jama software®

LIVRE BLANC

Top 15 des Erreurs en Matière de Standard ISO 26262

Comment éviter et réduire les erreurs commises par les équipes -

Par Vance Hilderman

Quels sont les points communs entre le golf et le développement de logiciels automobiles ? Pour un débutant, il n'y en a pas, a priori. Cependant, les similitudes sont plus nombreuses qu'il n'y paraît.

Quelques points communs entre le golf et le standard ISO 26262 mis en lumière par l'auteur :

- Les deux semblent faciles à démarrer, mais les complexités embrouillent les débutants.
- Les deux semblent être bon marché au départ, mais deviennent plus coûteux dès lors qu'ils deviennent des activités sérieuses.
- Leurs règlements officiels contiennent des subtilités qui prennent des années à être maîtrisées.
- Les deux peuvent être pratiqués pour le plaisir et sans grand effort, mais le stress augmente énormément lorsque l'on s'y engage dans un but lucratif.
- Les règles sont suivies par les joueurs et les arbitres, et regorgent de subjectivité.
- Les compétences viennent avec la formation, puis l'entraînement.
- Votre deuxième tentative sera plus réussie que la première.
- Parmi tous les participants, seuls quelques-uns sont des experts.

Tout comme le golf professionnel, le standard ISO 26262 nécessite d'améliorer constamment vos compétences tout en cherchant à minimiser les erreurs. Mais il est utile de savoir quelles sont les erreurs communes et peu communes rencontrées par les équipes qui appliquent le standard ISO 26262.

Continuez à lire pour connaître le top 15 des erreurs relatives au standard ISO 26262, et apprendre à les éviter ou les réduire.

Tout d'abord, il est important de comprendre que même les pratiquants chevronnés du standard ISO 26262 le considèrent comme étant subjectif, vague et générique. Malgré cela, la certification automobile nécessite une conformité rigoureuse et une qualité particulièrement élevée.

Dans un scénario avec un temps et des ressources infinis, il est possible de se conformer au standard ISO 26262 avec succès. Cependant, l'environnement concurrentiel du développement automobile change rapidement. Ainsi, l'objectif est d'avoir une mise en conformité qui atteint ou dépasse les standards minimums tout en égalant ou en dépassant la concurrence.

Pour briller dans le monde du standard ISO 26262, il faut développer l'électronique automobile et garantir la conformité de la manière la plus rapide et productive possible, en évitant les erreurs coûteuses. Rappelons aussi que les erreurs ne sont pas causées par la malchance, de la même façon qu'elles ne sont pas évitées par la chance.

Les erreurs en électronique automobile sont le résultat d'un manque de compréhension, de planification et d'application des véritables intentions et règles cachées du standard ISO 26262.

Comme l'a déclaré le golfeur professionnel Arnold Palmer à l'issue d'un tournoi très spectaculaire : « Moi, chanceux ? Je me suis aperçu que plus je m'entraînais, plus j'étais chanceux ! » Riche de cette information, j'espère que vous deviendrez plus chanceux à mesure que vous vous entraînerez pour le standard ISO 26262 et que vous affinerez vos processus pour éviter les erreurs courantes.

Mais quelles sont les erreurs courantes en développement automobile ? Commençons par l'erreur n° 15.

Ci-dessus : Quelques-uns des 35 pays dans lesquels les ingénieurs AFuzion ont travaillé sur place avec les clients.

Erreur n° 15 : Mauvaise Qualification des Outils et Plateformes

Que vous utilisiez un outil spécifique à une tâche ou une plateforme de développement de produits robuste comme Jama, vos options seront généralement logicielles, puisque ce sont des instructions numériques qui sont utilisées dans le développement ou la vérification de logiciel et/ou de matériel automobile(s).

Il est également bon de noter que TÜV SÜD (autorité de certification mondiale) a annoncé en 2016 que Jama était certifié « adapté » à la mise en conformité au standard ISO 26262 pour la sécurité fonctionnelle.

Les activités de qualification des outils répondant au standard ISO 26262 peuvent être classées en deux catégories : classification (analyse du logiciel) et qualification.

