Tests de performance à la vitesse d'Agile

Livre blanc de Neotys

Sommaire

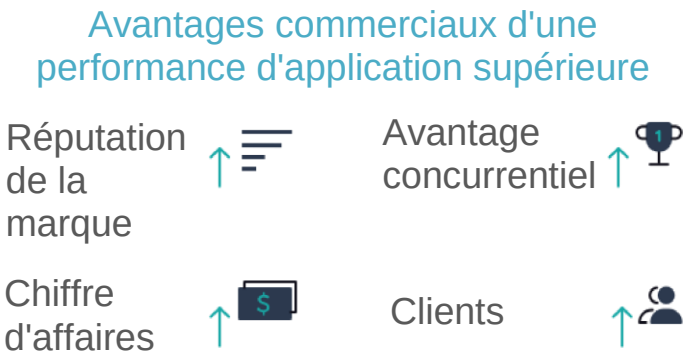
Présentation

Ce livre blanc a pour but de mettre en valeur certains des défis liés aux tests de charge et de performance dans un environnement Agile. Il fournit également de bonnes pratiques clés, telles que la hiérarchisation des objectifs de performance et l'automatisation des tests via des serveurs d'intégration continue. Les responsables des tests, les opérateurs et/ou toute personne impliquée dans les tests de développement Agile ou similaires peuvent bénéficier de ces conseils.

Admettons-le : la stratégie Agile est une réalité de la vie. Vous n'êtes peut-être pas « complètement Agile » et n'exécutez peut-être pas l'intégration continue, ni même ne parlez encore de DevOps, mais le fait est que la pression se fait toujours plus grande pour obtenir un grand nombre des avantages (tels que la qualité et la vitesse) inhérents aux méthodes de développement Agile ou similaires.

Lorsque vous devenez plus Agile, les développeurs rédigent plus de code, plus rapidement, dans le but de prendre en compte autant de tâches et de récits utilisateur que possible avant la fin du sprint, alors que de nombreux testeurs peinent à suivre ce rythme. En outre, les testeurs des équipes Agile sont souvent responsables des tests automatisés, unitaires et de régression, en plus des tests de charge et de performance. Dans cet environnement, vous devez pouvoir suivre le rythme de développement tout en répondant aux attentes accrues de qualité.

Les avantages techniques qui découlent d'un développement Agile sont bien documentés (vitesse supérieure de mise sur le marché, adaptation aux demandes changeantes des utilisateurs/nouvelles technologies, boucle de feed-back constante, etc.), mais les avantages commerciaux qu'apportent des performances d'application supérieures en raison d'une plus grande rigueur des tests de charge et de performance jouent également un rôle important.



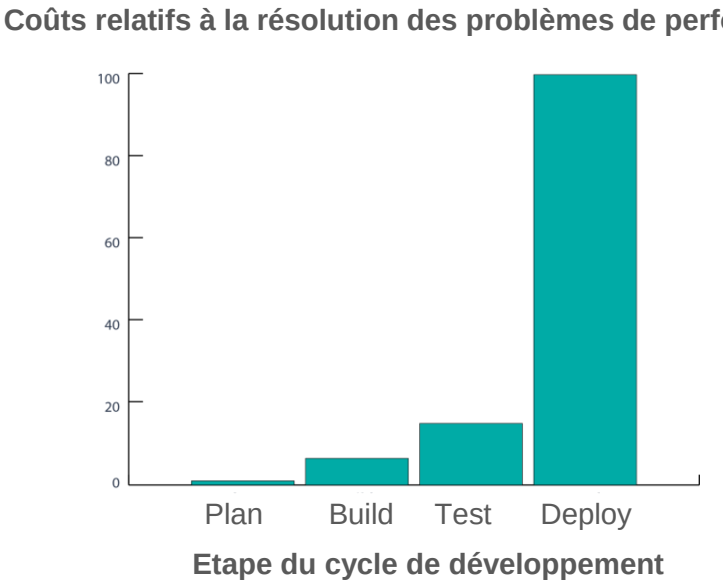
Avantages liés aux tests continus de charge et de performance

