there, which are mostly detected exploratively. If possible, test functions derived from this will be implemented on the supplier's unit level and automatically executed as regression tests in the supplier's system in the future.

Continuity and cooperation

The conventional waterfall model provides phases for quality assurance. There are no such phases in agile development. Quality assurance accompanies both the entire development process within the development team and the client's maturity test. For quality management on the client side, this means that it should be informed about implemented product backlog items (PBIs) during an iteration, be able to participate in reviews as required and in consultation with the Product Owner, and be familiar with the release plan of the Product Owner so that it can test the product more extensively before a planned release. The aim is to provide the agile team with feedback on the increments implemented as early as possible.





Such continuity in the cooperation between customer and supplier presupposes a different type of cooperation than with a classic approach. Quality management – whether represented alone by the Product Owner or by one or more stakeholders from a quality management department – participates as personally as possible in agile events such as backlog refinement or review. It knows the developers and is available to the development team as a partner.



Example Parking assistance system

The OEM's quality management checks the implementation of its own subject-specific quality requirements and functional safety requirements.

Requirements for eliminating bugs and deficits are included and prioritized in the product backlog in agreement with the Product Owner.

Quality managers, safety engineers and other quality experts of the OEM take part in discussions on requirements gathering and analysis as well as in the presentation of the increments and are thus in dialogue with the agile teams.



Release products



Release products

Depending on the type of product, quality management may have to formally release a delivery of the product to the customer or a transfer to the manufacturing process. This approval process is neither part of the agile manifest nor one of the common agile frameworks. Instead, the Product Owner, for example, can decide independently according to Scrum whether an increment should be delivered or not.

Nevertheless, a formal release process by quality management does not represent a fundamental contradiction to an agile approach if quality management has tested the product during the previous development period and reported defects to the team promptly, so that final acceptance should not result in extensive change requirements and release can take place without delay. It should be avoided that quality managers or other stakeholders report deficits that have been contained in the product for a long time and that could have been identified at an earlier point before the planned release or even impose new requirements and thus prevent a timely release.

This is avoided by all stakeholders working closely and continuously with the Product Owner and the agile team throughout the entire development period, focusing on what is necessary from a business and legal point of view.

Depending on the product class and context, release cycles in an agile environment can be much shorter than in a conventional development process. Particularly for short cycles, the release must be able to take place quickly. A high degree of test automation, which at least partially eliminates the need for time-consuming manual testing, makes this possible.





Example **Parking assistance system**

The development of the parking assistance system is embedded in the overall vehicle development with a specified date for Start-of-Production (SOP). Although the parking assistance system is developed incrementally in short iterations of a few weeks, there is only one release date.

The required formal acceptance and release by quality management takes place as usual, but due to the preceding cooperation between the agile teams and quality management and due to the high level of test automation that is now possible, it can take place faster than is usual with a conventional procedure. The effort for quality management is not concentrated shortly before SOP but is spread out over the entire development period.



Example Charging station search

The app for charging station search should be rolled out as quickly as possible in order to make it possible to evaluate usage behavior and acceptance in the market in a timely manner. The first release is only a Minimum Viable Product (MVP), which will be functionally extended successively. The Product Owner decides when the product is a MVP and should be rolled out as the first release. Updates are then performed approximately every two to three sprints; the Product Owner also decides on the rollout date in this case.

Quality management, which is in contact with the Product Owner throughout the entire development period and is informed about the development status, has made sure that there is a comprehensive suite of regression tests that can be viewed at any time by quality management and that the processes for continuous integration and continuous delivery are implemented. For this reason, quality management releases the respective increments promptly without extensive checks.



Observe the field and derive lessons learned



Observe the field and derive lessons learned

For products that have been developed in an agile manner, field observation is basically carried out in the same way as for products that have been conventionally developed. The failure correction processes established in the automotive industry can also be used for agilely developed products.

If a product is changed by updates, insights from the field can be used in an agile way in the current development. In particular, the agile team can check assumptions with the help of A/B tests. Frequent releases allow lessons learned to be derived promptly. These can be used by the Product Owner and stakeholders to identify and set new requirements, as well as by the agile team to optimize the product technically.



Example **Parking assistance system**

After the SOP, the new parking assistance system will be used in the field for the vehicle project for which the system was developed. Complaints about the parking assistance system are handled by the OEM as usual via the failure correction process and, if necessary, lessons learned are derived for the development of future comparable systems.



