ZENITY
Trust in quality

Non régression, une notion simple mais complexe

Les tests de non régression, symbolisés parfois TNR, sont les tests qui permettent de vérifier que des modifications n'ont pas entraîné d'effets de bord non prévus de nature à dégrader la qualité d'une version antérieurement validée. Contrairement à certaines idées reçues, cette nature de tests n'est pas propre aux phases de recette système et recette utilisateur. Ils concernent aussi la recette unitaire et la recette d'intégration de composants (ou recette d'assemblage) et interviennent lorsqu'une nouvelle version est produite afin de s'assurer que le système répond encore aux exigences spécifiées. Quelle est leur portée ? Les tests de non régression s'orientent sur une partie précédemment testée et validée conforme à certaines exigences. Ils ne portent donc jamais sur une partie non testée (les tests correspondants sont alors des tests de conformité), ni même sur une partie déjà testée et ayant donné lieu à des anomalies (les tests correspondants sont alors des tests de validation). Ces petits rappels peuvent paraître simplistes voire triviaux, néanmoins plus d'une aberration vont généralement à leur encontre. Comment planifier convenablement une recette fonctionnelle lorsqu'aucune recette de non régression unitaire n'a été opérée ? Comment ne pas dériver dans le planning de livraisons lorsque la notion même de non régression n'est pas correctement assimilée ?

Au sein des DSI cette notion est encore vague bien qu'utilisée très fréquemment, et est réellement complexe bien que considérée comme simpliste. Toute personne impliquée au sein d'une équipe projet se doit de la maîtriser. Développeur, analyste, concepteur, testeur, chef de projet, directeur de projet, chacun n'appréhendera pas la notion de la même manière, en fonction des intérêts qu'il a à en retirer, mais chacun a un lien, chacun aura un lien, plus ou moins proche avec la non régression.

Partant du principe qu'avant de comprendre la notion de régression, il faut comprendre les tourments qu'elle peut infliger sur les projets informatiques, débutons notre voyage par un retour aux sources originelles : les interprétations successives qui rythment la vie d'un projet et causent de nombreuses dérives. L'interprétation est comparable au mouvement des plaques tectoniques d'un projet. C'est aléatoire, propre à chaque acteur projet et difficilement prévisible. De ce phénomène résulte l'immersion de défauts qui, lorsqu'ils sont révélés (découverte d'anomalies), provoquent un séisme dont l'épicentre est l'anomalie elle-même et les dégâts les régressions aux alentours. Le phénomène simple pourrait s'arrêter là, mais il n'en est rien ! L'absence d'une politique de tests de non régression et d'analyse d'impacts rationnelle et structurée amplifie les dégâts en multipliant la découverte d'autres régressions : ce sont les répliques du séisme. La comparaison est imagée mais néanmoins pas si éloignée de la réalité… Il est courant de découvrir en phase de recette une importante anomalie de fonctionnement dont l'origine est une interprétation minime des spécifications fonctionnelles. Après correction de cette anomalie survient la première vague de dégâts : la régression de multiples composants environnants. Suivant le théorème bien connu des testeurs et dit de "concentration des défauts" se produit la première réplique : des effets de bord sont découverts autour de la première anomalie voire pire encore d'autres interprétations sont détectées, et ainsi de suite dans une fantastique réaction en chaîne pouvant déstabiliser n'importe quel projet.

Deux natures d'interprétations peuvent être isolées pour mieux comprendre la situation. La première se manifeste lors des phases de pré-codage, de l'analyse des besoins à la conception détaillée : l'interprétation du besoin fonctionnel ou métier. Ce risque d'interprétation d'un besoin diminue avec l'avancement du projet. En effet quel que soit le modèle projet utilisé, chaque phase dépend du résultat de la précédente : à partir d'un besoin sont produites des spécifications générales qui, elles-mêmes se déclinent en spécifications détaillées. Ainsi plus le cycle avance plus le formalisme détaillé affine le besoin exprimé et fournit une limitation naturelle à l'interprétation. Le problème est d'être certain que les déclinaisons successives de spécifications ne se basent pas sur une incompréhension initiale. Auquel cas la conséquence directe est la propagation de la divergence de compréhension. Le bilan stupéfiant mais néanmoins réaliste est que près de 70 % des anomalies d'un logiciel détectées depuis sa conception jusqu'à son utilisation en production sont issues de ces interprétations en pré-codage.

La seconde nature d'interprétation est quant à elle inhérente aux phases post-codage, de la recette unitaire à la recette utilisateur : l'interprétation du besoin en tests. La mauvaise définition de ce qui doit être testé ou non, et de ce qui doit être restesté ou non, a des impacts croissants avec l'avancement dans les phases de validation du cycle projet : un bout de code d'une méthode dans une classe a été modifié pour corriger une anomalie, que tester en recette utilisateurs ? Les sceptiques diront "tout", les inconscients diront "rien", les perdus diront "je ne sais pas, faisons quelques passes aléatoires" ! Le reflet de ces comportements c'est que près de 80% du total des anomalies d'un projet informatique sont détectées tardivement, en phase de recette utilisateur et pire encore en production… Pourquoi ? Dans ces deux dernières phases projet, le logiciel est complet et dans un environnement de données de production, les tests réalisés sont des flux de bout en

bout dévoilant ainsi tous les impacts enchevêtrés. En l'absence d'une stratégie de tests à tous les niveaux ISTQB, ce sont ces configurations de fin qui exacerbent les régressions complexes. Le constat c'est qu'à l'heure actuelle la majorité des projets est en réaction plutôt qu'en action, en mode curatif en lieu et place d'un mode préventif.

