Cell' INSIGHTS

From 0 To Hero

DEVOPS

de la vision à l'implémentation

#1 Nov 2014

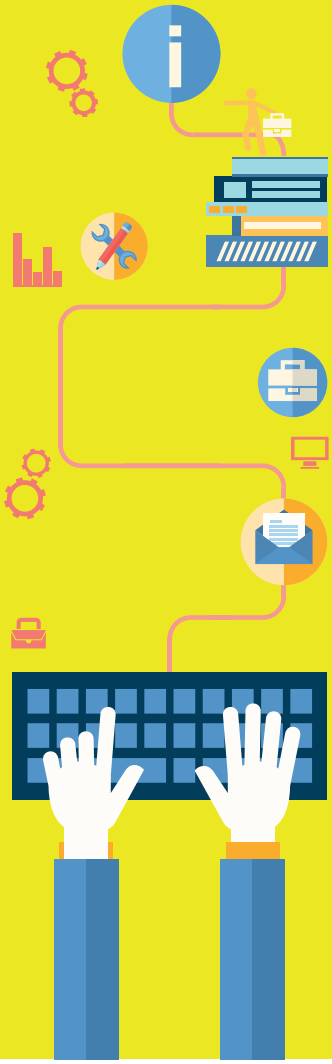
Cellenza

Design graphique : micheletmichel.com
Impression : megatop.fr

DEVOPS

de la vision à l'implémentation

SOMMAIRE



INTRODUCTION

C'est dans les grands principes du Lean Management, et plus précisément dans l'amélioration continue des processus, qu'apparaît le DevOps. Cette nouvelle approche, qui s'applique essentiellement au domaine du développement logiciel, prône l'efficacité et la productivité au sein des équipes projet, en y intégrant pleinement les rôles opérationnels traditionnellement dévolus à des équipes dédiées et séparées des développeurs.

Depuis plusieurs années, les divergences apparentes entre le métier de développeur et celui d'opérationnel ralentissent considérablement l'efficacité des équipes de développement logiciel ce qui ne contribue qu'à retarder la mise à disposition des produits aux clients finaux. Que l'on soit développeur ou opérationnel, nous avons dû faire face, au moins une fois, à une situation conflictuelle opposant le développeur et l'opérationnel. Ces divergences sont une conséquence directe de la différence de nature des deux métiers. Le rôle du développeur est d'apporter du changement dans un système par l'ajout de nouvelles fonctionnalités, celui de l'opérationnel est de maintenir ces mêmes systèmes stables. C'est donc précisément à cette problématique que nous tenterons de répondre grâce à l'approche DevOps.

Nous allons tenter, dans un premier temps, de définir, sous plusieurs angles, ce qu'est DevOps, en analysant les changements organisationnels structurants et nécessaires à l'adoption de DevOps; puis, nous nous intéresserons aux bonnes pratiques DevOps; pour finir, nous parlerons du paysage technologique proposé par Microsoft qui facilite l'adoption de DevOps en fournissant des outils permettant la mise en œuvre de ces pratiques.

La seconde partie sera composée d'une plongée plus profonde dans les outils Microsoft afin d'illustrer leur mise en œuvre dans le cadre d'une adoption de DevOps au sein d'une entreprise.

1

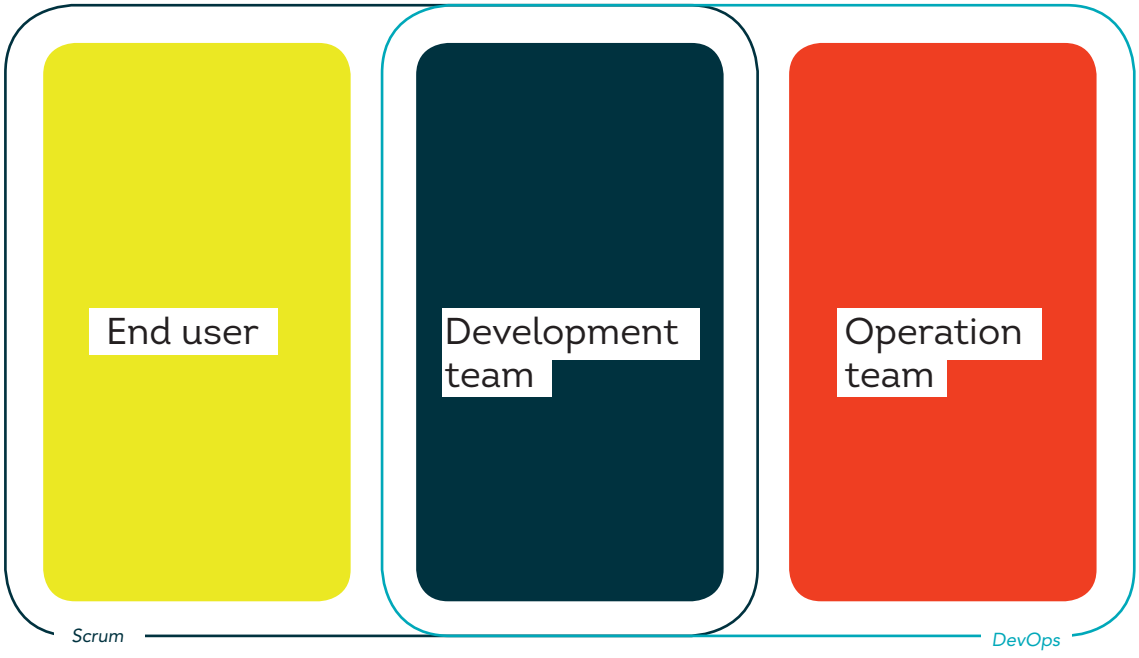
PREMIÈRE PARTIE

- Pourquoi faire du DevOps ?
- Quelles sont les principales étapes pour adopter DevOps ?
- Collaboration
- Time-to-Market
- Déploiement continu
- Changements organisationnels
- Métrique
- Outils
  - Team Foundation Server
  - Release Management
  - Desired State Configuration

POURQUOI FAIRE DU DEVOPS?

Les méthodologies Agiles et notamment SCRUM, pour ne citer qu'un exemple, ont permis de réduire considérablement l'écart entre le métier et l'IT. L'introduction de nouveaux rôles tels que le SCRUM Master ou le Product Owner, de nouveaux outils comme le Product Backlog, ont enfin permis aux équipes métier et IT de se mettre d'accord sur une nouvelle façon de travailler. Celle-ci a permis de recentrer les efforts des équipes sur les besoins du métier et, par conséquent, sur les performances de l'entreprise. Une part d'ombre subsiste néanmoins autour du cadre de travail permettant, cette fois-ci, aux développeurs et aux opérationnels, d'être les plus efficaces possibles.

DevOps est donc l'approche qui instaure le cadre de travail permettant aux développeurs et aux opérationnels de recadrer le tir et de converger vers des équipes homogènes qui ont pour objectif de livrer des services IT avec la qualité escomptée.



QUELLES SONT LES PRINCIPALES ÉTAPES POUR ADOPTER DEVOPS ?

La collaboration au sein d'une même équipe et entre les équipes représente une condition sine-qua-non à l'adoption de DevOps au sein d'une entreprise, quelle que soit sa taille.

La nature des changements apportés par DevOps induit une implémentation au sein d'une organisation par approche itérative et ce pour plusieurs raisons. Chaque entreprise est différente de par sa culture, son métier et les spécificités de son SI. La mise en place de DevOps sera donc, elle aussi, différente. Quand on parle de SI, on va s'intéresser plus spécifiquement à une composante essentielle durant toutes les étapes d'adoption de DevOps : le capital humain.

En effet, DevOps s'articule principalement autour de deux axes : le premier concerne la collaboration et le second est lié à la capacité d'automatiser des processus.

La collaboration est un axe essentiel et nous ne le mentionnerons jamais assez. Sans collaboration, il ne peut y avoir de DevOps. Cela étant, la collaboration entre les membres des équipes ne pourra se faire sans l'investissement et l'implication de l'ensemble de la hiérarchie.

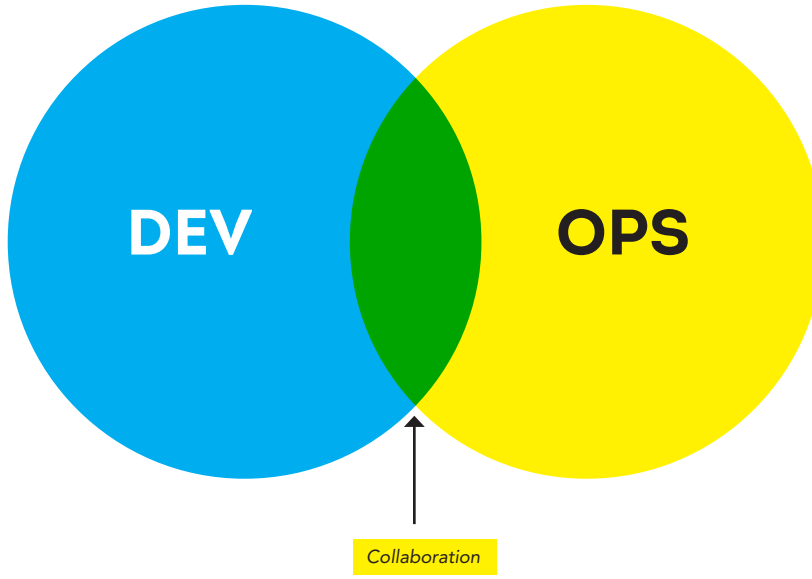
DevOps s'articule autour de deux axes : La collaboration et l'automatisation.

Le deuxième axe essentiel est celui de l'automatisation. En effet, DevOps prend toute son ampleur et son efficacité dans un environnement automatisé car cela permet d'accélérer des traitements et donc de gagner du temps. Pouvoir produire, tester, intégrer et livrer plus rapidement des applications ou des services permettra au métier de s'adapter plus vite et in-fine d'être plus performant.

Microsoft propose une large palette de produits qui s'inscrivent dans une démarche DevOps et qui permettent aux entreprises de tirer un maximum de profit dans la mise en place de cette démarche. Savoir si oui ou non ces outils sont les meilleurs du marché, est un débat que nous n'aurons pas ici. En revanche, il est certain que tous les outils fournis par Microsoft sont pleinement intégrés les uns aux autres, fournissant ainsi une des seules, si ce n'est la seule, suite complète permettant une automatisation et un total suivi d'un produit, de l'idée à son opération.

COLLABORATION

Le temps où la technologie et le business faisaient chacun chemin à part est révolu. Aujourd'hui, nous connaissons une véritable fusion entre le business et l'IT. On parle de business technology. Parmi les éléments qui permettent à une entreprise de regrouper le business et l'IT et de fournir le cadre de travail, il y a ce fameux Framework qui permet aux équipes de collaborer efficacement. Cette collaboration doit orienter les équipes vers un objectif commun.



DevOps s'abstrait de toute technologie. Mettre en place les bons outils ne suffit pas. Il faut adosser ceux-ci à une réorganisation des équipes dont le but est de fluidifier les échanges et de promouvoir la collaboration. Cette collaboration dont on parle tant, ne peut émaner intuitivement de la part des collaborateurs car, par nature, tout changement implique une certaine réticence. La vraie question est donc de trouver la meilleure approche pour accompagner ce type de changement avec un minimum d'incidence sur l'équilibre des équipes.

La première clef est d'impliquer les équipes durant les premières phases de la transformation DevOps. Pourquoi ? Prenons le cas de deux équipes de développement. La première, dont le projet a été identifié comme pilote DevOps, a été informée par le management de la transformation DevOps. De fait, l'équipe est intervenue sur les premières réflexions, à la fois sur le mode opératoire, les nouvelles compétences et sur les nouvelles technologies. La deuxième équipe a été informée ultérieurement et se doit d'appliquer les directives DevOps imposées par le management. Il est évident que la transition se passera plus naturellement au sein de la première équipe. Il est utile de rappeler ici qu'un des principes essentiels des méthodologies Agiles est de laisser l'équipe projet trouver sa propre organisation. Cela est d'autant plus valable dans le cadre de la mise en place de DevOps puisque, à l'instar de ce qui s'est produit avec SCRUM, il s'agit de fusionner, au sein de la même équipe, deux populations culturellement différentes.

Nous pouvons donc conclure que l'approche itérative qui permet d'identifier des projets et des équipes candidates à la transition DevOps, pour ensuite les embarquer dans cette transformation DevOps, représente une démarche plus efficace et moins risquée.

Intéressons-nous maintenant à un autre aspect très important quand il s'agit de collaboration, celui des compétences. Lorsque l'on s'intéresse de plus près à la transformation DevOps, l'impact de cette transition est beaucoup plus visible côté OPS que côté développement. Pourquoi ? La réponse est simple : L'automatisation des processus IT. Cela signifie le passage d'un traitement manuel (ex : déploiement d'application, construction d'une plateforme, etc.) vers un mode automatisé. Les Ops se posent donc la question suivante : si on automatise les tâches dont je suis responsable, à quoi servirai-je ? Cette question, bien que légitime puisqu'elle témoigne d'une certaine inquiétude, est loin d'être justifiée. Il n'y a pas de risque pour les Ops, bien au contraire et ce pour plusieurs raisons. D'abord, l'automatisation des processus va, certes, écarter les tâches manuelles, mais ces nouveaux automates ne sont pas suffisamment intelligents pour s'autogérer. Il faut donc mettre en place un système de monitoring « humain » qui s'assurera du bon fonctionnement, mais aussi des évolutions de ces automates. Cet aspect impliquera, aussi bien côté Ops que Dev, une montée en compétences considérable.

Ce constat reste tout aussi vrai côté Dev, dans la mesure où les développeurs, par une collaboration plus étroite avec les équipes opérationnelles, acquerront de nouvelles compétences sur les aspects liés aux infrastructures et aux opérations et pourront en tenir compte lors de leurs prochaines réalisations.