Éviter la découverte tardive de problèmes de performance

Lorsque les tests de charge et de performance sont repoussés à la fin d'un cycle de développement, les développeurs ont souvent peu ou pas de temps pour apporter des modifications. Cela peut obliger les équipes à retarder la sortie d'un produit, empêchant ainsi la livraison en temps opportun des fonctionnalités nécessaires aux clients. Alternativement, si les problèmes s'avèrent mineurs, les équipes peuvent décider de poursuivre le processus et de lancer l'application en production, en acceptant les risques associés. Si les problèmes de performance sont plus fondamentaux, ils peuvent nécessiter des changements architecturaux douloureux, dont la mise en œuvre peut prendre des semaines ou des mois.

Apporter des modifications plus tôt, quand leur coût est moindre

En incorporant des tests de charge et de performance dans les processus de test d'intégration continue, les entreprises peuvent détecter relativement tôt les problèmes de performance, alors que les coûts des correctifs sont beaucoup plus gérables. Si les développeurs peuvent immédiatement savoir que la nouvelle fonctionnalité de la dernière build a empêché l'application de respecter les contrats de niveau de service (SLA), ils peuvent résoudre le problème avant que son coût augmente de façon exponentielle. Ceci est particulièrement vrai pour les équipes Agile qui découvrent un problème de performance après plusieurs semaines (pour un problème déjà présent dans plusieurs builds antérieures). Cela accentue la difficulté à identifier la cause du problème.



Garantir aux utilisateurs d'obtenir de nouvelles fonctionnalités, et non de nouveaux problèmes de performances

Dans certaines organisations Agile, le changement se produit rapidement. Il est possible qu'une nouvelle fonctionnalité ou caractéristique soit vérifiée dans le cadre du contrôle de code source, soit intégrée dans une build d'intégration continue, passe tous les tests automatisés et soit déployée sur le serveur de production, en quelques minutes. Toutefois, si le code n'est pas optimisé pour gérer le nombre d'utilisateurs simultanés observés aux heures de pointe les plus élevées, une défaillance système peut survenir. L'intégration des tests de charge dans le processus avant que ces modifications soient déployées en production peut garantir que vos utilisateurs bénéficieront de tous les avantages offerts par votre marque/technologie, sans aucun compromis. Cela peut éviter à votre entreprise de subir une grosse perte de chiffre d'affaires (chiffrée en milliers voire en millions d'euros) alors que les utilisateurs se tourneraient vers les applications de vos concurrents ou dénigreraient votre marque en raison de problèmes qu'ils auraient rencontrés avec votre application.

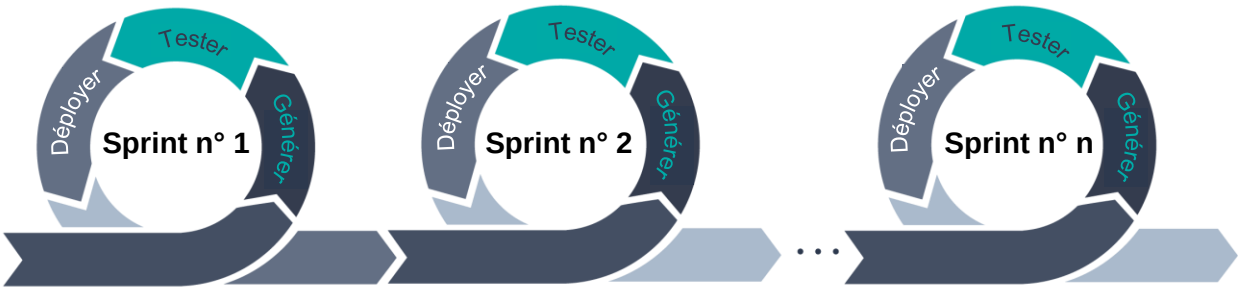
Défis propres aux tests de performance dans un environnement Agile

De la même manière que l'association d'Agile aux tests de charge peut offrir des avantages uniques, elle peut également présenter des défis majeurs que vos équipes n'ont peut-être encore jamais rencontrés.