Example **Charging station search**


The Product Owner analyses the usage behavior by evaluating the databases and log files with the help of analysis tools and by analyzing the relevant KPIs. From this, they can conclude what value the individual PBIs implemented so far have and what value the PBIs not yet implemented will probably have. Accordingly, they prioritize the product backlog and add new entries to it.



Live agile principles



Individuals and interactions



The quality of the product depends decisively on the competence of all employees involved. This is especially true for agile development and applies to the quality assurance experts within the team and the quality managers, who introduce and validate quality requirements on the client side. In an agile environment, personal interaction of experts and joint finding of solutions often replaces comprehensive process guidelines. This demands a lot from the employees regarding their professional, communication and team competence.

It changes the role and nature of the work of quality management: it is less the quality police that verifies formal compliance with standards but rather the partner that supports development by taking the customer's point of view and contributing to and ensuring development. Quality managers should think and act in a result-oriented and less process-oriented way. These competencies should be nurtured among quality management employees and should be reflected in a corresponding talent management system.

Working software

For the customer it is crucial that the product works and has the required quality. Documentation of the product and product development is not an end in itself but should serve this purpose. Conventional quality management assesses the development status primarily based on documents and demands what is in them accordingly. However, quality management should understand and accept that an agile team may communicate and archive information with each other and with its environment in a different way than is common in the conventional context.

Quality management should focus on the implemented business requirements and examine the usefulness of documentation in this respect.

Due to the short iterations and frequent revisions of the product, testing is challenging in an agile environment. Test automation plays a key role here. The task of quality management is to set appropriate requirements regarding quality-assured continuous integration and continuous delivery and to support and ensure their implementation.



Customer collaboration

The Agile Manifesto emphasizes the importance of close cooperation between the customer and the supplier. Cooperation based on trust is more important than the contractual arrangement between the two parties. For the client's quality management, this means that it continuously accompanies the development process instead of verifying adherence to contractual commitments in a quality assurance phase on the basis of the documents alone.

Personal contact and interaction with the agile team – whether with the Product Owner for clarifying requirements, the Agility Master for procedural questions or the development team for questions during a development iteration – is an important prerequisite for an agilely developed, quality-assured and high-quality product. Quality management should focus on continuous monitoring and participation in agile events like backlog refinements and sprint reviews.

Responding to change

Agile development offers the customer the opportunity to formulate requirements even during product development and to incorporate them into the development process. This also applies to quality requirements. Quality deficits that are detected during development and perhaps only after delivery can be remedied by new quality requirements in ongoing development. Critical requirements, especially regarding functional safety or cybersecurity, should certainly be known and implemented before delivery.

Uncritical quality requirements, for example regarding usability, can be implemented by the agile team later, if installation of updates does not involve a great deal of effort. With early deliveries and the use of analysis tools, quality management can gain experience in the field and derive new requirements. Quality management thus assumes an important task in closing the agile loop.



Conclusion

Quality management in the automotive industry encompasses the same fields of activity in an agile environment as in a conventional environment. Even for agile development, quality requirements must be introduced, suppliers evaluated, maturity levels inspected, deliveries released, the field observed, and lessons learned considered for further development. It is also important to consider and adhere to binding quality standards.

However, the way these activities are organized and carried out may differ significantly from the quality management of waterfall-type development. Accordingly, agile teams and quality managers must interpret standards differently than usual and at the same time understand and achieve the goals associated with the standards. This requires both a deep understanding of the individual standards and their meaningful application to the respective development project and, if necessary, adaptation of agile practices.

If an agile approach to a development project makes sense and is possible, the quality should be managed and safeguarded according to the agile principles:

- Personal interaction of quality managers and quality assurance experts with the team and its members is more valuable than an assessment according to what is filed.
- Continuous guidance over the entire product lifecycle is more effective than isolated phases of quality assurance.
- Validation of the quality requirements and their adaptation and honing even during the development process is more effective than pre-specification of all potentially required quality characteristics in advance.
- Iterative adaptation of the methods and tools of quality management to the respective project is better than the use of a general canon of methods for all developments.



Author

Gunnar Harde

Contact

AQI Automotive Quality Institute GmbH

Franzoesische Strasse 13-14

10117 Berlin

Germany

kontakt@aqigmbh.de

www.aqigmbh.de

This publication is based on the expertise of the AQI and its scientific partners. It represents a consolidated position on the topic under examination.

This work including all its parts is protected by copyright. Any use not expressly authorized by copyright law requires prior permission.

© 2019 AQI