En résumé, DevOps est là pour réduire les écarts entre les développeurs et les opérationnels de l'IT, en mettant en place un cadre de travail, permettant aux équipes de collaborer et d'atteindre les mêmes objectifs. Il appartient au management d'apporter les changements organisationnels et structurels afin d'instaurer ce nouveau cadre de travail.

VOICI QUELQUES RÈGLES DE BASE, SERVANT À FACILITER LA TRANSITION DES ÉQUIPES VERS UNE APPROCHE DEVOPS :

- Les développeurs et les opérationnels ont des responsabilités et des objectifs communs.
- Les développeurs et les opérationnels n'appartiennent plus à des équipes séparées, ils font partie de la même équipe.
- L'adoption de bons outils de collaboration, d'automatisation et de contrôle.

TIME-TO-MARKET

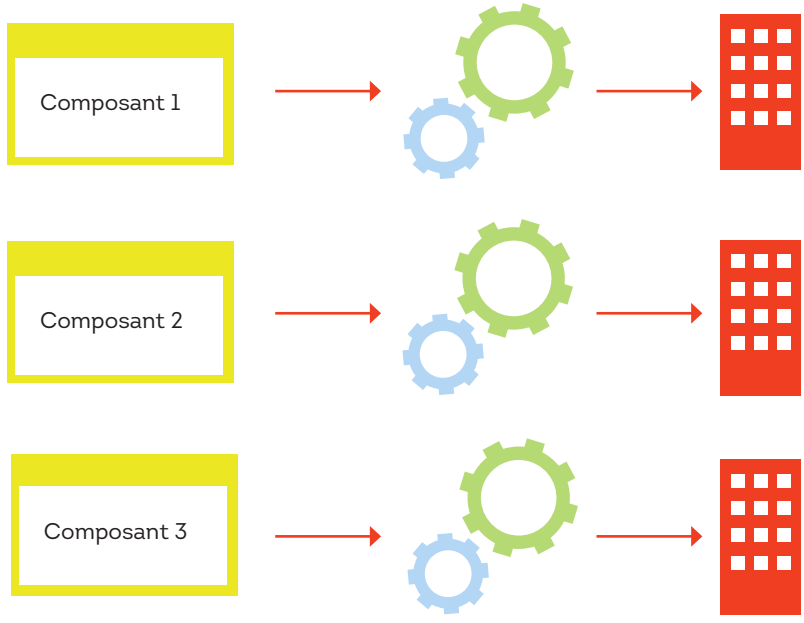
Toute entreprise, quel que soit sa taille et son secteur d'activité, s'appuie sur des outils et des services informatiques. La performance et la capacité d'une entreprise à répondre à des besoins de plus en plus changeants, de plus en plus complexes, dépend directement ou indirectement de sa capacité à produire des services IT de qualité, dans des délais de plus en plus courts.

Les organisations ayant adopté la démarche DevOps ont constaté une réduction de 20 % en moyenne sur le Time-to-market. Il s'agit d'une mesure tangible et concrète sur ce que peut apporter DevOps à la performance d'une entreprise.

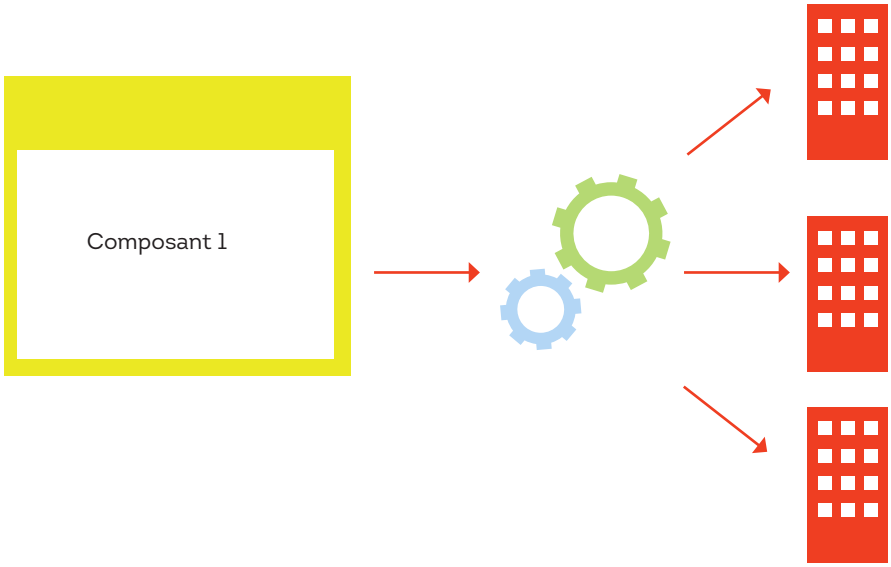
DÉPLOIEMENT CONTINU

Intéressons-nous maintenant aux applications. Le paradigme SOA, ayant pris une place de plus en plus importante dans les architectures applicatives, a permis de transformer les applications de briques monolithiques en des composants distribués. Une application typique sera composée, par exemple, d'une couche d'interface utilisateur, qui s'appuie sur une couche de services métiers, de données ou de traitement, et une couche de stockage et de traitement des données.

Partant de ce principe, la mise en place, dans son ensemble, d'un flux de déploiement continu d'une application, repose sur deux stratégies. La première consiste à considérer chaque composant d'une application comme une unité de déploiement à part entière. Le déploiement de cette unité est donc modélisé, exécuté et contrôlé de manière complétement indépendante.



La seconde approche consiste à mettre en place un flux de déploiement continu pour l'ensemble des composants de l'application, à savoir la modélisation, l'exécution et le contrôle.



Chacune des deux solutions offre des avantages et présente des inconvénients. La première solution offre la possibilité de découpler le déploiement des composants, de façon à ce qu'on puisse faire évoluer un composant sans nécessairement le faire pour toute l'application. L'inconvénient réside dans le fait que chaque flux de déploiement doit être maintenu et contrôlé unitairement.

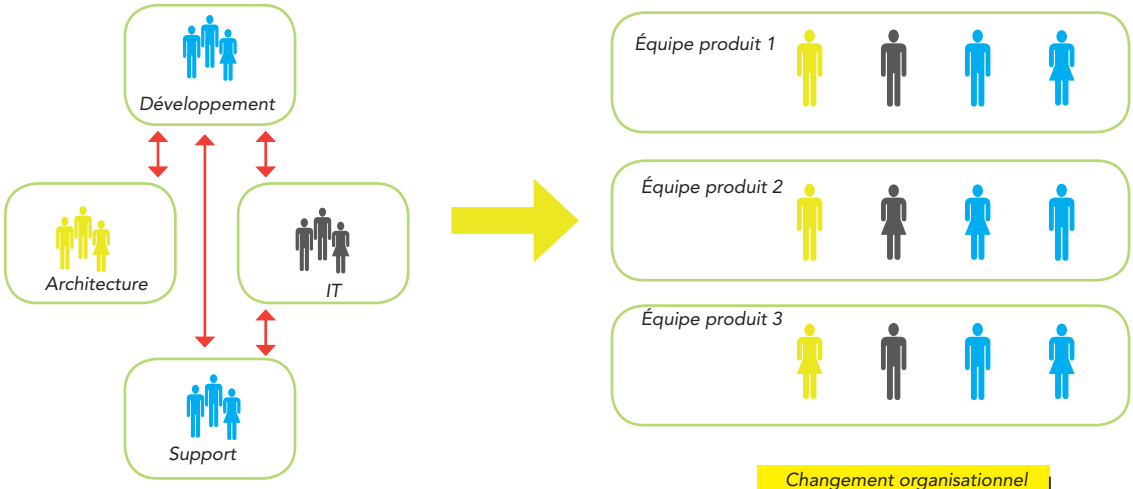
La seconde approche permet de réduire la complexité des déploiements. En revanche, un composant ne peut pas avoir un cycle de vie indépendamment des autres composants.

Il n'existe pas de solution qui réponde à toutes les problématiques. Il est donc nécessaire de se pencher, en amont, sur les spécificités de chaque application et de choisir la solution de déploiement continu la plus adaptée.

Il n'existe pas de solution qui réponde à toutes les problématiques.

CHANGEMENTS ORGANISATIONNELS

La collaboration entre les Devs et les Ops, qui représente un pilier dans la transformation DevOps, ne peut voir le jour sans l'investissement et l'implication du management de l'entreprise. Ce dernier a donc un rôle et une responsabilité prépondérants dans les changements organisationnels et structurels nécessaires à l'adoption DevOps.



Changement organisationnel des équipes Informatiques

Comme le montre la figure ci-dessus, la principale transformation organisationnelle consiste au décloisonnement des équipes et au passage d'un modèle mono-disciplinaire vers des équipes pluridisciplinaires, appelées équipes fonctionnelles ou Feature teams.
Ces équipes regroupent donc toutes les compétences nécessaires à la production d'un produit, d'une composante d'un produit ou de manière générale d'un service IT. Les échanges entre les équipes sont plus fluides, les compétences se répandent plus facilement et les équipes sont focalisées sur les mêmes objectifs.

Il est toutefois important de comprendre que, dans cette nouvelle organisation axée autour de DevOps, il n'est pas nécessaire de définir dans une équipe, dès le départ, un nouveau rôle DevOps. Cette attribution se fera de manière naturelle et émanera de l'initiative même de certains membres de l'équipe. Il faut donc laisser à l'équipe le temps de trouver son propre équilibre et retrouver le cadre de travail qui lui permettra d'être efficace et productive.

MÉTRIQUES

Il est important de mettre en place des indicateurs de contrôle.

Dans toute transition, il est important de mettre en place des indicateurs de contrôle, dont le principal objectif est de mesurer les résultats, améliorer ce qui doit être amélioré, et surtout éviter les dérives. Ces indicateurs peuvent être liés soit aux processus, auquel cas on peut mesurer un taux d'automatisation d'un processus donné, soit au mode manuel. En effet, les temps de traitement sont aussi importants et permettent de mesurer les résultats de l'automatisation.

Concernant le déploiement continu, il est intéressant de mesurer le nombre de déploiements effectués par une équipe de données et le taux de succès. Cela donnera une visibilité au business sur la capacité d'une équipe à répondre à une demande métier.

Une autre catégorie de métriques concerne cette fois l'infrastructure. Des métriques peuvent être mises en place pour mesurer le taux de mise à disposition de plateformes, cela peut inclure le réseau, le stockage et le traitement.

OUTILS

Les outils sont là pour donner la possibilité aux équipes DevOps de mettre en place les systèmes permettant d'automatiser les processus.

Microsoft propose aujourd'hui un panel de produits et de technologies permettant de répondre à l'ensemble des problématiques DevOps, aussi bien d'un point de vue développement qu'opérations.

Voici une présentation succincte des outils Microsoft permettant de supporter DevOps.

Team Foundation Server

Avec sa version 2013, Team Foundation Server est aujourd'hui la plateforme collaborative ALM par excellence. TFS regroupe un ensemble de services allant du contrôle de code source à l'exécution de tests, en passant par l'automatisation des builds.

TFS supporte également GIT comme contrôleur de code source et s'intègre nativement avec l'ensemble des produits Microsoft orientés DevOps, aussi bien les outils orientés développement que les outils dédiés aux opérationnels comme System Center Virtual Machine Manager.

http://msdn.microsoft.com/fr-fr/vstudio/ff637362.aspx

Visual Studio Online

On pourrait penser que ce n'est que la version PAAS de Team Foundation Server et pourtant c'est plus que cela. Non seulement Visual Studio Online fournit les mêmes services que Team Foundation Server, mais il possède aussi tous les avantages du cloud et en particulier de Windows Azure:

- Des mises à jours automatiques toutes les 3 semaines
- Le support des tests de charge dans le cloud
- Le déploiement natif dans Azure

Même si Visual Studio Online est dans le cloud, vous pouvez quand même connecter tous les outils de développement de la suite Visual Studio. Il est même possible de placer des serveurs de build dans vos propres datacenter.

Release Management

Cet outil permet de modéliser, d'implémenter et de contrôler des workflows de déploiement. Il s'intègre nativement avec Team Foundation Server et SQL Server. Release Management s'inscrit parfaitement dans une vision DevOps.

http://www.visualstudio.com/fr-fr/products/release-management-for-microsoft-visual-studio-vs.aspx

Desired State Configuration

Il s'agit probablement d'une petite révolution dans le monde Windows. Desired State Configuration (DSC) est un outil qui s'appuie sur PowerShell 4 et qui permet de décrire et d'appliquer une configuration spécifique.

