Mesurer la couverture de code :

10 critères pour choisir un outil de couverture de code

Pour développer un logiciel sûr et fiable, les tests sont un élément indispensable de l'assurance qualité. Sans tests suffisants et documentés, il est impossible de déterminer si un logiciel est sûr et fonctionnellement correct. La mesure de la couverture de code (couverture de test) joue ici un rôle particulièrement important, car elle permet de déterminer dans quelle mesure un logiciel a déjà été testé. La couverture de code indique le rapport entre le code testé et l'ensemble du code. Pour simplifier, la couverture de code est par exemple de 75 % si trois des quatre possibilités existantes sont exécutées pendant le test.

Dans le développement de logiciels critiques pour la sécurité en particulier, les normes du secteur fixent des exigences précises de couverture de code : les produits ne peuvent pas y être certifiés sans la preuve d'une couverture de test suffisante. Mais dans d'autres projets de développement aussi, les entreprises accordent de plus en plus d'importance à la qualité logicielle et mesurent la couverture de code.

Divers analyseurs de couverture de code sont disponibles sur le marché. Ils diffèrent sensiblement par leur maniement et leur qualité. C'est pourquoi nous présentons dix critères essentiels pour choisir un outil de couverture de code :

1 Indépendance vis-à-vis du compilateur

Un outil de couverture de code doit bien sûr fonctionner avec le compilateur utilisé dans le projet. Il est toutefois très judicieux de miser dès le départ sur un outil utilisable indépendamment du compilateur. Un tel outil peut alors servir dans tous les projets, et aussi dans le projet en cours si le compilateur change. Un outil de couverture indépendant du compilateur se prête à des usages bien plus variés : c'est donc un investissement rentable.

2 Facilité d'utilisation

Le meilleur logiciel est utilisé à contrecœur, et donc rarement, s'il est inutilement compliqué ou mal pensé. À l'inverse, un maniement simple peut nettement améliorer l'acceptation d'un outil de couverture de test par ses utilisateurs. Idéalement, l'outil fonctionne en arrière-plan et ne crée aucun travail supplémentaire pour l'utilisateur pendant les tests.

3 Lisibilité des rapports de couverture

À la lecture des rapports de couverture, on doit voir d'un coup d'œil quelles parties du code ont déjà été testées et où la couverture fait encore défaut. Avec un bon outil de couverture, le testeur identifie facilement, au niveau du code source, les cas de test qui restent à réaliser. En exécutant ces tests manquants, il augmente la couverture de code de façon ciblée. Il évite en même temps le travail inutile qu'entraîneraient des tests redondants.