Des cycles de développement plus courts nécessitent plus de tests en moins de temps

Les tests de charge et de performance sont généralement repoussés à la fin d'un cycle de développement. Avec un développement Agile, les cycles sont beaucoup plus courts et les tests de charge et de performance peuvent être retardés jusqu'au dernier jour d'un sprint ou parfois menés dans un mode de sprint différent. Cela peut souvent aboutir à la publication prématurée d'un code insuffisamment testé et/ou au report de récits utilisateur à la prochaine version, une fois testés. Théoriquement, la solution consiste à incorporer les tests plus tôt dans le cycle de développement, mais cela est plus facile à dire qu'à faire quand de nombreuses équipes manquent de ressources et d'outils pour cela.

Cycle méthodologique Agile



Cycle méthodologique en cascade

Un code « opérationnel » n'est pas toujours performant

Les développeurs travaillant dans les équipes Agile mettent fortement l'accent sur la livraison d'un code « opérationnel », mais ce code est-il réellement « opérationnel » s'il échoue en charge ? Les tâches/récits utilisateur doivent-ils être marqués comme « terminés » si le code associé entraîne le blocage de l'application pour 100 utilisateurs ? Qu'en est-il pour 1 000 ? 100,000? La pression pour conclure le développement du code est élevée, mais il en va de même du coût d'un blocage de l'application en production.

Les développeurs ont besoin d'un feed-back en temps opportun … sans délai !

Les développeurs Agile doivent en savoir plus que le simple fait que leur code cause des problèmes de performance : ils doivent savoir quand les problèmes ont commencé et sur quel récit ils travaillaient à ce moment-là. Il est très pénible pour des développeurs d'avoir à revenir en arrière afin de corriger du code pour un récit traité des semaines ou des mois auparavant. Cela signifie également qu'ils ne disposent pas du temps nécessaire pour mettre sur le marché de nouvelles fonctionnalités. La détection des problèmes de performance à un stade précoce du cycle, permettant de fournir rapidement un feed-back important aux développeurs, est essentielle pour réduire les coûts.

Automatiser la livraison du développement aux opérations peut paraître risqué

Alors que DevOps et le déploiement continu sont des pratiques encore assez jeunes, les équipes ops ont toujours craint que de nouveaux changements dans le code ralentiront voire bloqueront l'application lorsqu'elle sera déployée en production. L'automatisation de certains tests dans le processus d'intégration continue permet d'atténuer cette crainte, mais si des tests de performance appropriés ne sont pas inclus, le risque reste réel. Les équipes ops connaissent bien l'énorme impact que les temps d'arrêt des applications peuvent avoir sur l'entreprise.

Bonnes pratiques

Les bonnes pratiques suivantes peuvent vous aider à optimiser les avantages tout en surmontant les défis liés aux tests de charge dans un environnement Agile.

Placer les contrats SLA de performances au centre des préoccupations

Chaque application a des contrats de niveau de service (SLA) définissant des performances minimales à respecter, mais les équipes Agile sont souvent plus concentrées sur l'ajout de fonctionnalités/caractéristiques que sur l'optimisation des performances de l'application. Les récits utilisateur sont généralement écrits d'un point de vue fonctionnel (p. ex., « En tant qu'utilisateur, je peux cliquer sur le bouton "Afficher le panier" et consulter la page "Mon panier" ») sans spécifier les exigences de performance de l'application (p. ex., « En tant qu'utilisateur, je peux cliquer sur le bouton "Afficher le panier" et consulter la page "Mon panier" » en moins d'une seconde quand il y a moins de 1 000 autres utilisateurs sur le site »).

Ce n'est peut-être pas la meilleure façon d'écrire un récit utilisateur, mais cela illustre l'idée que les performances doivent figurer quelque part sur le tableau des tâches si l'équipe est censée y accorder de l'attention.

Une façon d'afficher les performances sur le tableau consiste à utiliser vos contrats SLA comme des tests d'acceptation pour chaque récit, de sorte qu'un récit ne puisse pas être « terminé » si les modifications entraînent le non-respect de ces contrats SLA par l'application (cela peut nécessiter la définition d'un ou de plusieurs nouveaux contrats SLA pour de nouvelles fonctionnalités/applications, par exemple : toutes les recherches d'une nouvelle fonctionnalité de recherche doivent renvoyer des résultats en moins de 2 secondes). Cette approche fonctionne bien lorsque les modifications apportées au récit affectent une partie relativement restreinte du code et que les problèmes de performance sont, par conséquent, limités à une partie de l'application.