La nouvelle syntaxe introduite par DSC est simple et intuitive. L'outil permet, par ailleurs, de gérer une infrastructure au même titre qu'un code applicatif.

http://blogs.technet.com/b/privatecloud/archive/2013/08/30/introducing-powershell-desired-state-configuration-dsc.aspx

Application Insight

Application Insight est, comme les tests de charge, un composant de Visual Studio Online. Application Insight utilise l'atout majeur du cloud pour fournir des métriques difficiles à obtenir autrement sans une infrastructure coûteuse, par exemple, le temps de réponse de vos sites webs à partir de plusieurs points dans le monde. Application Insight supporte de multiples technologies (.Net, java, javascript…) et plusieurs types d'applications (mobile, tablettes, site web). Les informations sont stockées dans le cloud et directement visibles depuis Visual Studio Online. Le système d'alerte vous permet d'être averti en cas de problèmes sur des métriques systèmes ou sur les vôtres.
Application Insight est nativement intégré à Visual Studio : les métriques personnalisées sont définies au moment du développement. Cela suit exactement les préconisations de DevOps : impliquer les développeurs dans la remontée d'informations en production.
http://msdn.microsoft.com/fr-fr/library/dn481095.aspx

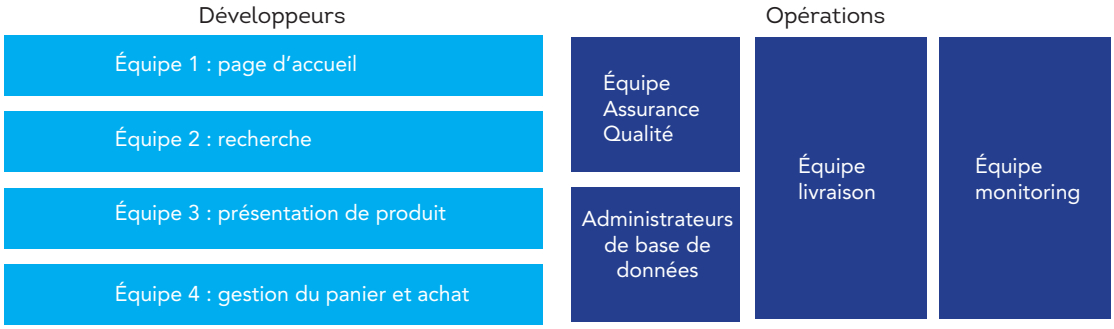
2

DEUXIÈME PARTIE

- Il était une fois
- La route vers DevOps
  - Episode I : Une nouvelle organisation
  - Episode II: Processus et outils
- Build
- Déploiement
- Release Management
- Release path
  - Desired State Configuration
  - DSC et Release Management
- Et ils vécurent heureux

IL ÉTAIT UNE FOIS

Afin d'illustrer comment amener une équipe à adopter DevOps, nous allons nous appuyer sur l'histoire (fictive) du site web de Contoso. Contoso est une entreprise de magasins en ligne, offrant une boutique en ligne pour ses produits et ceux d'autres compagnies. Métier et développement travaillent main dans la main en suivant les méthodologies agiles (SCRUM et XP). Toutefois, au début de notre exemple, l'équipe Operations travaille à l'ancienne. L'organisation reflète cette dichotomie :



Organisation de Contoso, avant DevOps

Les développeurs et le métier sont organisés en équipes fonctionnelles, chacune chargée d'une fonctionnalité du site web, chacune incluant tant des développeurs que des analystes métier et chacune responsable de la définition et de la construction de leur fonctionnalité.

Cependant, les Opérations ne suivent pas ce schéma. Chaque fois qu'une équipe fonctionnelle veut livrer, elle doit d'abord obtenir l'approbation des administrateurs de base de données (DBA), faire valider ses binaires par l'équipe assurance qualité (QA) puis les passer à l'équipe livraison (SD) qui, elle, les déploiera sur les environnements. La maintenance et le monitoring quotidiens sont effectués par l'équipe monitoring et aucun indicateur n'est remonté aux équipes fonctionnelles.

Afin de maximiser leur efficacité, QA et SD font les tests et le déploiement uniquement sur le site web complet et ont organisé leur agenda par périodes de 3 mois, assurant de fait 4 livraisons (théoriques) par an. Ainsi, pour avoir une fonctionnalité en production en juillet, elle doit être passée de QA à SD en mars et doit donc être livrée à QA en janvier. Il existe cependant un cycle court pour les cas où le Monitoring détecte un bug ou un problème de performance. Dans ce cas, un patch est livré en utilisant la procédure d'urgence qui ne prend que 2 semaines, mais qui a un certain coût : cela retarde le mécanisme trimestriel de livraison.

En réalité, le site web est livré en moyenne trois fois par an au lieu de quatre. Le processus donne ceci:

Cycle de vie d'une fonctionnalité, avant DevOps

L'objectif principal de Contoso est de livrer les fonctionnalités plus rapidement. Bien évidemment, le niveau de qualité doit augmenter en parallèle, car livrer des bugs plus souvent n'est pas vraiment une option viable. Deux éléments clefs sont nécessaires afin de parvenir à cet objectif.

Le premier est un changement organisationnel. Quelque temps plus tôt, Contoso a changé son organisation afin de fusionner métier et développement en équipes fonctionnelles. Ce changement doit maintenant être étendu jusqu'à la fin de la chaîne, incluant les opérations dans les équipes fonctionnelles.

Le second est la mise en place d'une usine de déploiement continu. Pour supporter le changement d'organisation, des processus et des outils sont nécessaires, visant à automatiser le plus possible la chaîne de livraison. Ainsi, les erreurs seront moins fréquentes et le processus de livraison actuel et monolithique pourra être morcelé afin de permettre une livraison indépendante de chaque fonctionnalité.

Épisode I : Une nouvelle organisation

Nous avons beaucoup parlé de changement organisationnel lors de la mise en place de DevOps. Il existe plusieurs clefs permettant de mettre en place ce changement de manière fluide dans une organisation. Tout d'abord, notons que si, à l'instar de Contoso, une entreprise a déjà fait sa révolution Agile à travers SCRUM, il ne s'agit pas tant de mettre en place une nouvelle organisation que d'étendre l'organisation existante, en équipes fonctionnelles, jusqu'à l'extrémité de la chaîne de valeur, constituée par les Ops.

Ainsi, l'équipe fonctionnelle, comprenant déjà des rôles liés au métier (Product Owner, business analyst) et au développement (Développeur, Tech-lead) va s'enrichir de rôles dédiés aux opérations (Qualité, déploiement, monitoring).

Quelques modifications sont cependant à prévoir dans les tâches dévolues à ces rôles.

Concernant la qualité, l'objectif est de modifier le processus de test de manière à ce qu'il soit non seulement automatisé, mais également proactif. Cela signifie que les tests d'intégration doivent être écrits avant le développement et automatisés, autant que faire se peut, par exemple à l'aide de SpecFlow. Si cette étape est menée à son terme, les fonctionnalités sont créées en mode ATDD (Acceptance Test Driven Development), c'est-à-dire que le point d'entrée pour les développeurs n'est plus une spécification détaillée mais un ensemble de tests d'acceptation à valider.

Concernant le déploiement, là encore un changement est à prévoir. L'objectif n'est plus de réaliser le déploiement mais de le modéliser dans l'outil approprié (TFS Release Management), en liaison avec les développeurs. On passe alors d'un rôle d'exécutant à un rôle de concepteur.

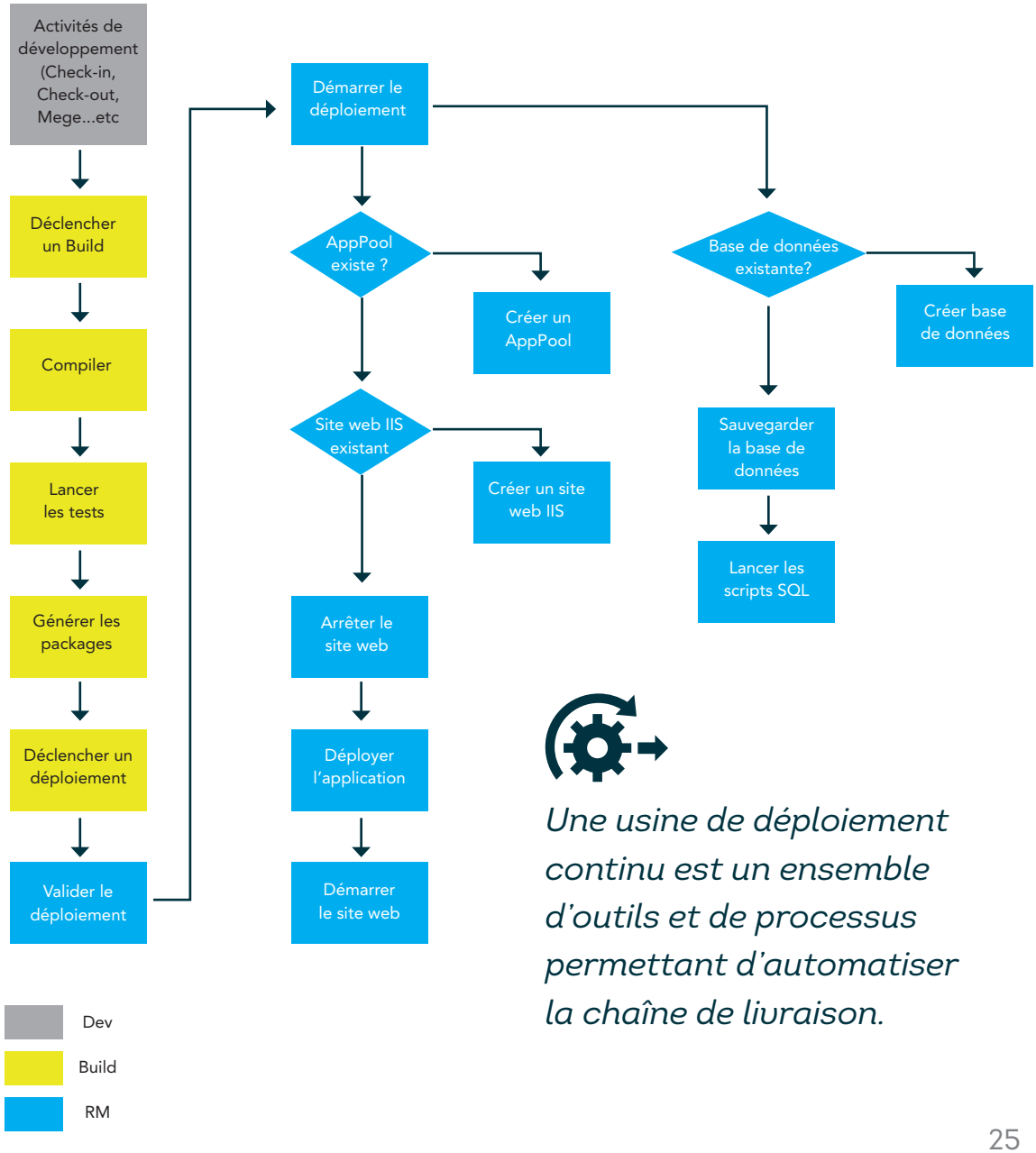
Coté monitoring, le métier conserve son aspect antérieur, mais s'y ajoute également la définition des métriques liées à la fonctionnalité, en liaison avec le rôle qualité, de manière à ce que la mise en place de ces métriques soit validée par des tests d'acceptation automatisés. Par ailleurs, un flux de retours d'information permettant à l'ensemble de l'équipe de suivre ces métriques et de réagir en cas d'anomalie doit être mis en place.

Épisode II : Processus et outils

Dans une organisation DevOps, le processus de construction d'une fonctionnalité ne s'arrête plus une fois le développement terminé. L'ensemble de la chaîne en fait partie, jusqu'au déploiement en production. Un processus typique est décrit ici et suivi pas à pas afin d'illustrer sa mise en œuvre à l'aide des outils Microsoft.

Le processus illustre les différentes phases du déploiement de l'application. Nous mettrons ensuite le focus sur chacune des parties avec un zoom sur chacune des technologies sous-jacentes.

Le processus de déploiement global se décline de la manière suivante :



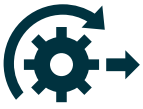
Une usine de déploiement continu est un ensemble d'outils et de processus permettant d'automatiser la chaîne de livraison.

Trois principales catégories sont présentes, liée chacune à un outil différent. La première concerne les activités de développement et de conception, par exemple la gestion des work items, l'archivage de code, la récupération d'une version de code ou la fusion entre des branches de code. Ce sont des activités courantes de développement, qui sont gérées en grande partie par le briques Gestion de code source de TFS. Ces activités sont déjà largement couvertes par d'autres documents, nous n'allons donc pas nous y attarder plus longuement.

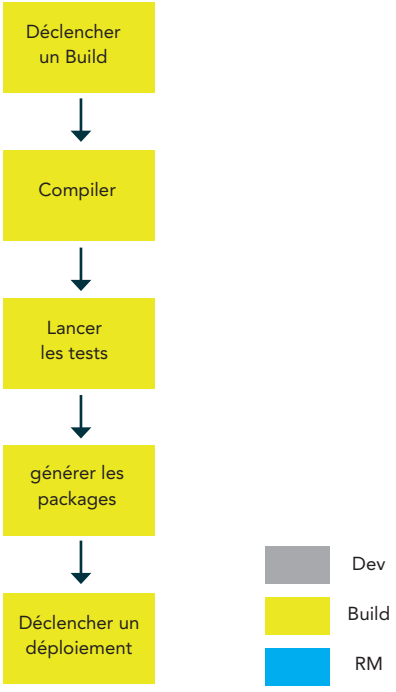
La deuxième activité concerne la construction des packages. La brique Build de TFS permet d'assurer l'ensemble de ces activités, et permet, entre autres, de construire les packages applicatifs.

La troisième activité concerne le déploiement de l'application. Celle-ci comporte également la préparation des plateformes (ex : configuration des machines, installation des rôles et fonctionnalités Windows...). Elle s'appuie sur TFS Release Management et Powershell DSC (Desired Sate Configuration).

A noter que l'objectif n'est pas de couvrir l'ensemble des fonctionnalités de chacune de ces technologies, nous nous intéresserons spécifiquement à celles utilisées dans l'exemple du processus de déploiement continu.