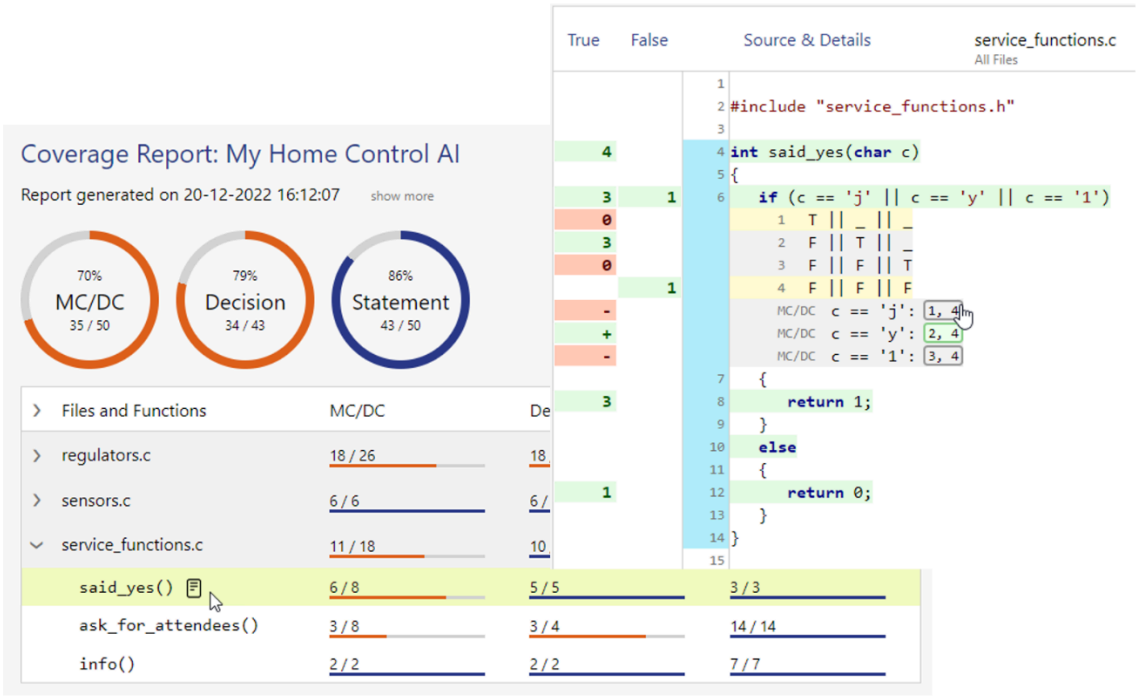


Fig. 1 : Outre une vue d'ensemble de la couverture des différentes parties du code, Testwell CTC++ affiche des informations détaillées qui montrent précisément dans quelle mesure le code source est couvert par les tests, y compris pour les niveaux de couverture les plus élevés.

4 Prise en charge des niveaux de couverture élevés pour le développement critique

Pour tester les logiciels critiques pour la sécurité, les normes (par exemple l'ISO 26262 dans l'automobile, la DO-178C dans l'aéronautique et l'EN 50128 dans le ferroviaire) imposent des niveaux de couverture élevés, jusqu'à la couverture MC/DC. Il est donc impératif de s'assurer que l'outil de couverture prend en charge tous les niveaux de couverture exigés. Pour pouvoir utiliser une solution dans la durée, il faut tenir compte non seulement des exigences actuelles, mais aussi des exigences futures. Bon à savoir : de nombreux outils de couverture ne proposent que la couverture des décisions ou des branches, et sont donc insuffisants pour le développement de logiciels critiques pour la sécurité.

5 Intégration souple

Même au sein d'une seule entreprise, les environnements de développement et les chaînes d'outils sont souvent très hétérogènes. Un outil de couverture doit s'adapter facilement à tous ces environnements. Son intégration dans le processus de build et dans l'exécution des tests doit se faire sans rupture et sans gros effort. Un outil également utilisable en ligne de commande offre des avantages pour la mise en place de builds automatisés.

6 Faible surcoût d'instrumentation

La plupart des outils de couverture mesurent la couverture de code en instrumentant le code source. L'outil enrichit le code source de « compteurs », qui comptent où et combien de fois les parties de code concernées ont été exécutées pendant les tests. Cela augmente toutefois

la taille du code d'origine. Pour les tests sur des cibles embarquées à la mémoire limitée, il faut donc veiller à réduire autant que possible ce que l'on appelle le surcoût d'instrumentation. Les besoins en mémoire varient parfois considérablement d'un outil de couverture de code à l'autre. L'analyseur de couverture de code Testwell CTC++ de Verifysoft Technology, par exemple, est très économe en ressources sur ce point. Avec Testwell CTC++, il est même possible de réduire encore l'espace mémoire nécessaire grâce à l'option de couverture par bits (bit coverage) : elle mesure alors seulement si une partie du code a été testée, et non combien de fois elle l'a été.

7 Prise en charge de plusieurs langages de programmation

Les entreprises travaillent souvent avec plusieurs langages de programmation, ou prévoient d'en introduire de nouveaux. Il est donc judicieux de choisir dès le départ un outil qui prend en charge tous ces langages, ou le plus grand nombre possible d'entre eux.

8 Prise en charge d'une programmation « créative »

Certains outils de couverture rencontrent des difficultés pour analyser des constructions de langage qui s'écartent des standards courants ou présentent une forte profondeur d'imbrication. Un bon outil de mesure de la couverture de test doit pourtant aussi venir à bout d'un style de programmation « créatif ».