Pour les contrats SLA portant sur l'*ensemble* de l'application (p. ex., chaque page doit être chargée en moins d'une seconde), les tests doivent être ajoutés à une plus grande liste de contraintes (pouvant inclure des tests fonctionnels) et être testés pour chaque récit, afin de déterminer si ce récit répond à la DoD (Definition of Done) minimale sans manquer à aucune de ces contraintes.

Travailler en étroite collaboration avec les développeurs pour anticiper les changements

L'un des avantages pour les testeurs qui travaillent dans un environnement Agile est qu'ils sont généralement informés des mises à jour des tâches de développement au cours d'échanges quotidiens ou de réunions de type scrum similaires. Afin de retirer un bénéfice maximum de ce niveau de collaboration, les testeurs doivent constamment réfléchir à la façon dont les récits en cours de codage seront finalement testés. Exigeront-ils de nouveaux tests de charge ? Provoqueront-ils des erreurs dans les scripts de test actuels ? Pouvez-vous vous permettre de légères modifications des scripts de test actuels si vous planifiez à l'avance ? La plupart du temps, il s'agit de petites modifications, si bien que les testeurs peuvent conserver leur avance s'ils restent en contact avec l'équipe.

Intégrer au serveur de build

Même si vous n'avez pas encore complètement adopté les méthodes Agile, vous avez probablement un serveur de build qui lance des tests automatisés, unitaires, de fumée et/ou de régression. De la même manière que les objectifs de performance doivent être ajoutés au tableau des tâches, les tests de performance doivent faire partie des tests récurrents pour chaque build. Cela peut se résumer à la configuration d'un déclencheur pour amener le serveur de build à lancer le test et à afficher les résultats de test dans l'outil de génération en fonction de la sophistication de l'intégration. Dans l'idéal, vous voulez que la personne qui a lancé la génération voie instantanément les résultats et sache quels changements ont été incorporés à cette build afin qu'ils puissent être corrigés en cas de problème de performance.

Intégration continue + Build nocturne + Tests de charge en fin de sprint

Il peut y avoir de grandes différences entre des builds d'intégration continue, des builds nocturnes et des builds post-sprint. Nous parlons des différences entre une modification individuelle apportée à un serveur de contrôle de versions, toutes les modifications effectuées dans une journée et toutes les modifications effectuées au cours d'un sprint. Dans cet esprit, vous devez ajuster vos tests de charge au type de build que vous exécutez.

Une bonne pratique consiste à commencer petit et en interne. Pour les builds d'intégration continue qui sont générées chaque fois que quelqu'un apporte une modification, vous voulez que ces tests soient exécutés rapidement, afin de pouvoir présenter au développeur des résultats sur la façon dont ses modifications affectent le système. Pensez à exécuter un petit test de performance avec les scénarios les plus courants, en appliquant à votre application la charge standard produite par vos propres générateurs de charge internes.

Pour les builds nocturnes, augmentez le nombre de scénarios à contraintes multiples et augmentez la charge pour obtenir ce que vous voyez aux heures de pointe, afin de constater si des problèmes de performance sont passés inaperçus durant les tests d'intégration continue. À la fin du sprint, vous voudrez sortir le « grand jeu » : envisagez de générer une charge à partir du cloud pour voir ce qui se passe quand des utilisateurs accèdent à votre application via le pare-feu. Veillez à ce que chaque contrat SLA de la liste des contraintes soit adopté, afin que chaque récit terminé au cours du sprint puisse être marqué comme « terminé ».

Procédure pour choisir une solution de test de charge Agile

Toute solution de test de charge vous permettra d'effectuer certaines formes de tests de charge dans votre environnement Agile, mais peu vous permettent de suivre toutes les bonnes pratiques indiquées ici tout en suivant le rythme de vos équipes de développement Agile.