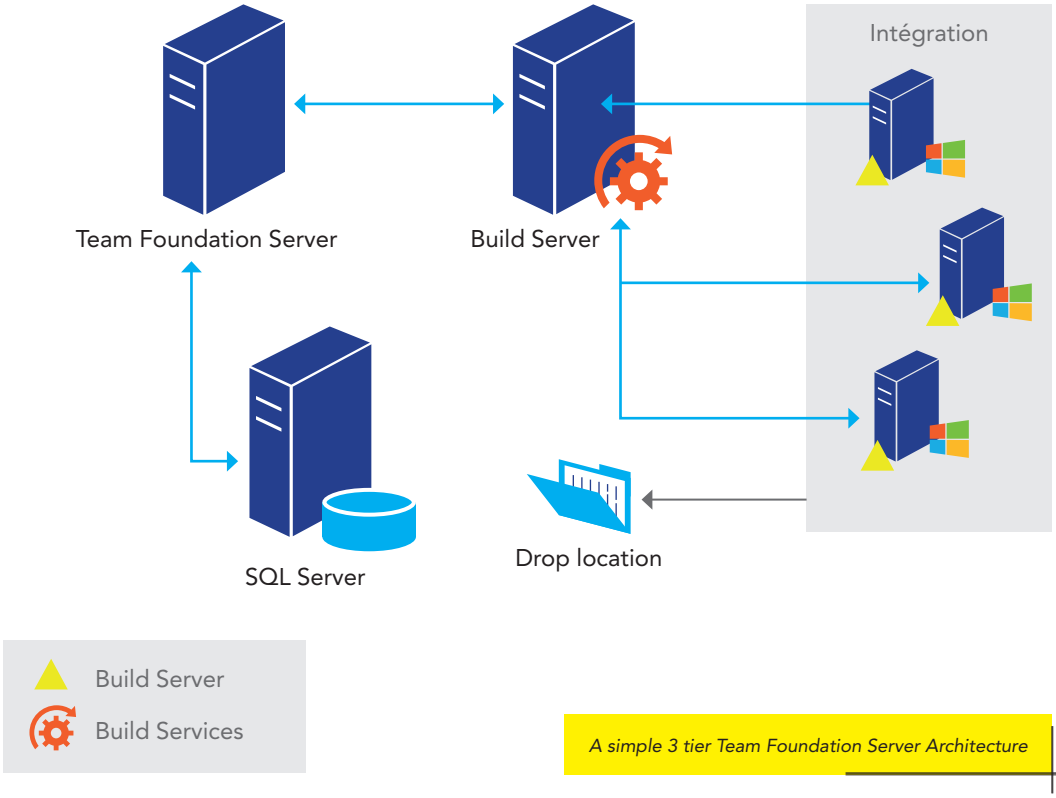


Le processus de construction d'une fonctionnalité ne s'arrête pas une fois le développement terminé.



Team Build 2013 est la brique TFS qui permet de gérer un ou plusieurs Builds d'une ou plusieurs applications. Team Build 2013 adopte une architecture client-serveur.

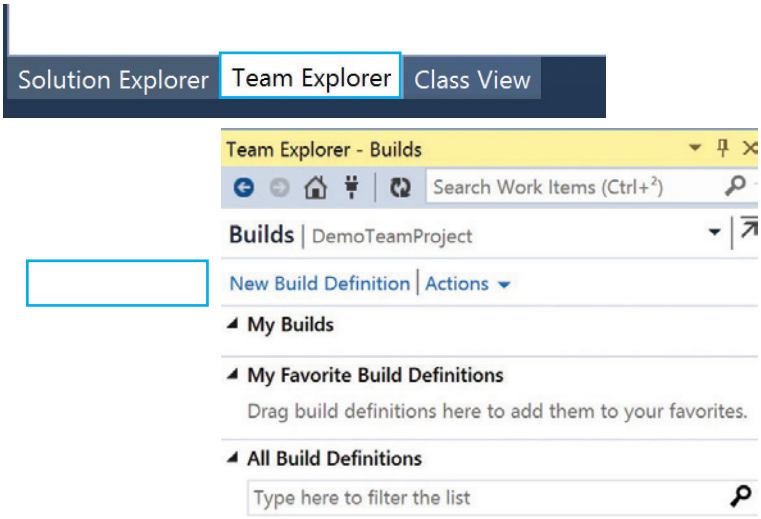
Ci-dessous un schéma simplifié d'une architecture Team Build 2013 :



A simple 3 tier Team Foundation Server Architecture

On y retrouve le serveur de Build, appelé le Build controller, le serveur SQL, le frontal TFS et les machines d'agents de Build, ainsi que la Drop Location dans laquelle seront déversés l'ensemble des artefacts issus du Build.

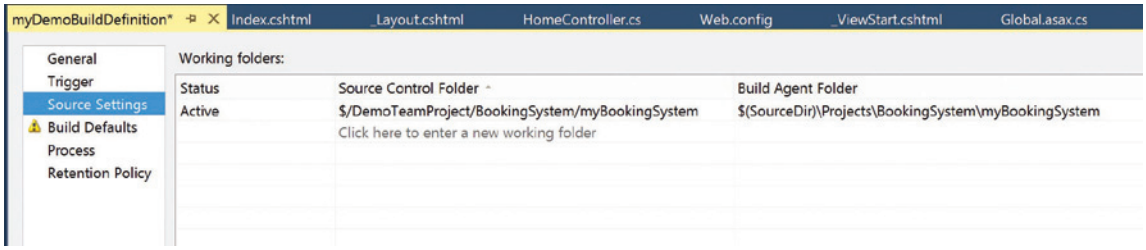
La première étape, en référence à notre processus, consiste à créer une Définition de Build. Cette définition va permettre de décrire les étapes nécessaires à la construction d'un package applicatif.



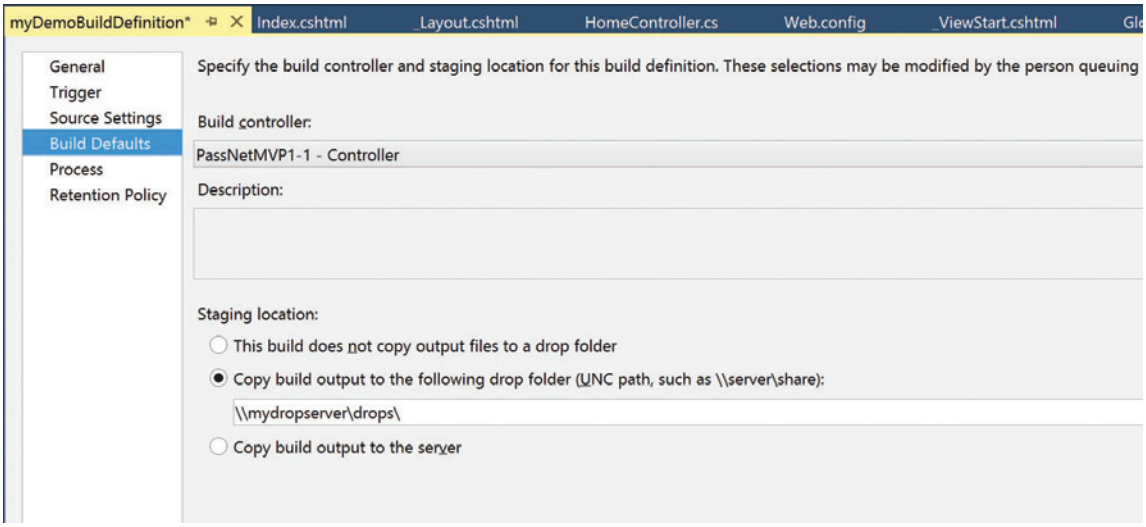
Create a newbuild definition

L'écran de configuration permet de spécifier les paramètres du Build, à savoir le déclenchement du Build, ce qui signifie que l'on peut soit le déclencher manuellement soit le déclencher automatiquement après chaque check-in. Le déclenchement automatisé (sur check-in ou nocturne) peut, en effet, être intéressant lorsque l'on s'approche, par exemple, d'une date de livraison et que l'on veut s'assurer qu'un archivage de code ne perturbe pas le processus de Build.

Les autres paramètres du Build peuvent être spécifiés en fonction du projet, comme le montrent les deux figures ci-dessous :



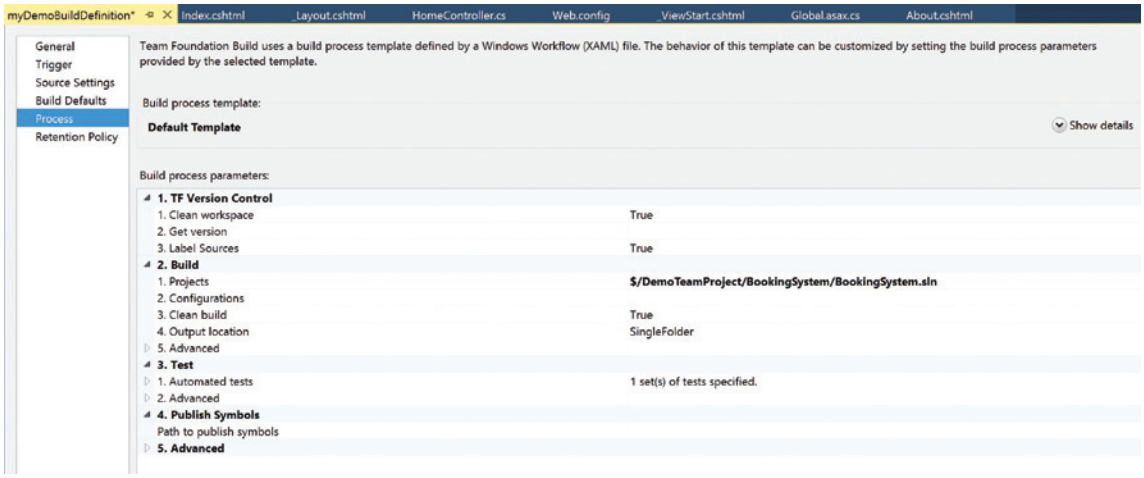
Source settings in build definition



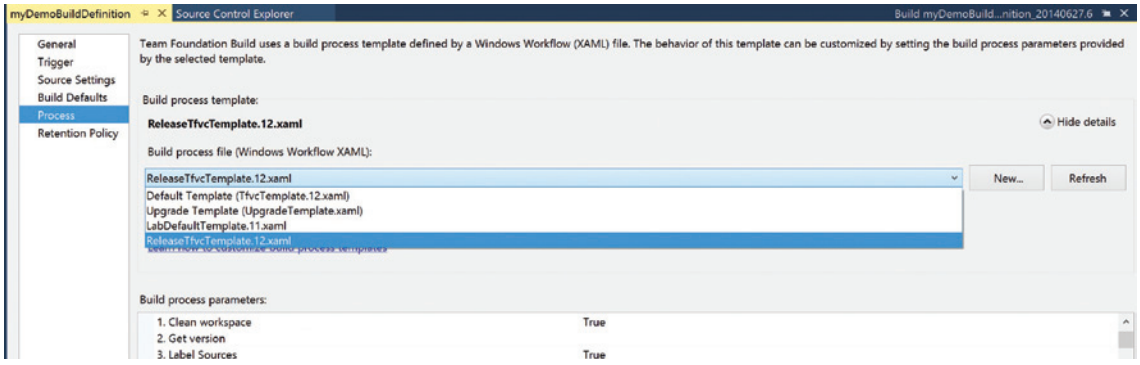
Drop location for output

Intéressons-nous maintenant au Template du processus. Livré avec Release Management, ReleaseTfvcTemplate12.xaml est le template qui permet à l'agent de build de déclencher un déploiement de manière automatique sur le serveur Release Management, sous réserve que les étapes précédentes (compilation & tests unitaires) soient réussies. L'idée est, en effet, de chaîner de manière automatique l'ensemble des éléments du processus. Nous aborderons Release Management plus en détails dans le chapitre suivant.

Les écrans ci-dessous montrent la sélection du template de processus en question :



Build process template 1



Build process template

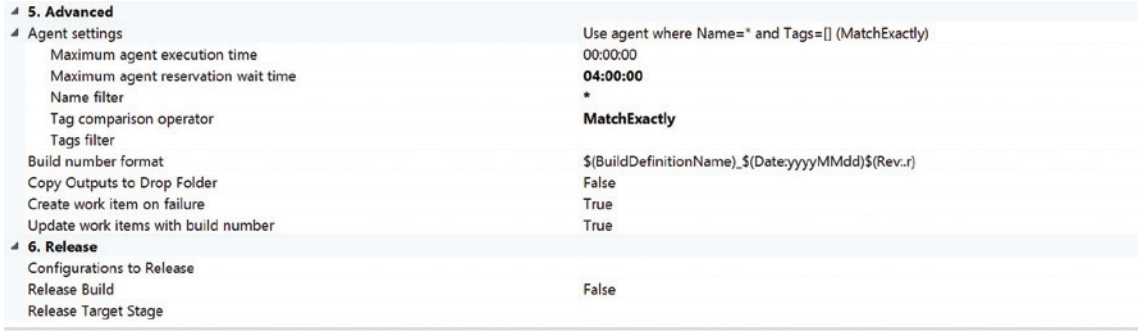
Le template de release contient les activités nécessaires qui permettent de configurer la connexion entre le Build et Release Management.

Ce template se trouve sur le serveur RM à l'endroit suivant:

C:\Program Files (x86)\Microsoft Visual Studio 12.0\Release Management\bin

Le fichier XAML du template Release Management devra être copié au même emplacement que les autres templates de Build.