Certaines organisations omettent l'activité de classification, qui consiste à analyser l'utilisation attendue de l'outil (cas d'usage). Cette analyse doit être documentée, analysée et évaluée afin de déterminer si une défaillance dans l'outil peut engendrer des erreurs dans le logiciel/matériel en développement ou une incapacité à les détecter. La classification doit permettre de déterminer la classe d'impact de l'outil (TI1 ou TI2). Il faut ensuite déterminer la classe de détection d'erreur de l'outil (TD1, TD2 ou TD3).

Il est obligatoire que cette analyse définisse le niveau de confiance de l'outil, par exemple TCL1, TCL2 ou TCL3. Les outils avec le TCL le plus bas (c.-à-d. TCL 1) ne nécessitent pas davantage de qualification. Pour TCL2 et TCL3, il faut poursuivre la qualification de l'outil.

Le choix de la méthode de qualification d'outil doit être basé sur le TCL requis, ainsi que sur le niveau d'intégrité de sécurité automobile

(ASIL) du matériel ou logiciel lié à la sécurité développé ou vérifié avec l'outil en question.

De nombreuses entreprises ne parviennent pas à exécuter pleinement les activités de classification et de qualification des outils. Elles travaillent souvent trop peu sur la qualification des outils essentiels, mais qualifient parfois des outils plus simples pour lesquels cela n'est pas nécessaire.

Erreur n° 14 : Croire que les Tests Améliorent Directement la Qualité

On croit souvent à tort que dans le cas du standard ISO 26262, les tests servent à améliorer directement la qualité et la sécurité automobiles. **En vérité, les tests servent à évaluer la qualité. Ils n'améliorent pas intrinsèquement la qualité ou la sécurité automobile. Si c'était le cas, l'électronique automobile bas de gamme pourrait atteindre la perfection grâce à des tests répétés.** Les tests doivent plutôt servir à évaluer la qualité grâce à des boucles de feedback dans le processus de développement. Cela permet d'améliorer les exigences, la traçabilité, la couverture et la robustesse par le biais de processus de sécurité rigoureux axés sur l'ASIL. Tout ceci peut être facilement accompli avec une plateforme de développement de produits robuste.

Erreur n° 13 : Faible Visibilité de Gestion et Revues Manuelles

Le standard ISO 26262 nécessite d'adhérer à de nombreux objectifs liés à la sécurité et de passer en revue des dizaines d'étapes de processus, de documents et d'artefacts. Cependant, les managers peuvent rarement voir le véritable statut des projets et revues. **Une réponse : automatisez le processus de revue grâce à des checklists conformes au standard ISO 26262 et automatisez le processus de gestion de projets avec un outil de suivi de projet propre au standard ISO 26262.** Dites à la direction d'exiger que les managers les tiennent informés régulièrement de l'atteinte d'objectifs et de revues majeurs tout au long du processus d'ingénierie. Encore une fois, une plateforme de développement produit robuste peut faciliter considérablement ces processus et éviter les erreurs soulignées dans cet article.

Erreur n° 12 : Ne Pas Détecter les Erreurs de Conception le Plus tôt Possible

Le standard ISO 26262 est conçu pour minimiser, réduire et détecter les erreurs de conception. Ceci étant, le standard ISO 26262 ne tient pas compte du budget pour un système ASIL donné : les constructeurs automobiles sont libres de dépenser autant qu'ils le veulent pour leur mise en conformité.

Le marché, en revanche, est différent : les études montrent que le coût pour corriger une anomalie dans la phase de tests formels est 5 à 10 fois plus élevé que lorsque l'anomalie est détectée et corrigée pendant la phase de développement.

Les standards de développement de logiciel et de matériel, les revues par des pairs avec checklists détaillées et les tests basés sur les composants sont autant d'étapes

importantes. Mais l'analyse statique de logique à l'aide d'un outil d'analyse statique payant est une étape souvent négligée. Astuce : faites une recherche « analyse statique de logique du standard ISO 26262 ». L'analyse statique (analyser la logique avant les tests) est différente de l'analyse dynamique, qui exécute la logique avec des tests en temps réel.