Lorsque vous envisagez une solution de test de charge Agile, vous devriez vous poser les questions suivantes :

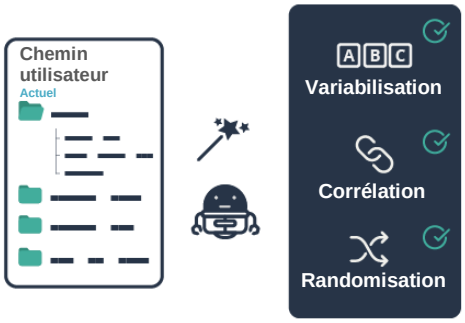
- Avec quelle rapidité les tests peuvent-ils être conçus dans l'outil par un testeur ou un développeur quelconque (et non par vos experts en tests de performance) ?
- Mes équipes pourront-elles partager des scénarios/résultats de test entre elles et avec d'autres intervenants ?
- Dans quelle mesure l'outil s'intègre-t-il aux serveurs d'intégration continue pour automatiser le processus de test ?
- Chaque membre de mes équipes sera-t-il capable de créer et de comprendre les rapports de test et d'agir conformément ?
- Les conditions de test seront-elles capables de reproduire ce qui se passe en production ?
- La solution prend-elle en charge les technologies que nous avons utilisées pour générer l'application et les technologies que nous envisageons d'utiliser à l'avenir ?

Conception de test simple avec automatisation

Les équipes Agile disposent rarement d'un ingénieur Performance dédié qui intervient à la demande pour élaborer le script d'un nouveau test lorsque les modifications apportées à une application provoquent des erreurs dans l'ancien test. Vous pouvez avoir un ou deux testeurs dans votre équipe, mais il y a peu de chance qu'ils soient experts en performances. Dans certains cas, vous pouvez disposer de développeurs qui écrivent leurs propres tests. Dans tous ces scénarios, ces membres d'équipe seront pressés par le temps, ce qui ne laisse pas de place pour un outil difficile à utiliser et à configurer.

Dans le cadre du développement et de l'exécution de tests de performance, plusieurs fonctionnalités clés contribuent grandement à améliorer la productivité des tests, notamment la prise en charge des éléments suivants :

- Lancement facile de l'enregistrement d'un profil utilisateur virtuel (de préférence en un seul clic)
- Définition de comportements avancés (avec des structures telles que des conditions/boucles) via une interface graphique
- Gestion automatique des paramètres dynamiques. Cela inclut un ensemble de règles de corrélation pour les infrastructures de serveurs connues. Dans l'idéal, la solution détectera et traitera de façon dynamique les paramètres personnalisés spécifiques à votre application.
- Partage de parties de script courantes, telles que les transactions de connexion et déconnexion, entre plusieurs profils utilisateurs virtuels



Conception très rapide de tests de charge à l'aide d'assistants et d'une automatisation

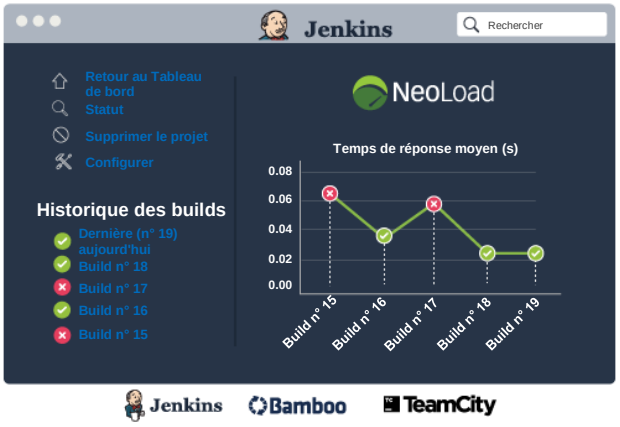
Cela ne se veut pas une liste exhaustive des fonctionnalités d'utilisation pouvant aider les membres d'équipe à travailler plus efficacement, mais devrait plutôt être considéré comme le socle des capacités minimales requises pour fournir une solution de test de charge efficace.