Lorsque l'on utilise le template Release Management nous voyons apparaître, dans le panneau de configuration du Build, les propriétés de Release Management, comme le montre l'écran ci-dessous :



5. Advanced	
Agent settings	Use agent where Name=* and Tags=[] (MatchExactly)
Maximum agent execution time	00:00:00
Maximum agent reservation wait time	**04:00:00**
Name filter	*
Tag comparison operator	**MatchExactly**
Tags filter	
Build number format	$(BuildDefinitionName)_$(Date:yyyyMMdd)$(Rev:.r)
Copy Outputs to Drop Folder	False
Create work item on failure	True
Update work items with build number	True
6. Release	
Configurations to Release	
Release Build	False
Release Target Stage	

Advanced build properties

Dans cette partie, nous pouvons effectivement configurer les éléments suivants:

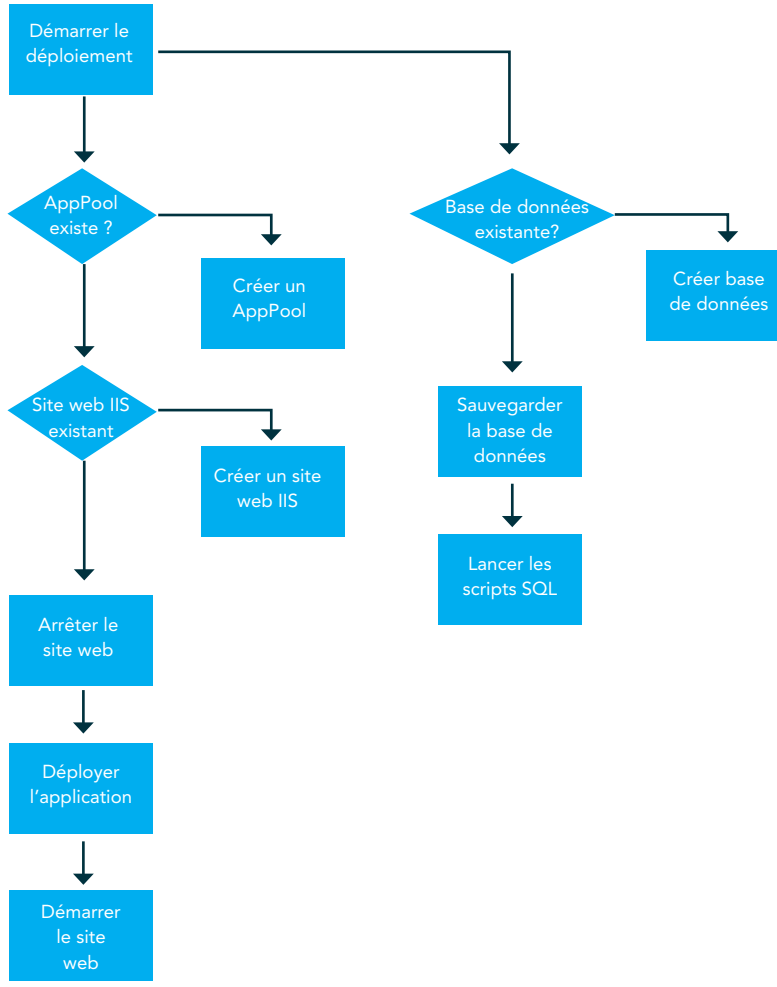
- La configuration à déployer : la configuration du projet concerné (Debug, Release...etc). A noter que cette configuration doit également se retrouver au niveau de celles compilées lors de ce build.
- Déployer un Build : Ce paramètre permet de déclencher ou non un déploiement suite à la fin d'un Build réussi.
- Le Target Stage : Il s'agit d'une étape dans le processus de déploiement. Bien que l'on aborde ces éléments dans le chapitre dédié à Release Management, une étape dans un processus peut, par exemple, être l'étape Intégration, Test ou Pré-production. Le déploiement issu de ce build ne dépassera jamais le stage indiqué ici, donc si l'objectif est de déployer in fine en production, mieux vaut indiquer ce stage. Le déploiement pourra de toutes manières être interrompu à une étape antérieure si nécessaire.

La brique Build de TFS permet, entre autres, de construire les packages applicatifs.

DÉPLOIEMENT

Contoso a fait le choix des MSI pour déployer ses fonctionnalités. A noter que cela a peu d'importance sur Release Management car les outils proposés par ce dernier supportent aussi bien les MSI que les packages compatibles web deploy ou même la copie directe vers les serveurs cibles. Il faut noter également que plus les packages ont d'opérations à réaliser sur les machines cibles, plus le processus de déploiement s'enrichit.

Dans ce chapitre, nous introduirons l'outil Release Management et DSC, dans le contexte de l'application Contoso, et nous nous nous intéresserons plus particulièrement aux étapes présentes dans le processus ci-dessous :

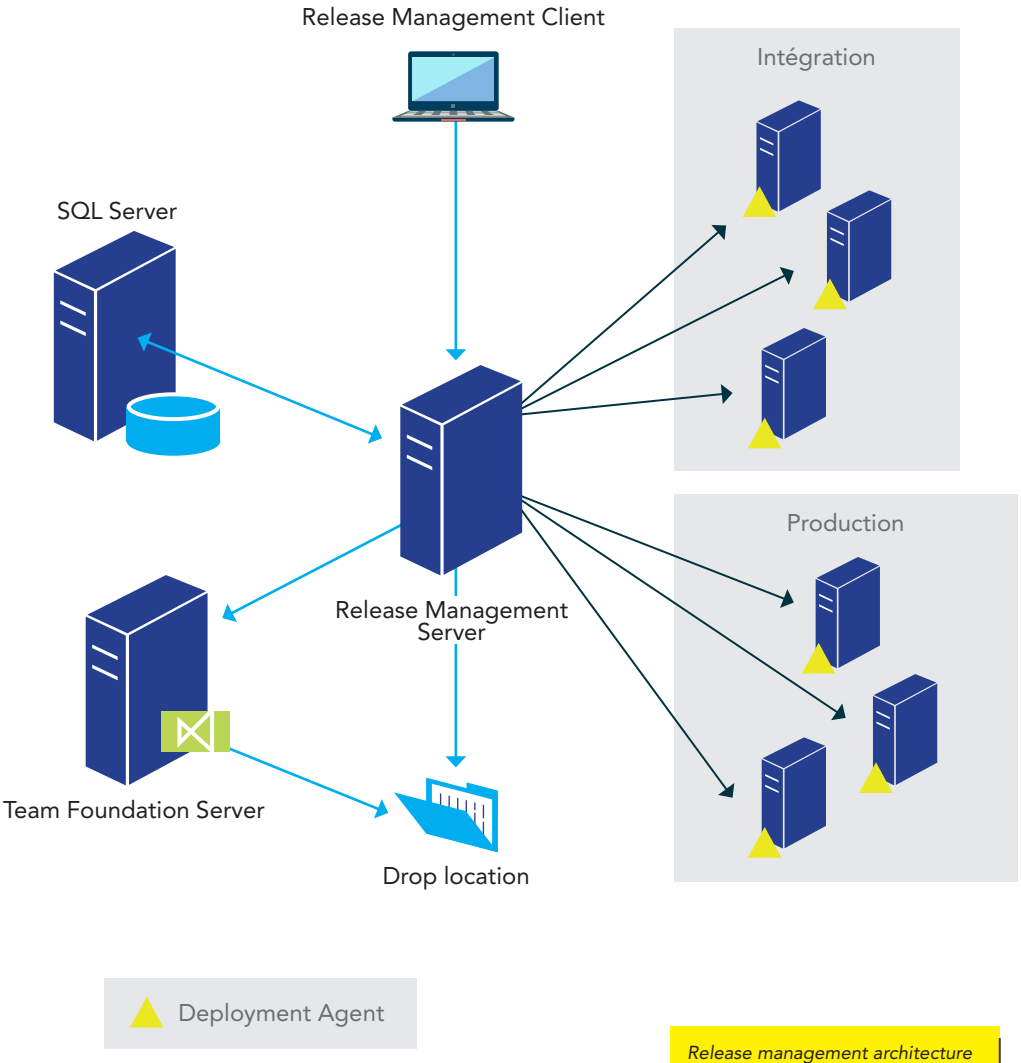


RELEASE MANAGEMENT

Release Management (ex “InRelease”) est une technologie Client-Serveur qui permet de modéliser, d’automatiser et de contrôler des processus de déploiement sur différents environnements.

L’architecture de Release Management est, en quelque sorte, similaire à celle de Team Foundation Server.

Le schéma ci-dessous présente l’architecture simplifiée de Release Management :



Release management architecture

La brique principale est le serveur Release Management. Il sert de point de pivot central à l'ensemble de l'infrastructure, permettant aux serveurs cibles d'obtenir les packages (à travers les agents) et aux utilisateurs de définir et déclencher les déploiements (à travers le client lourd).

Chacun des serveurs faisant partie d'un environnement cible doit contenir l'agent de déploiement. Cet agent va se connecter au serveur afin d'obtenir la liste des éléments à déployer, permettant ainsi une communication en mode Pull. Le mode Push n'est pas encore disponible à l'heure où nous écrivons ces lignes, mais il est prévu.

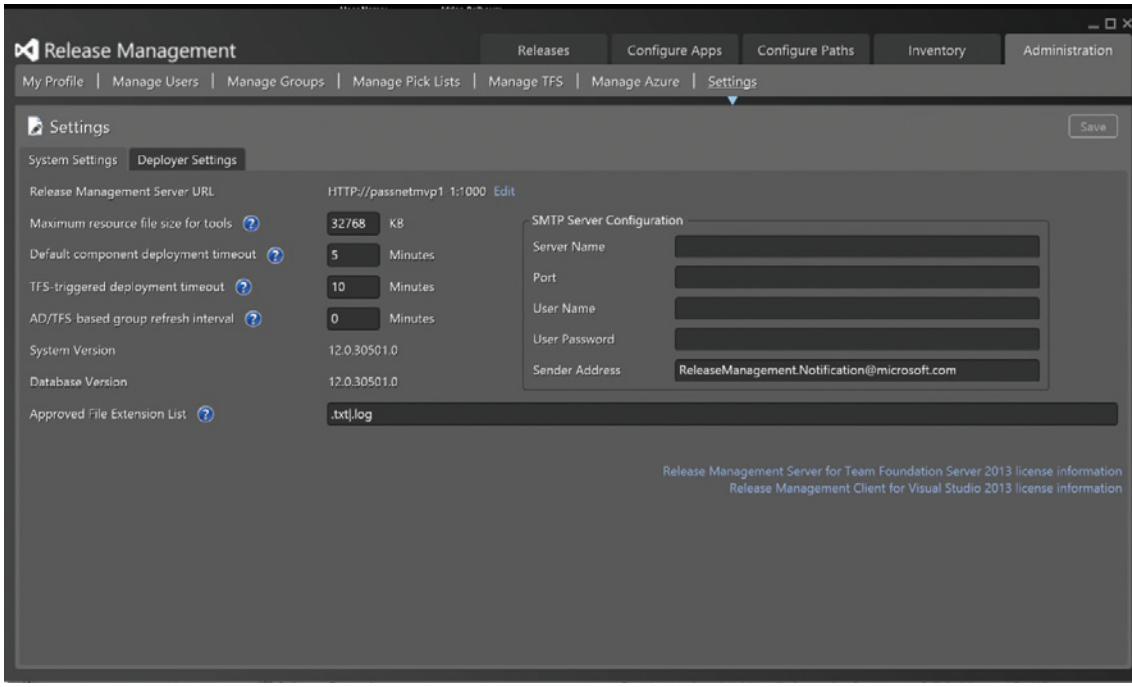
Le lien ci-dessous explique, en détails, les étapes d'installation de Release Management :

http://msdn.microsoft.com/en-us/library/dn593700.aspx

Le client RM est une application « Client Lourd » qui donne accès à une console d'administration et de suivi de tous les éléments de déploiement.

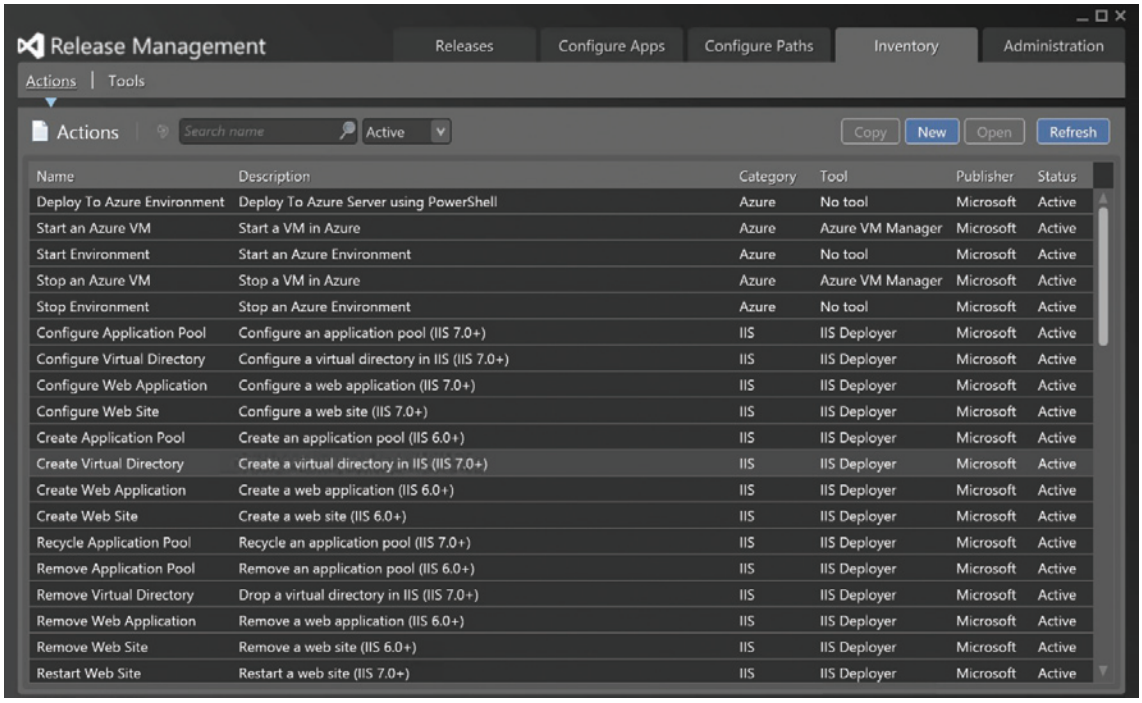
Faisons un tour d'horizon pour mieux comprendre les principales fonctionnalités proposées par Release Management.