Les deux sont importantes, mais l'analyse statique est souvent négligée, ce qui engendre plus de coûts que nécessaire. Dans la vaste majorité des cas, une analyse de code coûte moins cher que les conséquences de son omission.

Les analyses statiques de code avancées peuvent fournir des revues de code automatisées et continues (C, C++, VHDL, etc.) avec chaque archivage afin de vérifier que les modifications n'ont pas causé de nouveau problème (récursivité ou violation de limites ou standards de complexité cyclomatique).

Erreur n° 11 : Ne Pas Utiliser un RTOS Prêt à la Certification Payant

Ces dix dernières années, un nombre croissant de systèmes pour des applications en aéronautique, en contrôle industriel, en appareillage médical, et maintenant en logiciels automobiles utilisent un système d'exploitation temps réel (RTOS). Les projets plus anciens ou plus simples se contentent d'un simple noyau ou d'une boucle d'exécution/boucle de poll pour contrôler l'exécution des programmes.

Cependant, la tendance au développement rapide, les conceptions évolutives et réutilisables, les catalogues & drivers tiers, les protocoles de communication, le partitionnement, la sécurité et le standard ISO 26262 justifient tous le besoin d'un RTOS payant.

Pensez à effectuer des analyses à taux monotones (RMA) afin de diminuer davantage les risques d'erreur de votre RTOS lors de la phase de qualité des tests. Pour le standard ISO 26262, il existe quinze à vingt critères principaux applicables au choix de RTOS.

La conformité au standard ISO 26262 d'un RTOS peut prendre des années et coûter des centaines de milliers d'euros, d'où la nécessité d'un RTOS dont la conformité est prouvée.

Erreur n° 10 : Manque de Tests Automatisés

Les tests logiciel sont un aspect essentiel du standard ISO 26262, et ils prennent souvent plus de temps que l'écriture du code. Si la programmation bénéficie (ou devrait bénéficier) pleinement des outils logiciels modernes, les tests sont souvent négligés et peu pris en compte dans les outils et la productivité.

Les tests n'améliorent pas intrinsèquement la qualité, ils servent plutôt à la mesurer, améliorant ainsi les processus, qui améliorent à leur tour la qualité.

Les tests seront effectués des dizaines voire des centaines de fois pour les mêmes exigences tout au long de la vie du produit et couvriront de nombreuses évolutions.

La meilleure stratégie pour les tests de régression consiste à tester à nouveau automatiquement autant de composants logiciels que possible. Ainsi, les tests doivent toujours être automatisés, sauf pour les produits les plus simples.

L'équipe de test doit participer à toutes les revues des exigences, de conception et de code afin de garantir que ces artefacts sont testables et de préparer des cas de tests automatisés aussi tôt que possible. Tous ces processus complexes peuvent être simplifiés avec une bonne plateforme de développement de produits.

Erreur n° 9 : Ne Pas Enregistrer le Path Coverage Lors des Tests Fonctionnels

La couverture structurelle (couramment appelée « Path Coverage ») est de plus en plus requise pour les logiciels de niveau A, B et C. (Le Path Coverage n'est pas requis pour le niveau D, et aucune étape du processus du standard ISO 26262 n'est requise pour le niveau E). Cependant, le Path Coverage est souvent demandé et effectué après les autres formes de tests exigés par le standard ISO 26262. C'est à la fois inefficace et inutile.

Il faut réfléchir davantage en avance à l'environnement logiciel et à la suite de tests, et effectuer le Path Coverage lors des tests boîte noire des exigences (c.-à-d. les tests en fonctionnement réel).

N'oubliez pas : quand vous faites l'instrumentation du logiciel pour obtenir les données de Path Coverage, vous devez effectuer les tests deux fois. Une fois avec l'instrumentation, puis une autre fois sans, afin de pouvoir corréler les résultats et confirmer que l'instrumentation n'a pas masqué de défaillance lors du test. Une erreur étroitement liée consiste à n'effectuer la couverture structurelle que dans le but de répondre aux exigences du standard ISO 26262 pour les ASIL élevés.