Capacités de collaboration

La collaboration est toujours au cœur des activités d'une équipe Agile, et vous voudrez probablement partager des éléments tels que les profils utilisateurs virtuels, les groupes, les configurations de supervision, les profils de charge et les résultats des tests avec les membres de l'équipe. Vous devriez examiner les solutions de test de charge qui permettent à plusieurs membres d'équipe de s'impliquer dans la création et l'analyse de cas de test et qui permettent aux testeurs de partager rapidement les résultats avec les développeurs pour les aider à comprendre les dysfonctionnements et les corrections à apporter.

Intégration à des serveurs d'intégration continue

Vous devez envisager une solution de test de charge qui s'intègre aux serveurs d'intégration continue, afin de pouvoir automatiser le déclenchement d'un test de performance pour chaque build et de passer en revue rapidement les résultats des tests (ainsi que ceux des autres tests lancés pendant la génération) pour fournir un feed-back instantané au développement. La capacité à suivre les tendances de performance sur plusieurs builds est essentielle pour les tests de régression de performance. Les testeurs doivent pouvoir identifier de façon exacte les modifications de code et de build qui ont initié une tendance à la dégradation des performances.



Intégration à des serveurs d'intégration continue tels que Jenkins, Bamboo et TeamCity

L'automatisation de l'interaction entre le serveur d'intégration continue et les tests de charge est essentielle pour garantir que les performances constitueront un objectif standard pour chaque itération. Lorsque les tests de charge sont uniquement manuels dans le cadre des équipes Agile, il est aisé de les reporter au sprint suivant.

Rapports faciles à générer et à comprendre

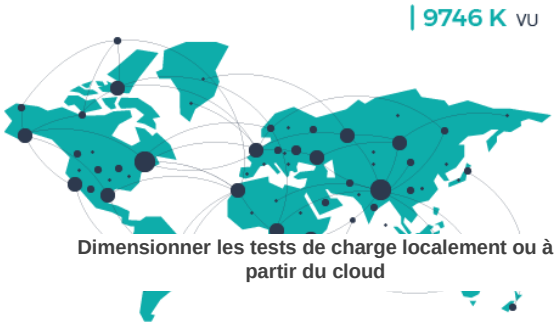
Comme tous les membres de votre équipe ne sont pas des experts en performances, vous devez rechercher des solutions de test de charge qui fournissent des résultats faciles à appliquer, comprendre et prendre en compte. Cela inclut des rapports que les développeurs peuvent interpréter pour apporter des modifications importantes au code, que les équipes ops peuvent utiliser pour modifier les configurations d'infrastructure et que la direction peut lire pour obtenir un aperçu des performances globales. Des fonctionnalités telles que les métriques de glisser-déposer et le filtrage réduisent considérablement le temps nécessaire à la création de rapports. Étant donné que les besoins en matière de création de rapports changent, il est judicieux de conserver toutes vos options avec un outil prenant en charge plusieurs formats, y compris les formats PDF, Word, HTML et XML, pour l'intégration avec d'autres systèmes.

Tests réalistes

La plupart des applications web et mobiles d'aujourd'hui seront utilisées par des utilisateurs via de nombreux sites/appareils, au moyen de divers navigateurs, dans des conditions réseau variées. Vous devriez rechercher une solution de test de charge capable d'effectuer les actions suivantes :

- **Émulation de conditions de réseau réelles.** Lorsque vous choisissez une solution de test de charge, recherchez-en une qui offre une émulation de réseau étendu pouvant limiter la bande passante/simuler la latence et la perte de paquets pour garantir que les utilisateurs virtuels téléchargeront le contenu de l'application web selon un débit réaliste. Cette fonctionnalité est particulièrement importante lors de tests d'applications mobiles, car les appareils mobiles utilisent généralement une bande passante moindre que les ordinateurs portables et les ordinateurs de bureau, et ils sont fortement affectés par des variations de latence et de perte de paquets, notamment quand le signal est faible.