Le premier élément est celui de la configuration de l'outil. Via l'onglet Administration, on accède aux différents paramètres de RM.



Paramètres généraux de Release Management

L'écran ci-dessous présente la liste des outils disponibles par défaut dans Release Management. Cette liste peut être étendue en créant un nouvel outil. Un outil est un exécutable ou un script qui est mis à disposition des processus de déploiements à travers des actions. Ainsi, les outils sont assez génériques (IIS deployer pour tout ce qui concerne IIS…).



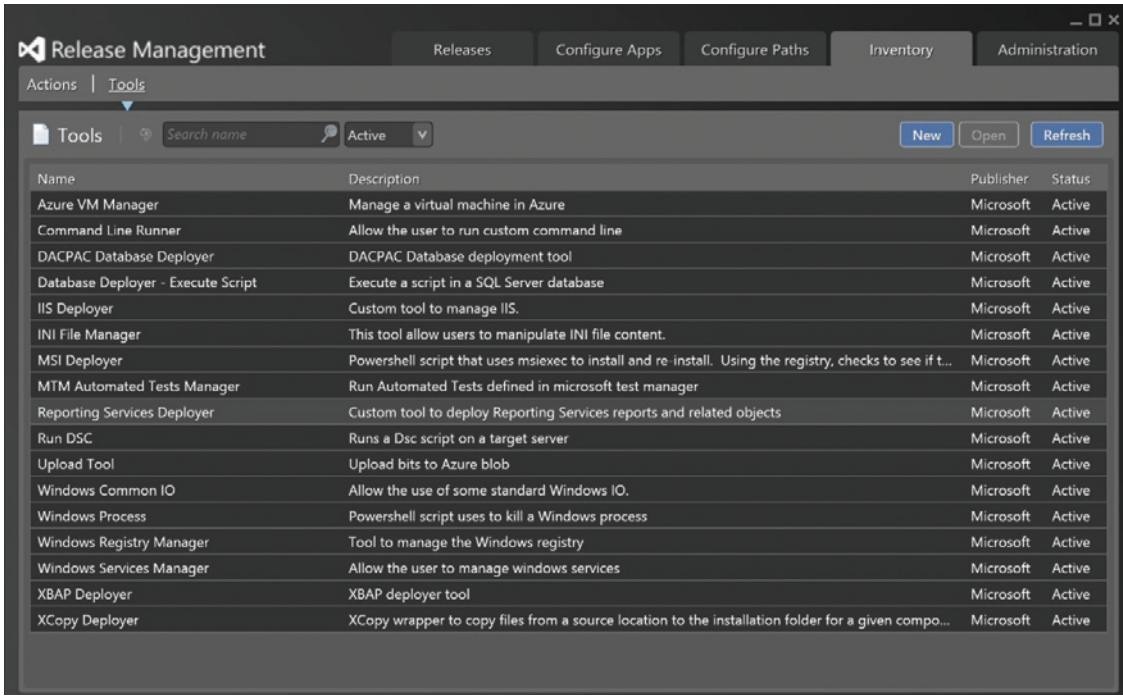
Name	Description	Category	Tool	Publisher	Status
Deploy To Azure Environment	Deploy To Azure Server using PowerShell	Azure	No tool	Microsoft	Active
Start an Azure VM	Start a VM in Azure	Azure	Azure VM Manager	Microsoft	Active
Start Environment	Start an Azure Environment	Azure	No tool	Microsoft	Active
Stop an Azure VM	Stop a VM in Azure	Azure	Azure VM Manager	Microsoft	Active
Stop Environment	Stop an Azure Environment	Azure	No tool	Microsoft	Active
Configure Application Pool	Configure an application pool (IIS 7.0+)	IIS	IIS Deployer	Microsoft	Active
Configure Virtual Directory	Configure a virtual directory in IIS (IIS 7.0+)	IIS	IIS Deployer	Microsoft	Active
Configure Web Application	Configure a web application (IIS 7.0+)	IIS	IIS Deployer	Microsoft	Active
Configure Web Site	Configure a web site (IIS 7.0+)	IIS	IIS Deployer	Microsoft	Active
Create Application Pool	Create an application pool (IIS 6.0+)	IIS	IIS Deployer	Microsoft	Active
Create Virtual Directory	Create a virtual directory in IIS (IIS 7.0+)	IIS	IIS Deployer	Microsoft	Active
Create Web Application	Create a web application (IIS 6.0+)	IIS	IIS Deployer	Microsoft	Active
Create Web Site	Create a web site (IIS 6.0+)	IIS	IIS Deployer	Microsoft	Active
Recycle Application Pool	Recycle an application pool (IIS 7.0+)	IIS	IIS Deployer	Microsoft	Active
Remove Application Pool	Remove an application pool (IIS 6.0+)	IIS	IIS Deployer	Microsoft	Active
Remove Virtual Directory	Drop a virtual directory in IIS (IIS 7.0+)	IIS	IIS Deployer	Microsoft	Active
Remove Web Application	Remove a web application (IIS 6.0+)	IIS	IIS Deployer	Microsoft	Active
Remove Web Site	Remove a web site (IIS 6.0+)	IIS	IIS Deployer	Microsoft	Active
Restart Web Site	Restart a web site (IIS 7.0+)	IIS	IIS Deployer	Microsoft	Active

Liste des outils dans Release Management

Release Management est une technologie Client-Serveur qui permet de modéliser, d'automatiser et de contrôler des processus de déploiement sur différents environnements.

L'écran ci-dessous présente, quant à lui, la liste des actions disponibles, par défaut, dans Release Management. Cette liste peut être étendue en créant une nouvelle action.

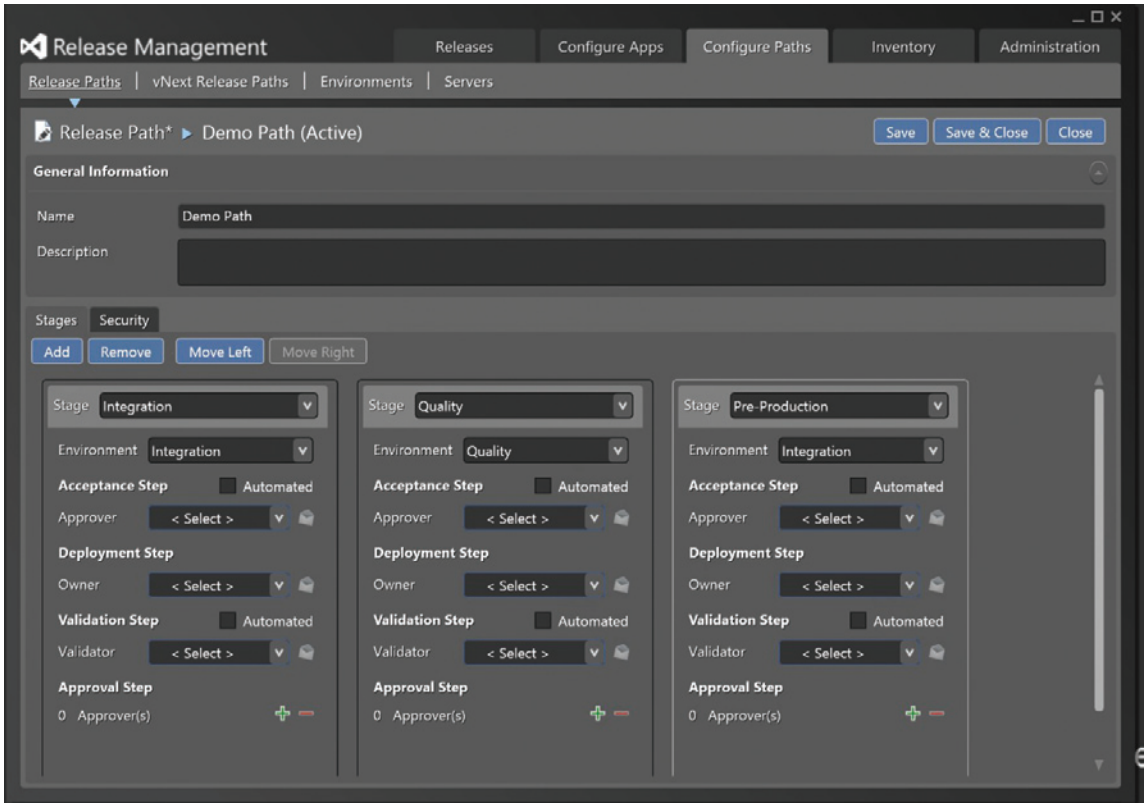
Une action est l'utilisation d'un outil pour réaliser une tâche précise et (idéalement) atomique. Ainsi, en reprenant l'outil IIS déployer, plusieurs actions sont créées, par exemple pour créer un Application Pool ou un Web Site, pour configurer les bindings ou autorisations d'un site, etc.



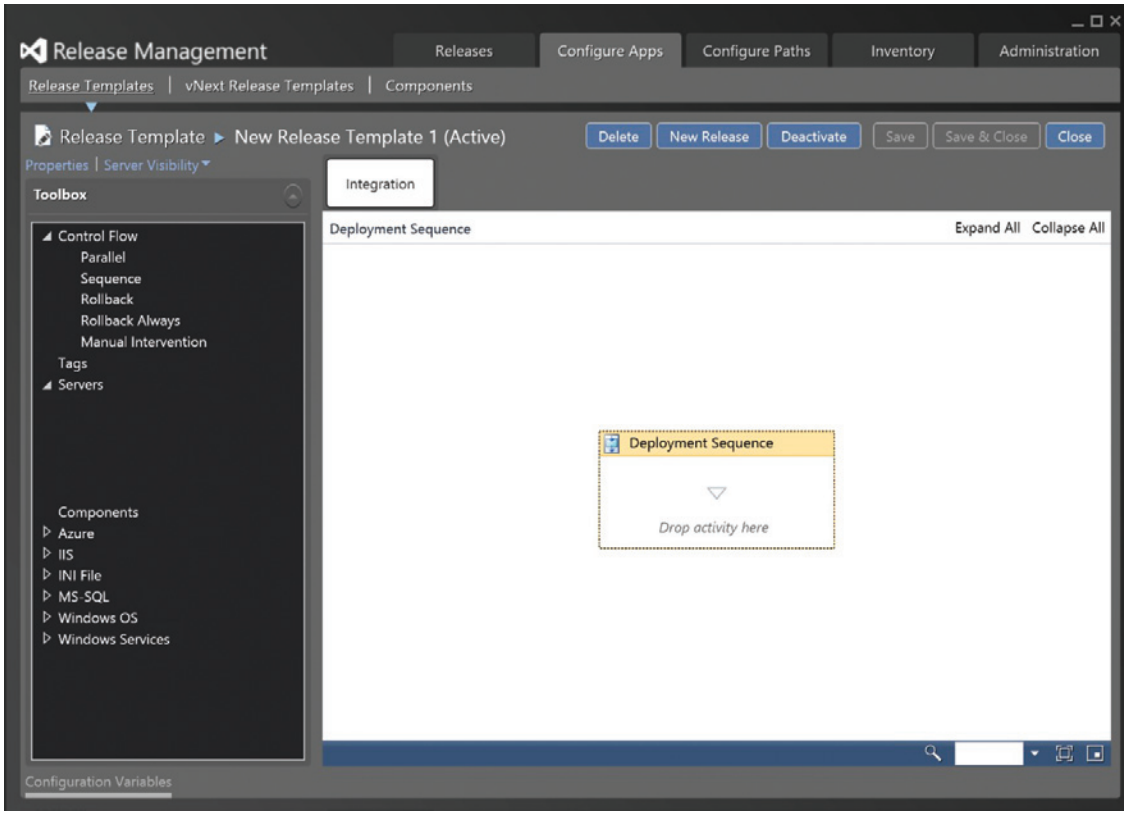
Actions par défaut dans Release Management

La configuration des processus de déploiement se fait via l'onglet Configure Path, comme le montre la figure ci-dessous.

On peut effectivement spécifier différentes étapes (stage) dans ce processus, chaque étape fait référence à un environnement. La création de ces environnements n'est pas illustrée ici. Chaque environnement regroupe plusieurs serveurs (voir un seul, pour un environnement de tests, par exemple) contenant l'ensemble des prérequis nécessaires à l'application. Si ces prérequis peuvent être installés grâce à DSC, pour certains d'entre eux (SQL Server par exemple), il peut être préférable de les préinstaller, voire de les inclure directement dans les templates de VM.