La couverture structurelle sert à 1) vérifier si le logiciel répond à ses exigences, 2) déterminer si les performances du logiciel sont bonnes et 3) montrer que tous les chemins sont couverts (selon l'ASIL), ou qu'il n'y a pas de code mort pour les ASIL plus élevés.

Ne vous contentez pas d'activer le compteur d'instructions d'un débuggeur, de paramétrer les registres et de faire une exécution pas à pas afin « d'obtenir la couverture structurelle ». Malgré les apparences, cela ne répond pas aux exigences de couverture structurelle du standard ISO 26262.

Erreur n°8 : Itérations de Logique Trop Nombreuses

Il est normal qu'un nouveau projet contienne une logique de référence avec de multiples itérations. Cependant, la plupart des projets dépassent largement le nombre raisonnable d'itérations car le développement logiciel est perçu comme un processus itératif et non d'ingénierie. C'est d'autant plus vrai pour les organisations faibles, qui pratiquent des méthodes Agile ou Lean faibles. Agile et Lean sont de bonnes méthodes, dont on peut tirer de nombreux avantages, à condition d'y mettre de la vraie logique. Oui, il est nécessaire d'inclure de l'ingénierie logicielle dans la création de logiciel. Les développements liés à la sécurité de l'électronique automobile ne peuvent pas simplement se baser sur des process d'itération « informelle ».

Les itérations de code trop nombreuses sont causées par l'une des défaillances suivantes:

1. Exigences trop faibles
2. Standards de code trop faibles
3. Checklists trop faibles

Il existe diverses sources où trouver des standards de code et des checklists du standard ISO 26262. L'AQ doit surveiller et rapporter les versions de code trop nombreuses, car elles peuvent indiquer une définition de conception faible, des exigences faibles ou inutilement complexes.

Erreur n° 7 : Traçabilité Inadéquate et Non Automatisée

La certification du standard ISO 26262 exige la traçabilité descendante et ascendante, cette dernière étant nécessaire dans la couverture structurelle pour les ASIL élevés. La traçabilité descendante garantit l'intégrité des exigences au niveau du système, des exigences de logiciel, du code de logiciel et des tests du logiciel. La traçabilité ascendante garantit que seules les fonctionnalités spécifiées dans les exigences sont présentes.

La traçabilité doit faire l'objet de vérifications régulières afin de confirmer que les revues appropriées ont été effectuées. La traçabilité doit être complète et vérifiée en tant que telle afin de prouver la conformité au standard ISO 26262.

Ceci étant, il est possible d'améliorer l'efficacité de la gestion de projet et de la productivité en assurant une traçabilité précise dès le début du projet et en continu par la suite. De plus, le développement automobile fait de plus en plus usage de modèles avec le développement basé sur les modèles (MBD).

Lorsqu'on utilise des modèles, la traçabilité doit être fournie dans le modèle. Cela signifie qu'elle doit montrer les exigences sur lesquelles le modèle est basé, les éléments du modèle qui les représentent et le code ou les tests qui suivent lesdits éléments.

La traçabilité sera nécessaire tout au long de la vie du produit, qui atteint souvent plusieurs dizaines d'années pour les modules de logique communs. À l'exception des projets particulièrement petits (moins de 2000 à 3000 lignes de code), la traçabilité doit être semi-automatisée par le biais d'un outil interne ou payant tel que Jama. N'oubliez pas que la traçabilité ne doit pas être seulement gérée par l'automatisation : elle doit toujours faire l'objet de revues.

Erreur n° 6 : Exigences Trop Faibles

Il est impossible de quantifier précisément et objectivement le niveau de détail nécessaire relatif aux exigences de matériel et de logiciel automobiles mentionnées dans le standard ISO 26262. Cependant, le niveau de détail doit être assez élevé pour que des concepteurs indépendants puissent parvenir à des conceptions équivalentes sur le plan fonctionnel sans avoir recours à des clarifications majeures ou à des suppositions.

Pendant des dizaines d'années, les chercheurs (Dr. Barry Boehm, Watts Humphrey, et leurs collaborateurs) ont prouvé que les suppositions étaient la cause première d'anomalies de logique. Ces suppositions sont causées par des exigences faibles.