- **Prise en charge des appareils et des navigateurs.** De façon similaire, recherchez une solution capable d'enregistrer à partir de n'importe quel navigateur ou appareil mobile, et de les simuler en retour au cours des tests de charge. La simulation des appareils est importante car vous avez besoin du nombre de connexions parallèles et du contenu de serveur appropriés pour obtenir une charge de serveur et des temps de réponse réalistes. De plus, les navigateurs modernes ont la capacité de mettre en parallèle les requêtes HTTP lorsqu'ils récupèrent les ressources statiques d'une page web. Ces requêtes parallèles nécessitent plus de connexions avec le serveur et peuvent allonger les temps de réponse. Les solutions de test de charge qui ne mettent pas en parallèle les requêtes sont incapables de produire des tests de performance vraiment réalistes pour les applications web.



Dimensionner les tests de charge localement ou à partir du cloud

- **Tests à l'échelle réalisés depuis l'extérieur du pare-feu, globalement.** La très grande majorité des applications web et mobiles sont accessibles aux utilisateurs situés à l'extérieur du pare-feu. Bien qu'il soit absolument essentiel de pouvoir effectuer des tests en interne dans un environnement Agile, afin de comprendre réellement l'impact de l'emplacement sur les performances de vos utilisateurs, vous devez envisager une solution capable de générer une charge à partir de serveurs cloud dans le monde entier. Vous pouvez également envisager une solution qui offre la possibilité de générer une charge à partir de plusieurs fournisseurs cloud, afin de réduire le risque lié au fait de s'appuyer sur une seule source de génération de charge cloud.

Prise en charge des dernières technologies

Pour tester des applications générées à l'aide des technologies Adobe Flex, Microsoft Silverlight, RTMP (Real-Time Messaging Protocol), AJAX Push, HTML5 ou WebSocket, vous avez besoin d'un outil de test de charge intégrant la prise en charge de toutes les technologies que vous utilisez. Sans cette prise en charge spécialisée, il peut s'avérer très difficile, voire impossible, de tester efficacement les performances de vos applications.

Enregistrement et replay de tout type d'applications : applications complexes, mobiles, IoT

Cela peut être particulièrement problématique pour les équipes Agile qui ajoutent de nouvelles fonctionnalités/caractéristiques à un rythme rapide, car les développeurs veulent souvent utiliser les dernières technologies disponibles. D'un côté, vos développeurs peuvent expérimenter avec Google SPDY et, d'un autre côté, ils peuvent ajouter des vidéos en streaming à une application. Vous devez savoir que votre solution de test de charge sera capable de tester comment ces changements affecteront les performances.

Commentaires de nos utilisateurs sur NeoLoad

« NeoLoad a été choisi en raison de ses fonctionnalités équilibrées, de sa facilité d'utilisation et de son coût ... LoadRunner utilise un modèle tarifaire totalement déraisonnable. » – Brian Daniel, ingénieur QA, ICW Group

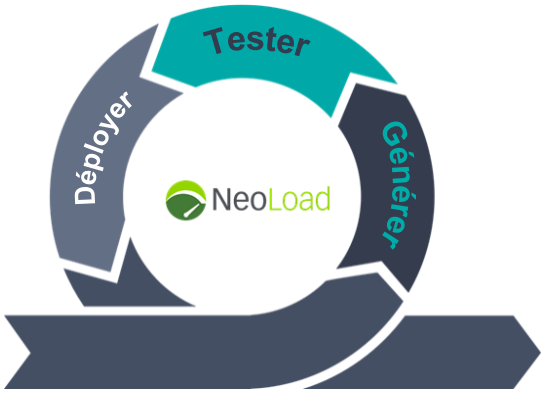
« Nous avons bénéficié d'une réduction des coûts lorsque nous avons adopté NeoLoad (pour remplacer LoadRunner), mais peut-être plus important encore, nous avons pu réduire les risques lors de la publication de nouveaux logiciels. En générant un plus grand nombre d'utilisateurs virtuels et en prenant en charge la technologie PUSH, NeoLoad nous a permis d'élaborer des scénarios de charge beaucoup plus réalistes. » – Adam Dance, responsable QA, IG Group

« Nous avons abandonné LoadRunner en raison de facteurs divers, dont le coût, la facilité d'utilisation et la robustesse des fonctionnalités. Nous pouvons utiliser NeoLoad pour étendre rapidement les tests de performance à toutes nos applications métier critiques et pour résoudre certains problèmes de performance clés. » - Georges Lauture, superviseur adjoint, Automatisation QA, Assurance qualité, NJM Insurance Group