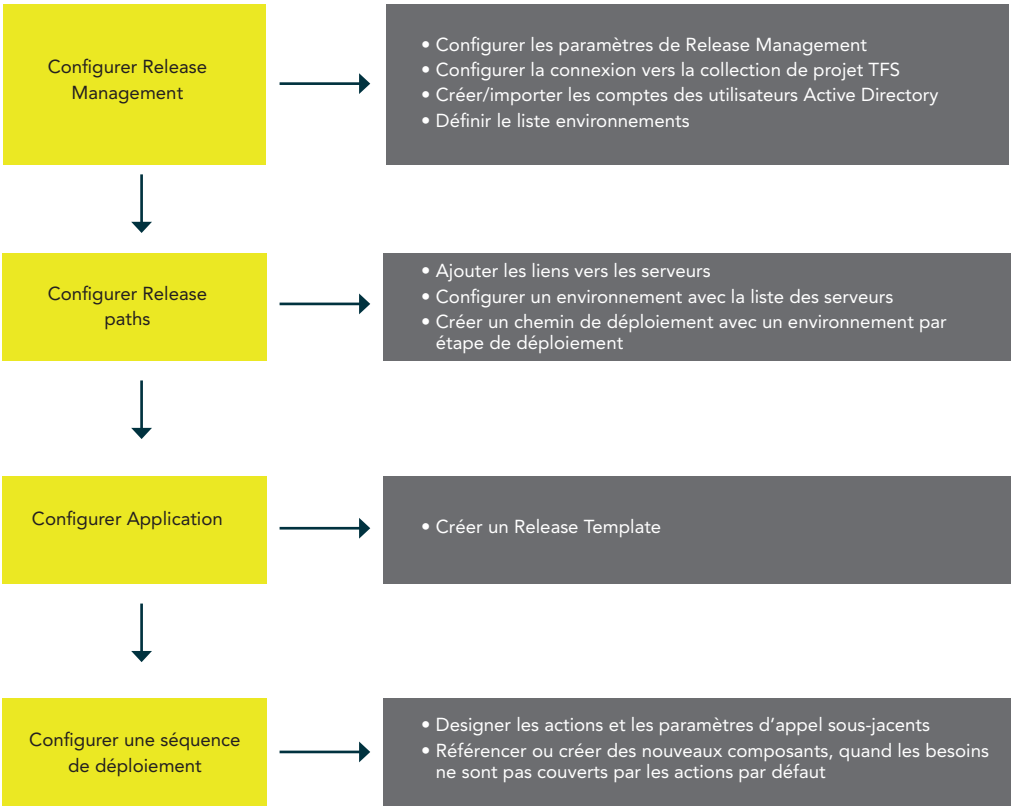
Processus de déploiement



Séquence de déploiement

La création d'une séquence de déploiement (écran ci-dessus) permet de décrire les différentes étapes dans le déploiement d'une application. L'écran principal fonctionne par glisser-déposer. Sur un serveur donné, le déploiement prend obligatoirement la forme d'une séquence linéaire d'actions à effectuer les unes après les autres, le moindre échec interrompant la chaîne. Si rien n'est prévu, l'échec peut donc entraîner un état semi-fini du déploiement. C'est pourquoi une action de rollback peut être prévue, visant à restaurer l'état antérieur en cas d'échec du déploiement.

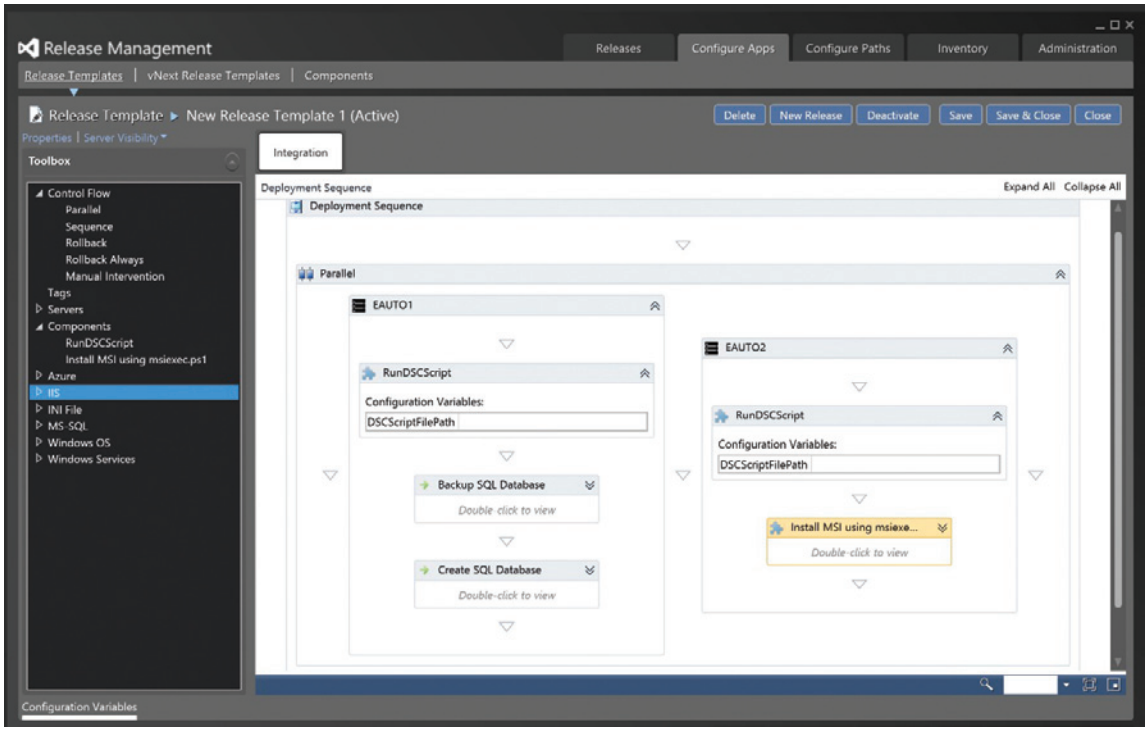
Le schéma ci-après sert de synthèse aux différentes étapes nécessaires à la mise en place d'un déploiement automatique dans Release Management



Processus simplifié de mise en place d'un déploiement

4 configurations importantes pour le déploiement automatique dans Release Management: Release Management, Release Paths, Application et une séquence de déploiement.

Nous allons, à présent, nous focaliser sur la séquence de déploiement de fonctionnalités. Comme nous le verrons dans le chapitre suivant, la séquence de déploiement est considérablement simplifiée, grâce à l'intégration de scripts DSC. Ces scripts permettent, en effet, de regrouper un ensemble d'opérations de vérification et de configuration nécessaires au déploiement de la fonctionnalité.

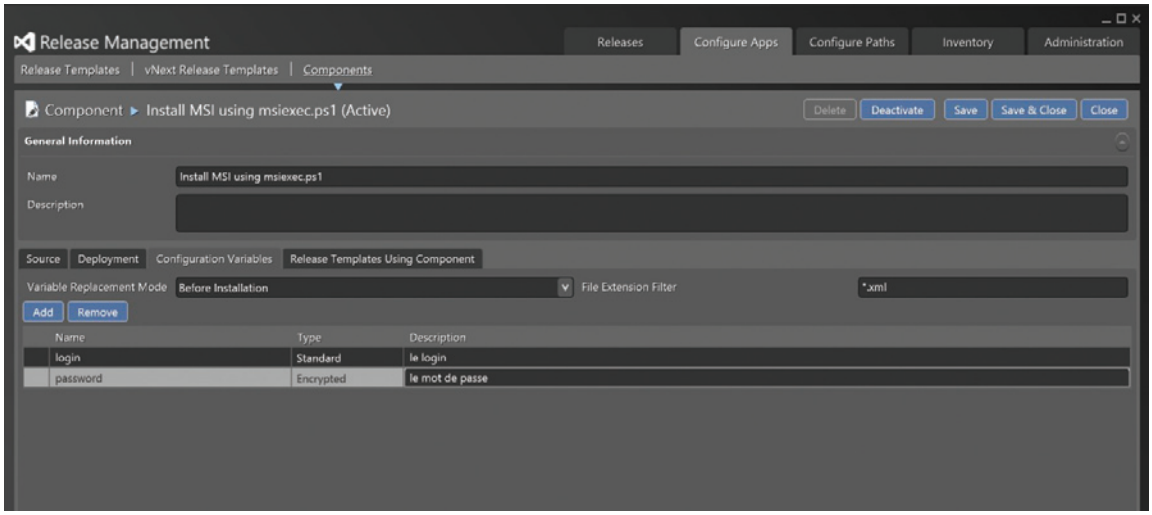


Séquence de déploiement d'une fonctionnalité

Nous constatons, dans le schéma, ci-dessus plusieurs éléments. Le premier concerne l'exécution parallèle entre les deux serveurs. Le deuxième élément est l'utilisation de composants spécifiques (RunDSCScript et Install MSI using msiexec). Le premier exécute les scripts DSC sur les deux serveurs. Le deuxième composant fait appel à la commande msiexec qui permet, quant à elle, d'exécuter un MSI d'installation.

Lorsque l'on déploie une fonctionnalité sous forme d'une Feature, il n'est pas suffisant de lancer le MSI, l'étape de mise à jour des fichiers de configuration avec les bonnes valeurs est essentielle.

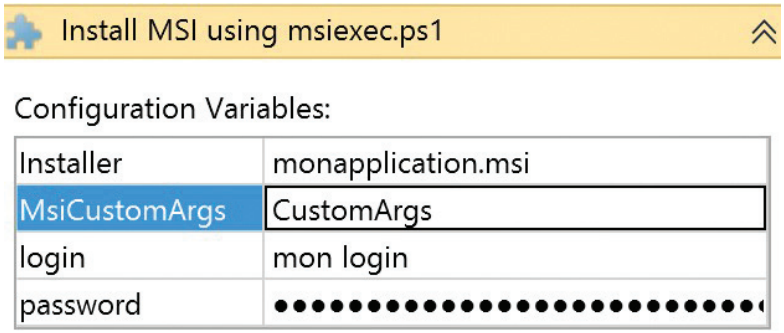
La configuration des composants personnalisés RM permet de configurer des paramètres de configuration de l'application. Il faut, par ailleurs, s'assurer que les valeurs dans les fichiers de configuration soient préfixées et suffixées par '__' (ex : __mavariable__). La figure ci-dessous montre comment configurer un composant RM pour appliquer des valeurs à des variables de configuration :



Paramètres de configuration

Sous l'onglet « Configuration variables », on spécifie le mode de remplacement. Lorsque l'on décide de lancer la modification avant l'installation, on doit spécifier l'extension des fichiers concernés par cette modification (*.xml dans notre exemple). Enfin, on spécifie la liste des variables de configuration (ex : Login et password). Notez bien que la variable Password est encryptée, cela signifie que lors de la saisie la valeur sera masquée.

Voici ci-dessous un exemple d'utilisation de ces variables dans la séquence de déploiement :



Chacune de ces variables peut prendre une valeur par environnement. Ainsi, le compte utilisateur utilisé en intégration et celui en production peuvent être différents.

RELEASE PATH

Comme illustré plus haut, un release path permet de chaîner différents environnements. Ainsi, Release Management poussera le code sur l'environnement de préproduction une fois l'environnement d'intégration déployé correctement.

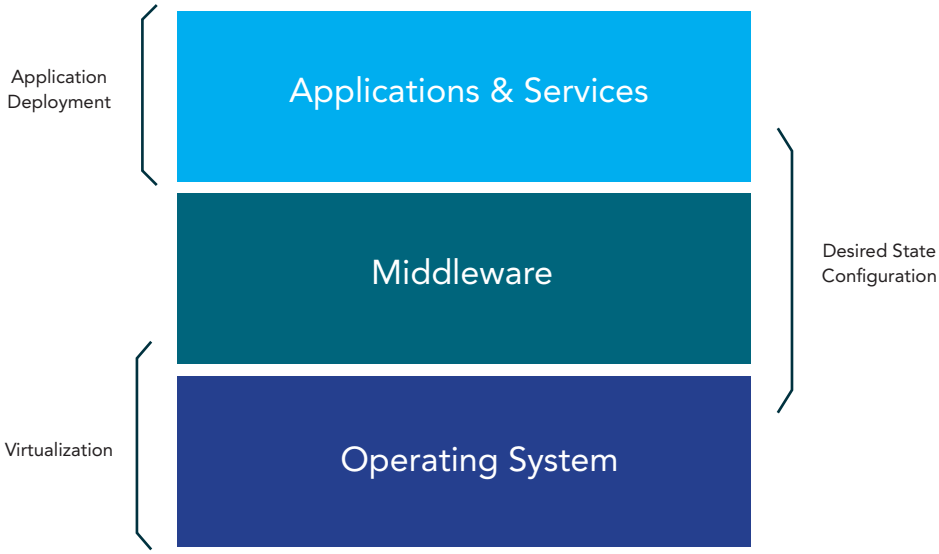
BIEN ÉVIDEMMENT, IL EST POSSIBLE D'AJOUTER DES ÉTAPES DE VALIDATION LE LONG DE CE PROCESSUS.
TROIS BARRIÈRES SONT POSSIBLES PAR ENVIRONNEMENT :

- Une étape pré-déploiement, permettant notamment de planifier le déploiement à une date donnée. Naturellement, il est hors de question de déployer une fonctionnalité nécessitant l'arrêt du site web (par exemple la page d'accueil) en plein milieu de journée. Cette barrière pré-déploiement permet ainsi de planifier le déploiement pour qu'il survienne, par exemple, le week-end en milieu de nuit.
- Une étape de validation technique post- déploiement. Cette étape permet de s'assurer que le déploiement a bien fonctionné et que le site web fonctionne toujours. Elle est à différencier de la troisième barrière, qui est la suivante :
- Une étape de validation fonctionnelle. Ici, l'objectif est de s'assurer que les fonctionnalités, nouvelles, comme anciennes sont bien présentes et pleinement opérationnelles sur le site web.

Dans le chapitre suivant, nous allons nous pencher sur DSC et nous allons mettre le focus spécifiquement sur l'intégration entre Release Management et DSC.

Desired State Configuration

Desired State Configuration (DSC) est une technologie développée par Microsoft qui s'appuie principalement sur PowerShell v4.
Cette technologie répond essentiellement à deux problématiques : la première concerne les différences de configuration entre des machines faisant partie d'un même environnement. L'automatisation de cette configuration via DSC permet effectivement d'écarter les risques liés aux opérations manuelles de configuration.
La deuxième problématique est principalement liée à l'incapacité de gérer au niveau d'un gestionnaire de code source une configuration machine. DSC résout ce problème par le biais d'une configuration machine représentée par un fichier DSC, et qui pourra, par conséquent, être conservée et gérée par TFS.