En effet, le standard ISO 26262 conseille de baser les tests sur les exigences, et non sur la logique elle-même, ce qui veut dire que les exigences sont suffisamment précises pour couvrir la plupart des décisions de logique.

Les suppositions doivent être documentées en vue de faciliter la maintenance de la logique. La meilleure documentation prend la forme d'exigences indépendantes, elle ne se limite pas à des commentaires intégrés au code. Si les suppositions ne sont pas documentées, la personne qui modifiera les exigences dans trois ans (probablement quelqu'un d'autre) risque de mal comprendre pourquoi les exigences sont rédigées ainsi.

De manière générale, il faut essayer d'appliquer la méthode 1/25, c'est-à-dire un minimum d'1 exigence pour 25 lignes de code dans un langage de haut niveau.

Notez que la plupart des projets automobiles souffrent d'exigences trop faibles. Cela veut dire que des suppositions dangereuses, ou tout du moins inexactes, sont faites par des développeurs qui s'éloignent des fonctionnalités prévues.

Afin de garantir des exigences suffisamment détaillées, il est conseillé de développer des cas de tests fonctionnels (basés sur les exigences) en même temps que la spécification de ces exigences, par un ingénieur indépendant de l'ingénierie système en charge de la spécification des exigences. L'ingénieur de tests validera pour ainsi dire les exigences en rédigeant les cas de test, et les tests ainsi produits seront revus indépendamment par les ingénieurs système associés afin de confirmer que l'objectif des exigences est correct.

Réfléchissez aussi aux moyens de repérer et corriger ce genre de problèmes pendant les premières phases de revues des exigences. L'AQ peut appliquer des contrôles statistiques de processus de revue par des pairs afin de suivre la variation des caractéristiques essentielles (les types d'anomalies), le nombre de revues et la durée des revues que l'équipe compte atteindre, et les mettre en évidence. C'est une excellente opportunité pour améliorer les processus.

Erreur n° 5 : Planification Formelle et Analyse des Lacunes Inadaptées

Le standard ISO 26262 exige des documents de planification formelle. La plupart des entreprises ont déjà des processus conformes sur certains aspects, mais ne savent pas comment en tirer parti.

Au lieu de repartir de zéro et d'adopter de nouveaux processus, les entreprises doivent effectuer une analyse des lacunes pour le standard ISO 26262 afin de déterminer ce qui manque à leurs processus existants et ce qu'il faut faire pour combler ces manques.

Elles doivent ensuite se procurer ou élaborer des modèles et checklists relatives au standard ISO 26262, ainsi qu'une analyse des lacunes visant à adopter les bonnes pratiques propres à leur secteur.

Erreur n° 4 : Assurance Qualité Inadaptée

L'assurance qualité joue un rôle précis dans la certification automobile : l'apport de preuves que les directives du standard ISO 26262 ont été suivies.

AL'AQ sert à deux choses :

1. Garantir l'existence de plans, standards et checklists d'ingénierie conformes à toutes les directives de certification applicables
2. Effectuer des vérifications de l'ingénierie afin de déterminer si ces plans et standards sont appliqués.

Le personnel d'assurance qualité doit être indépendant et, idéalement, ne pas pratiquer d'activité d'ingénierie, tests compris. Cela peut s'avérer difficile dans les petites entreprises, où le candidat doit préciser la façon dont l'aspect d'indépendance est solutionné. L'assurance qualité est également une aide à la conformité des processus dans la mesure où l'équipe collecte les preuves de son passage réussi à la phase d'ingénierie suivante.

Erreur n° 3 : Repousser le Choix des Outils

Il n'est pas rare que les projets d'ingénierie choisissent le processeur et le langage de programmation avant de choisir les outils relatifs au standard ISO 26262. Quels outils ? L'environnement de développement, le RTOS, les drivers, les outils de tests, de modèles, de couverture structurelle, de gestion des exigences et de traçabilité comme ceux intégrés à la plateforme de développement de produit et aux outils de gestion de configuration de Jama Software.

Comme ces outils impactent directement la qualité, la productivité, les délais, le budget et la certification, le choix de l'outil doit être fait de manière globale et basé sur l'environnement de développement, les compétences et cultures des utilisateurs, le niveau de criticité, la réutilisabilité en aval tout en restant dans un budget réaliste.