« ... les développeurs étaient pleinement opérationnels après trois jours seulement, alors que cela avait pris plusieurs semaines avec LoadRunner ... Il s'agit d'une avancée importante, car le même test aurait pris trois ou quatre jours avec notre ancien outil, ce qui est incompatible avec la fréquence à laquelle nous publions de nouvelles versions ... À présent, BCBS réalise des tests de performance sur une base hebdomadaire ... » – Corey Bradley, consultant-analyste AppSystems, BlueCross BlueShield of Tennessee

Conclusion

Les test continus et le développement Agile augmentent la productivité des équipes et la qualité de leurs applications. Lors de l'ajout de tests de charge et de performance dans ces processus, une planification minutieuse doit être effectuée pour garantir que les performances sont une priorité à chaque itération. L'objectif de réaliser des tests de charge à la vitesse d'Agile est de fournir une valeur applicative maximale aux utilisateurs via l'évolution des fonctionnalités et des caractéristiques tout en garantissant les performances indépendamment du nombre d'utilisateurs simultanés.



Pour vous assurer de tirer le meilleur parti de l'association des méthodologies Agile et des tests de charge, vous devez :

- Vous assurer que les contrats SLA de performances figurent sur votre tableau de tâches/liste de contraintes pour garantir que le code associé à une tâche s'exécute bien avant que cette tâche soit marquée comme « terminée »
- Collaborer avec les développeurs pour anticiper les modifications de code qui nécessitent des changements dans les scénarios de test de performance
- Organiser le lancement automatique de tests de performance pour chaque nouvelle build et suivre les tendances des performances de build en build
- Augmenter la complexité et la charge des tests avec la taille de la build pour conserver de faibles durées de génération (Intégration continue => test de fumée de performance, Build nocturne => test de charge complet, Fin de sprint => test de contraintes avec générateurs de charge cloud)

En choisissant une solution de test de charge pour le développement Agile, vous voulez un outil capable de s'adapter aux besoins de votre équipe et de votre application. La solution doit être capable de :

- Permettre aux testeurs de performance novices et experts de concevoir et de modifier rapidement des scénarios de test
- Permettre à tous les membres de l'équipe de partager des scénarios de test et des résultats de test
- S'intégrer complètement aux serveurs d'intégration continue afin de lancer des tests et d'afficher les tendances des résultats sur plusieurs builds, pour permettre des tests de régression
- Produire des rapports que tous les membres de l'équipe pourront comprendre et appliquer
- Exécuter des tests qui ressemblent au monde réel en tenant compte des appareils des utilisateurs, des navigateurs, des conditions réseau et des emplacements géographiques
- Prendre en charge les protocoles et les autres technologies que vous utilisez pour générer vos applications maintenant et dans le futur

À propos de Neotys

Le succès de votre stratégie numérique repose sur votre capacité à livrer régulièrement des logiciels rapides et fiables. La création rapide de bons logiciels, à l'aide d'un processus optimisé de test de performance constitue votre avantage concurrentiel, et Agile et DevOps font partie de cette solution.

Neotys a investi plus de 12 ans de développement dans NeoLoad, la plateforme de test de performance conçue pour accélérer les processus Agile et DevOps. Elle a été élaborée par des ingénieurs qui ont reconnu que, pour atteindre leur propre objectif d'adoption d'Agile, ils devaient créer un produit favorisant des tests continus de charge et de performance plus efficaces. Le résultat final est une création et une maintenance de tests jusqu'à 10 fois plus rapides avec NeoLoad.

Le résultat final est une création et une maintenance de tests jusqu'à 10 fois plus rapides avec NeoLoad.

Nous croyons vraiment que l'ingénieur Performance peut devenir le partenaire clé de la performance des applications qui assurera la meilleure couverture de test tout en respectant la cadence du processus de livraison continue. Les performances devenant la responsabilité de l'ensemble de l'équipe, nous travaillons quotidiennement à fournir une plateforme de tests de performance optimisée.

Pour plus d'informations sur Neotys et NeoLoad, visitez : www.neotys.com ou contactez sales@neotys.com