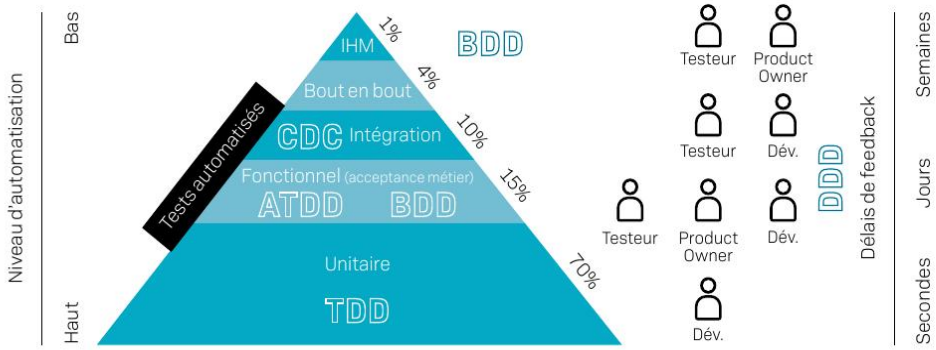
Le testing veut prendre sa part dans la révolution des développements

Le succès des Journées françaises des tests logiciels (JFTL) illustre la place grandissante de cette activité dans les nouvelles organisations de production rapide et largement automatisée de logiciels.



BDD : Behavior Driven Development / CDC : Consumer Driven Contract, dans le cadre d'un CDD, Contract Driven Development / ATDD : Acceptance Test Driven Development / BDD : Behavior Driven Development / TDD : Test Driven Development) / DDD : Domain Driven Design

SOURCE : ACPQUALIFE

Affluence impressionnante aux 10èmes JFTL, organisées par le Comité français des tests logiciels (CFTL) début avril à Montrouge. Plus de 1 100 participants revendiqués par les organisateurs et, à en juger par le taux de remplissage du grand auditorium et des différentes salles de conférences, le compte est bon ! Même impression d'ébullition sur les stands pendant les pauses : la grosse quarantaine d'exposants semblaient satisfaits de la fréquentation, puisque certains d'entre eux n'ont même pas attendu la fin de la journée pour réserver leur emplacement pour les JFTL 2019.

Pourquoi un tel succès, qui s'amplifie année après année ? C'est que, après des premières éditions marquées par un souci évident de pédagogie, voire de travail sur l'estime de soi d'un testeur pas vraiment valorisé dans l'organigramme de la DSI, le temps semble venu où ce dernier devient peu à peu incontournable. La faute, ou plutôt grâce, à l'avènement de DevOps et du Continuous Delivery. En effet, la synergie entre les équipes de développement et celles de l'exploitation, pour l'accélération de la mise en production des nouveaux logiciels, n'a aucun sens si les phases de tests demeurent des goulets d'étranglement.

La réponse des éditeurs de logiciels est donc centrée sur l'automatisation des différents types de tests, à partir de bibliothèques de scripts enregistrés et de leur utilisation optimisée en fonction des scénarios rencontrés, et notamment des composants concernés (IHM, logiciels unitaires, etc.). C'est ainsi que le PMU utilise, par exemple, l'atelier Neoload de Neotys pour une intégration continue sur la chaîne Code/Build/Test/Release/Operate de ses applications pour mobiles ou pour terminaux points de vente. « *Sans interrompre le run, les simulations de transactions - plusieurs milliers - sont lancées automatiquement, et l'ensemble de l'infrastructure est re-testée à chaque livraison* », explique Jonathan Fontana, responsable informatique. « *Cela permet d'anticiper sur les parcours avec des utilisateurs virtuels, d'automatiser la mise en place de contrôleurs et d'injecteurs dans les applications à tester. Grâce à cela, l'atelier peut fonctionner les week-ends et se relance sans intervention humaine en cas d'arrêts* ». Selon le PMU, les créneaux de tests disponibles ont été ainsi augmentés de 30 %, et plusieurs projets peuvent être testés en parallèle.

Sensibiliser les équipes agiles et DevOps aux nécessités du testing est l'autre combat du moment, comme l'explique Christophe Barbery, Delivery Manager chez Sogeti. Lors de la présentation de l'organisation mise en place pour vérifier en continu l'app Movin Blue développée avec Valéo pour proposer aux clients des loueurs de voitures des accès via leur smartphone aux véhicules réservés, il a relaté comment la question des tests de couverture réseau et de robustesse – entre autres – est devenue une préoccupation pour les équipes de développement. « *Nous sommes partis d'une organisation avec des testeurs externalisés, que nous avons progressivement sensibilisés aux concepts des méthodes agiles. En sens inverse, nous avons mis en place chez les développeurs des KPI, par exemple autourde leur capacité à livrer des composants logiciels qui passent du premier coup les tests* ».

ACPQualife confirme l'objectif, en présentant une pyramide des tests qui illustre à la fois les degrés d'automatisation décroissants selon qu'on parle de tests unitaires, de tests d'intégration ou encore de tests d'IHM, mais aussi de délais de feedback qui s'allongent sur le même axe. « *L'idée est d'utiliser au maximum des méthodes comme TDD* [test driven development, qui date des débuts de l'extreme programming en 1999, NDLR] *qui obligent les développeurs à écrire du code testable* », soulignent ainsi Laurent Bouhier et Cyril Tardieu. En matière de tests unitaires, il existe également d'autres méthodes (ATDD, BDD ou encore CDD) pour traiter des autres étapes de tests.

De quoi se noyer sous les possibilités peut-être ? C'est justement pourquoi Sogeti lance une première version de son outil Cognitive QA, qui met de l'IA dans la sélection des scénarios de tests à mettre en œuvre, en fonction du contexte des développements et même des profils des développeurs. Prometteur, mais encore largement en devenir, si l'on en croit le peu de fonctionnalités disponibles pour le moment.

FRANÇOIS JEANNE