Figure 2 : L'analyseur de couverture de test Testwell CTC++ est souvent comparé à un véhicule tout-terrain : l'outil fonctionne toujours de façon fiable, même sur un « terrain » difficile…

9 Adéquation au développement de logiciels critiques pour la sécurité

Pour le développement de logiciels critiques pour la sécurité, les normes concernées exigent que toute la chaîne d'outils soit qualifiée. Il s'agit de prouver que l'analyseur de couverture, comme les autres outils utilisés dans l'ensemble de la chaîne d'outils, fonctionne de façon fiable. Les éditeurs d'outils de couverture de code professionnels accompagnent les projets logiciels avec des kits de qualification et des conseils sur la qualification d'outils.

Les outils dotés d'un certificat (par exemple du TÜV) peuvent être utilisés selon de nombreuses normes de sécurité sans activité de qualification supplémentaire. La certification d'un outil est donc un critère important pour le choisir.

Dans ce contexte, il faut aussi vérifier si l'outil de couverture choisi est déjà utilisé avec succès dans des projets critiques pour la sécurité.

10 Licences d'évaluation, support technique et références clients

L'adéquation d'un outil de couverture à ses propres projets doit être vérifiée lors d'une évaluation de l'outil. Pendant cette période, vous vous faites déjà une idée de la qualité du support technique. Le support est-il joignable par téléphone, ou seulement par e-mail ? Quelle est la compétence de ses équipes ? Quels sont les délais de réponse ? Le manuel utilisateur est-il bien fait et pratique ? L'éditeur propose-t-il aussi des formations ? Enfin, il est conseillé de consulter les références clients de l'éditeur. Elles renseignent davantage sur la qualité de l'analyseur de couverture et sur le sérieux de son éditeur.

Synthèse

« Ce n'est pas sans raison que la couverture de code est obligatoire pour le développement de logiciels critiques pour la sécurité. Mais pour quiconque veut améliorer la qualité de son logiciel en général, mesurer et augmenter la couverture, et donc la valeur des tests, est aussi une bonne méthode », explique Klaus Lambertz, directeur général de Verifysoft Technology GmbH. « Lors du choix d'un analyseur de couverture de code, il faut veiller à ce que l'outil réponde aux exigences fixées. Des facteurs comme la facilité d'utilisation et un support professionnel jouent aussi un rôle important. Bien utilisé, un bon outil de couverture de test aide à améliorer nettement la qualité, à motiver davantage les développeurs et les testeurs, et à réaliser les tests de façon économique. »

La couverture de code en bref : les différents niveaux de couverture de test

Couverture des fonctions (Function Coverage)
La couverture des fonctions mesure si toutes les fonctions du programme ont été appelées. C'est le plus « faible » des niveaux de couverture de test usuels.

Couverture des instructions (Statement Coverage)
La couverture des instructions mesure le pourcentage d'instructions testées par rapport à l'ensemble des instructions.

Couverture des décisions / des branches (Decision / Branch Coverage)
À ce niveau de couverture, chaque décision doit être testée au moins une fois à vrai et une fois à faux. Pour les instructions *if* ordinaires, cela correspond à la couverture des branches, où chaque branche doit avoir été exécutée.

Couverture des conditions (Condition Coverage)
La couverture des conditions examine en détail les décisions composées. Pour les décisions formées de plusieurs conditions atomiques reliées par des opérateurs booléens, chacune de ces conditions doit être testée individuellement à « vrai » et à « faux ».

Couverture multicondition et couverture des conditions/décisions modifiée (MC/DC)
Pour la couverture multicondition, toutes les combinaisons vrai-faux possibles doivent être vérifiées pour les décisions composées. Lorsqu'une décision contient plusieurs conditions, cela demande un nombre de cas de test le plus souvent irréaliste. En pratique et dans les normes, c'est donc la couverture MC/DC (Modified Condition/Decision Coverage) qui est pertinente : elle réduit le nombre de cas de test tout en gardant une valeur informative suffisante à la couverture de test.

Auteur : Klaus Lambertz, Verifysoft Technology GmbH
© 2023 Verifysoft Technology GmbH

Traduction française : Precilog, distributeur de Verifysoft en France (www.precilog.com), 2026.
Document original en anglais : « Measuring Code Coverage: 10 Criteria for selecting a Code Coverage Tool », Verifysoft Technology GmbH.