Fiche méthodologique de projet

T.2 Connaître d'autres tests

Objectif

Appréhender d'autres tests dans le développement logiciel.

Préambule

Il existe de nombreux tests dans le développement logiciel. Une présentation succincte des plus connus.

Méthode

Deux principes de base :

- Tests de **boîte noire** (test fonctionnel) :
Le test porte sur le fonctionnement externe du système. La façon dont le système réalise les traitements n'entre pas dans le champ du test.

- Tests de **boîte blanche** (test structurel) :
Le test vérifie les détails de l'implémentation, c'est à dire le comportement interne du logiciel.

Tests de conformité

Le test vérifie la conformité du logiciel par rapport à ses spécifications et sa conception.

Tests de non conformité

Le test vérifie que les "cas non prévus" ne perturbent pas le fonctionnement du système.

Tests bêta

Réalisés par des développeurs ou des utilisateurs sélectionnés, ils vérifient que le logiciel se comporte pour l'utilisateur final comme prévu par le cahier des charges.

Tests alpha

Le logiciel n'est pas encore entièrement fonctionnel, les testeurs alpha vérifient la pré-version.

Tests de non régression

Après chaque modification, correction ou adaptation du logiciel, il faut vérifier que le comportement des fonctionnalités n'a pas été perturbé, même lorsqu'elle ne sont pas concernées directement par la modification.

Tests fonctionnels

L'ensemble des fonctionnalités prévues est testé : fiabilité, performance, sécurité, affichages, etc

Test de Performance

Proche du Test de Charge, il s'agit d'un test au cours duquel on va mesurer les performances de l'application soumise à une charge d'utilisateurs. Les informations recueillies concernent les temps de réponse utilisateurs, les temps de réponse réseau et les temps de traitement d'une requête sur le(s) serveur(s). La nuance avec le type précédent réside dans le fait qu'on ne cherche pas ici à valider les performances pour la charge attendue en production, mais plutôt vérifier les performances intrinsèques à différents niveaux de charge d'utilisateurs.

Test de Charge

Il s'agit d'un test au cours duquel on va simuler un nombre d'utilisateurs virtuels prédéfinis, afin de valider l'application pour une charge attendue d'utilisateurs. Ce type de test permet de mettre en évidence les points sensibles et critiques de l'architecture technique. Il permet en outre de mesurer le dimensionnement des serveurs, de la bande passante nécessaire sur le réseau, etc.

Test de Robustesse, d'endurance, de fiabilité

Il s'agit de tests au cours duquel on va simuler une charge importante d'utilisateurs sur une durée relativement longue, pour voir si le système testé est capable de supporter une activité intense sur une longue période sans dégradations des performances et des ressources applicatives ou système. Le résultat est satisfaisant lorsque l'application a supporté une charge supérieure à la moitié de la capacité maximale du système, ou lorsque l'application a supporté l'activité d'une journée ou plusieurs jours/mois/années, pendant 8 à 10 heures, sans dégradation de performance (temps, erreurs), ni perte de ressources systèmes.

Test de capacité
Test de montée en charge

Il s'agit d'un test au cours duquel on va simuler un nombre d'utilisateurs sans cesse croissant de manière à déterminer quelle charge limite le système est capable de supporter. Éventuellement, des paramétrages peuvent être effectués, dans la même logique que lors des tests de dégradation, l'objectif du test étant néanmoins ici de déterminer la capacité maximale de l'ensemble système-applicatif dans une optique prévisionnelle.

Test aux limites

Il s'agit d'un test au cours duquel on va simuler en général une activité bien supérieure à l'activité normale, pour voir comment le système réagit aux limites du modèle d'usage de l'application. Proche du test de capacité, il ne recouvre pas seulement l'augmentation d'un nombre d'utilisateurs simultanés qui se limite ici à un pourcentage en principe prédéfini, mais aussi les possibilités d'augmentation du nombre de processus métier réalisés dans une plage de temps ou en simultané, en jouant sur les cadences d'exécutions, les temps d'attente, mais aussi les configurations de la plateforme de test dans le cadre d'architectures redondées (Crash Tests).

Test de Dégradations des Transactions

Il s'agit d'un test technique primordial au cours duquel on ne va simuler que l'activité transactionnelle d'un seul scénario fonctionnel parmi tous les scénarios du périmètre des tests, de manière à déterminer quelle charge limite simultanée le système est capable de supporter pour chaque scénario fonctionnel et d'isoler éventuellement les transactions qui dégradent le plus l'ensemble du système. Ce test peut tenir compte ou non de la cadence des itérations, la représentativité en termes d'utilisateurs simultanés vs. sessions simultanées n'étant pas ici un objectif obligatoire, s'agissant ici plutôt de déterminer les points de contention générés par chaque scénario fonctionnel. La démarche utilisée est d'effectuer une montée en charge linéaire jusqu'au premier point de blocage ou d'inflexion. Pour dépasser celui-ci, il faut paramétrer méthodiquement les composants systèmes ou applicatifs afin d'identifier les paramètres pertinents, ce jusqu'à obtention de résultats satisfaisants.
Méthodologiquement, ce test est effectué avant les autres types de tests tels que performance, robustesse, etc. où tous les scénarios fonctionnels sont impliqués.

Test de stress

Il s'agit d'un test au cours duquel on va simuler l'activité maximale attendue tous scénarios fonctionnels confondus en heures de pointe de l'application, pour voir comment le système réagit au maximum de l'activité attendue des utilisateurs. La durée du palier en pleine charge, en général de 2 heures, doit tenir compte du remplissage des différents caches applicatifs et clients, ainsi que de la stabilisation de la plateforme de test suite à l'éventuel effet de pic-rebond consécutif à la montée en charge. Dans le cadre de ces tests, il est possible de pousser le stress jusqu'à simuler des défaillances systèmes ou applicatives afin d'effectuer des tests de récupération sur incident (Fail-over) ou pour vérifier le niveau de service en cas de défaillance.