Considérons en sus de ces deux natures d'interprétations la courbe d'évolution du coût de correction d'une anomalie fonction de la phase projet dans laquelle elle est détectée. Ce coût débute à quelques dizaines d'euros en phase de spécification et est multiplié par 10 entre chaque phase pour atteindre plusieurs dizaines de milliers d'euros pour la correction d'une anomalie en production (incluant l'ensemble des activités : correction, mise en place d'une solution de contournement, helpdesk, assistance, etc.). Il ne reste plus qu'à combiner tous ces chiffres en saupoudrant le cocktail de statistiques sur l'injection des défauts et la détection des défaillances par le coût moyen de leur correction fonction de la phase projet, et on comprend aisément l'analogie au séisme et aux ravages que causent les interprétations et l'absence d'une politique structurée de tests de non régression au sein des DSI.

LES TYPES DE TESTS DE NON RÉGRESSION

Les tests de non régression ne peuvent se réduire à un ensemble figé à exécuter aléatoirement. Cet ensemble doit vivre, des tests doivent en rentrer, d'autres doivent en sortir à chaque livraison à tester. Qui plus est, chaque phase de test couvre un périmètre de plus en plus complet. Plus le projet avance, plus nombreuses sont les étapes de conception concernées, plus la campagne de non régression devient volumineuse. Le volume ne doit cependant pas être à lui seul un indicateur de la qualité des campagnes réalisées. Au contraire même et pour le comprendre il est essentiel de distinguer les deux types de tests de non régression. Le premier constitue un socle de garantie de bon fonctionnement des fonctions sensibles du système qui représentent le métier. Leur recensement et la mise en place de vérifications systématiques sont indispensables. Le second type de test de non régression est quant à lui bien plus mouvant et volatile. Il s'agit des tests qui sont intrinsèquement liés à la modification opérée, constituant par conséquent un ensemble plus circonspect. C'est plus précisément en considérant ces tests que le volume ne doit pas être un indicateur de qualité. Il est plus profitable d'analyser en amont les impacts potentiels pour dérouler une campagne éclair ciblée à la façon « blitzkrieg », plutôt que de ne se poser aucune question et exécuter une grosse quantité de tests de non régression pour se donner bonne conscience ! Non seulement ils ne seront pas pertinents et ont donc peu de chance d'être efficaces, mais en plus ce type d'actions, par nature rébarbative, a vocation à démobiliser les ressources qui ne voient aucune plus-value dans un rôle réduit à de l'exécution.

Comment déterminer le terrain de jeu idéal, ni trop volumineux ni trop restreint, des campagnes de non régression ? En effet, comment une Maîtrise d'Ouvrage peut-elle se concentrer à analyser les impacts d'une évolution ou d'une correction sur le système d'information si elle est engluée dans la réalisation de tests récurrents sans valeur ajoutée prouvée ? Une solution est de se tourner vers des outils. Certains outils du marché permettent de réaliser des analyses d'impacts afin de cibler précisément le périmètre des tests de non régression, et ainsi réduire au minimum les interprétations du besoin fonctionnel comme les interprétations du besoin en tests, toutes deux souvent issues d'un excès de confiance et/ou de savoir. C'est le cas dans le monde SAP du module BPCA (Business Process Change Analyzer) de Solution Manager qui, couplé avec HP ALM, indique avec précision les exigences de test impactées et par transitivité les cas de tests à réexécuter.

Une autre solution, organisationnelle cette fois, trouve naissance dans la constitution de binômes Expert métier ou technique / Expert test. Le test étant par définition une activité transverse aux silos métier et aux technologies, il est bien souvent difficile de trouver la personne idéale à la fois expert test, sachant métier et architecte de la technologie utilisée ! Il convient donc de transférer une partie du savoir métier et du savoir technique à la cellule de tests, ce qui permettra aux sachants de se recentrer sur leur cœur d'activité tout en accroissant la qualité des tests conçus. Et bien que le fondement du monde informatique soit le 0 et le 1, rien n'est binaire lorsqu'il s'agit d'organisation humaine ! Le rôle du test est de placer le curseur à l'endroit adapté entre les acteurs métier et les acteurs IT, fonction du contexte de l'entreprise, afin de prendre suffisamment de connaissance pour réaliser des tests sans pour autant nécessiter au préalable un transfert long et coûteux. Le traitement des spécificités ardues techniques et fonctionnelles restent néanmoins dans un premier temps à la charge des sachants, puis est transféré progressivement à moyen terme. Ce modèle de montée en compétence progressif et itératif joue en faveur d'un soulagement des tâches récurrentes des experts leur permettant ainsi d'analyser avec plus de précision chaque bouleversement qu'il soit correctif ou évolutif et de déterminer par conséquent avec exactitude le périmètre des tests de non régression à réaliser.

Silvain EVRARD
Certifié ISTQB Advanced Test Manager et co-fondateur de Zenity, société de services spécialisée dans les tests logiciels.