Avant de choisir le processeur, le RTOS et le langage de programmation, il est préférable de consulter les experts du secteur sur les meilleurs outils actuels afin d'éviter que des incompatibilités n'impactent négativement le budget et les délais.

Erreur n° 2 : Négliger l'Indépendance

Plus l'ASIL augmente, plus le standard ISO 26262 exige de « l'indépendance ». On parle d'indépendance lorsque la personne à l'origine d'une étape ou d'un artefact de cycle de vie lié au standard ISO 26262 n'est pas celle qui vérifie cette étape ou artefact. La personne procédant aux vérifications peut être celle chargée de la revue pour les exigences nécessitant une revue.

Si le degré d'indépendance requis pour le niveau de criticité associé à l'étape ou artefact de cycle de vie n'est pas atteint, toute conformité au standard ISO 26262 est mise en échec. Il faut donc refaire ou reconcevoir l'étape, voire le produit dans sa totalité.

Si une itération ultérieure du produit a un ASIL plus élevé (ce qui est fréquent) et si le degré d'indépendance pour cet ASIL n'est pas atteint (ce qui est tout aussi fréquent), des modifications majeures sont nécessaires.

Ainsi, lorsque vous créez cette indépendance, prenez en compte l'ASIL le plus élevé que le produit « pourrait » un jour atteindre, et créez toujours plus d'indépendance que nécessaire. Le niveau de qualité de revue obtenu grâce à cette plus grande indépendance vaut la légère perte d'efficacité qu'elle engendre.

Erreur n° 1 : Ne Pas Considérer l'Ensemble du Standard ISO 26262 Comme un Écosystème Intégré

Le standard ISO 26262 compte 10 parties, ou « chapitres » :

1. Glossaire
2. Gestion de la sécurité fonctionnelle
3. Phase de concept
4. Développement produit au niveau du système
5. Développement produit au niveau du matériel
6. Développement produit au niveau du logiciel
7. Développement et fonctionnement
8. Processus connexes
9. Analyses orientées niveau d'intégrité de sécurité automobile (ASIL), et orientées sécurité
10. Directives pour le standard ISO 26262

Certains utilisateurs choisissent de suivre quelques parties du standard ISO 26262 comme s'il s'agissait d'une carte de restaurant. Or, ce n'en est pas une ! Plus précisément, le standard ISO 26262 exige que les utilisateurs adhèrent à tous ses objectifs applicables, au sens « déterminés par l'ASIL ».

Bien que nous aimerions pouvoir dire que les erreurs mentionnées plus haut sont les seules que nous ayons rencontrées, ou faites, avec le standard ISO 26262, nous ne le pouvons pas ! En effet, nous en avons faites ou observées des dizaines d'autres.

Nous espérons que les informations présentées ici permettront au moins au lecteur d'éviter ces erreurs majeures et de commencer à améliorer ses processus et outils, grâce à des choix judicieux, des procédures documentées, des formations mais aussi via ces enseignements.

À PROPOS DE VANCE HILDERMAN

Vance Hilderman est le fondateur de deux des plus grandes sociétés au monde de services de développement en avionique. Il a également créé la première formation DO-178 au monde et formé plus de 8 000 ingénieurs dans 45 pays à DO-178, DO-254, DO-278 et DO-200A. Enfin, c'est l'auteur principal du premier livre au monde sur DO-178 et DO-254.

À PROPOS DE JAMA SOFTWARE

Jama Software fournit la plateforme leader pour la gestion des exigences, des risques et des tests. Les équipes en charge de la production de produits, systèmes et logiciels complexes peuvent s'appuyer sur Jama Connect et sur des services pour raccourcir les temps de cycle (par secteur industriel), améliorer la qualité, réduire les modifications et minimiser les efforts tout en garantissant la conformité du produit. À la pointe des évolutions en matière de développement, Jama compte une clientèle toujours plus nombreuse, avec aujourd'hui plus de 600 organisations, dont SpaceX, Boston Scientific, Lyft, Deloitte, Alight, Samsung et Caterpillar.