Desired State Configuration

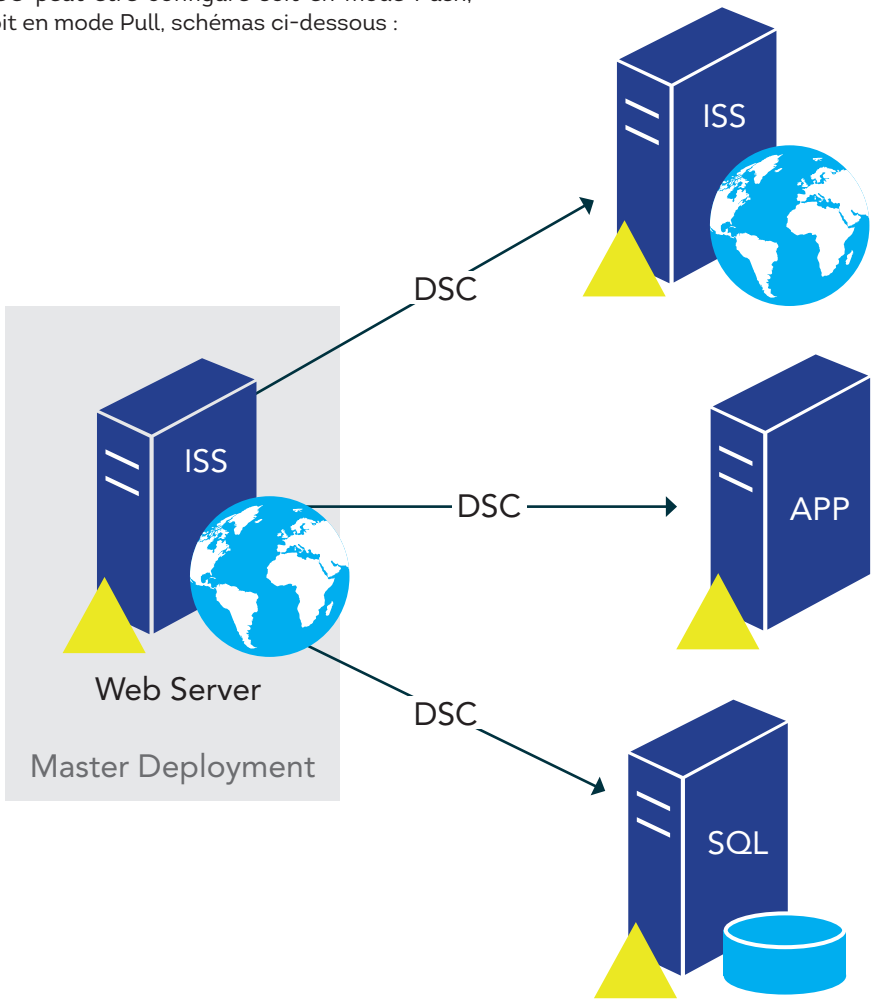
DSC a introduit la notion de ressources. Une ressource est représentée par un module PowerShell contenant trois fonctions qui sont:
1- Get-TargetResource
2- Set-TargetResource
3- Test-TargetResournce
Microsoft propose une série de Ressources DSC installées par défaut. Ces ressources peuvent être étendues et on peut également créer nos propres ressources. Les deux liens ci-dessous abordent plus en détails les sujets DSC :
http://blog.cellenza.com/devops/desired-state-configuration-powershell-dsc-introduction/
http://blog.cellenza.com/devops/desired-state-configuration-dsc-les-ressources/

L'enchaînement des appels de ces trois fonctions donne à DSC son caractère idempotent, ce qui signifie qu'un script DSC peut être exécuté plusieurs fois sans être modifié.

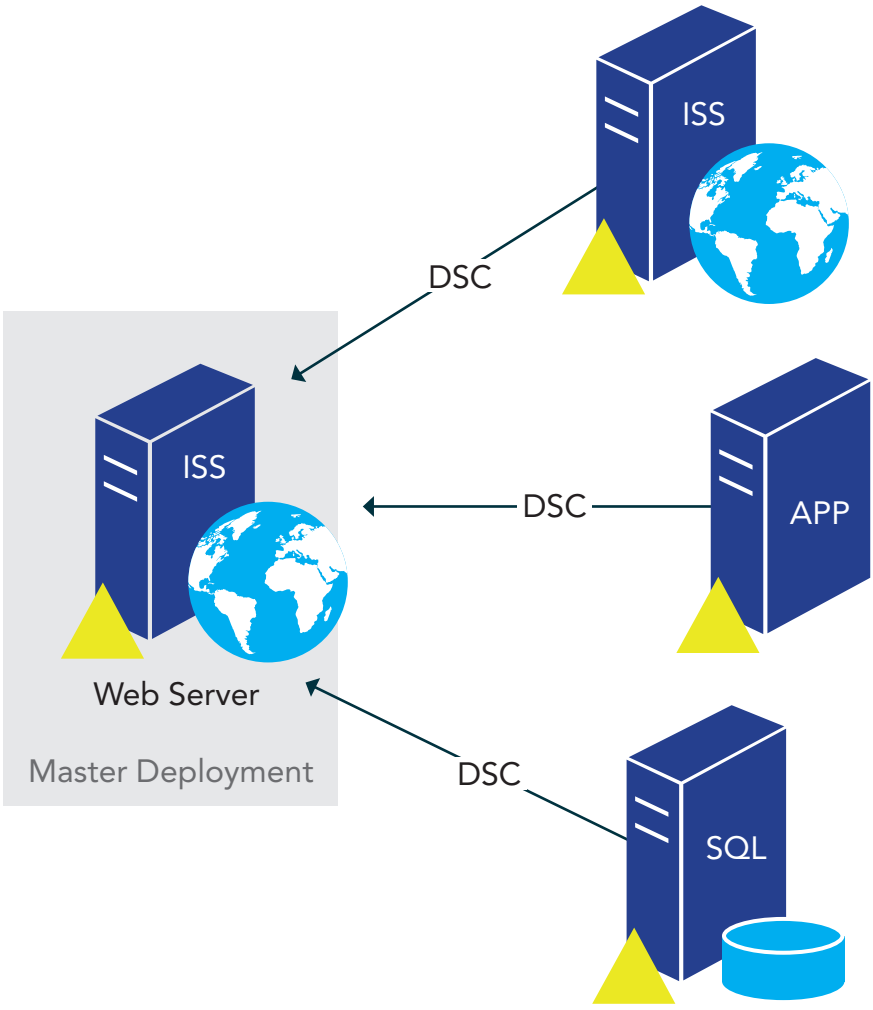
Voici un exemple de script DSC:

```
Configuration IISProvisionning
{

  Node "WIN2012VM"
  {
     WindowsFeature IIS
     {
        Name = "Web-Server"
        Ensure ="Present"
     }
  }
}

IISProvisionning -OutputPath "C:\Scripts"

Start-DscConfiguration -Path "C:\Scripts" -Verbose -Wait
```

Ce script s'exécute sur une machine WIN2012VM et s'assure que la fonctionnalité IIS est installée. Rappelez-vous l'enchaînement des fonctions : la première à s'exécuter est Get-TargetResource. Cette méthode récupère les paramètres d'appel. La seconde fonction est le Test-TargetResource, si cette dernière retourne true¸ la fonction Set n'est pas exécutée, sinon cela signifie que la configuration n'est pas respectée et la fonction Set-TargetResource est déclenchée et effectue la configuration ou l'installation demandée.
DSC peut être configuré soit en mode Push, soit en mode Pull, schémas ci-dessous :



DSC Push mode

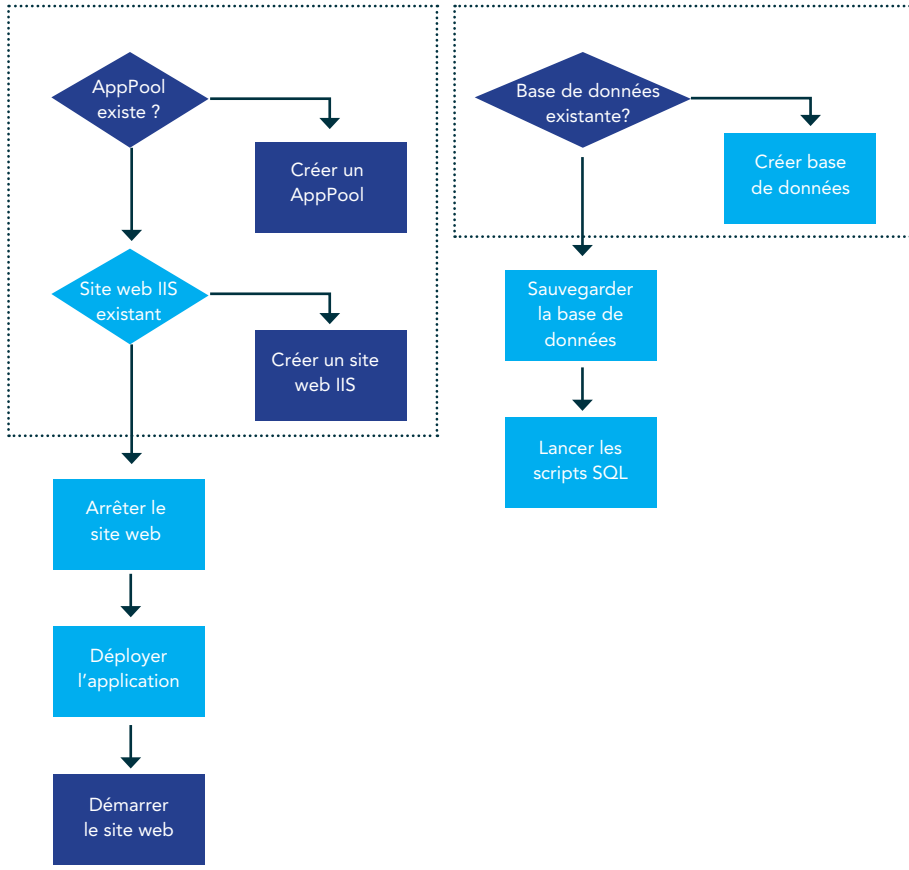


DSC, qui s'appuie principalement sur PowerShell v4, répond à deux problématiques : l'automatisation de configuration entre des machines faisant partie d'un même environnement et la gestion d' une configuration machine au niveau d'un gestionnaire de code source.

DSC pull mode

DSC et Release Management

Les appels DSC depuis Release Management se font via une action RM qui permet de lancer un script PowerShell en passant en paramètre le fichier .ps1 contenant le DSC à exécuter.
Revenons maintenant à notre processus global de déploiement. Les étapes soulignées représentent celles gérées par des scripts DSC.



Le Tableau ci-dessous montre, pour chacune des actions soulignées le script DSC correspondant :

AppPool Existe
& Créer un AppPool
	xWebAppPool AppPool
	{
	Name = «monAppPool»
	Ensure = «Present»
	State = «Started»
	DependsOn = «[WindowsFeature]IIS»
	}

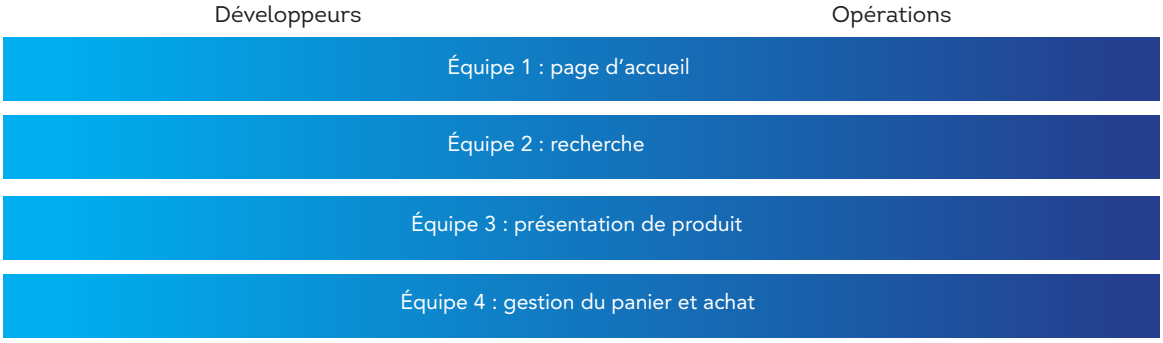
Site web existant
et Créer un site web
	xWebSite WebSite
	{
	Name=»monSiteWeb»
	Ensure = «Present»
	ApplicationPool = « monAppPool «
	PhysicalPath=»C:\inetpub\wwwroot\monSiteWeb «
	State = «Started»
	BindingInfo = MSFT_xWebBindingInformation
	{
	Protocol = «HTTP»
	Port = 81
	}
	}

Arrêter /
Démarrer le site web
	xWebSite WebSite
	{
	Name=»monSiteWeb»
	Ensure = «Present»
	ApplicationPool = « monAppPool «
	PhysicalPath=»C:\inetpub\wwwroot\monSiteWeb «
	State = «Started»
	BindingInfo = MSFT_xWebBindingInformation
	{
	Protocol = «HTTP»
	Port = 81
	}
	}

Base de données existante	xDatabase BasedeDonnés { Ensure = «Present» SqlServer = «ServeurSQL» SqlServerVersion = «2012» DatabaseName = «maBaseDeDonnées» Credentials = (New-Object System.Management. Automation.PSCredential(«login», (ConvertTo- SecureString «xxxx» -AsPlainText -

ET ILS VÉCURENT HEUREUX

Que conclure de la transformation subie par Contoso ? Plusieurs éléments sont à souligner afin de comparer l'avant et l'après.
Le plus visible est le changement d'organisation du département Opérations. Les quatre équipes, structurées par fonction, présentes avant la transformation DevOps, ne sont plus. Chaque membre du département Opération fait désormais partie d'une ou plusieurs équipes fonctionnelles. Par ailleurs, pendant la phase de construction d'un produit, l'ensemble de l'équipe est responsable de la bonne santé de la production et plus seulement l'équipe Monitoring. Cela contribue à responsabiliser tous les acteurs du projet.



Organisation de Contoso, après DevOps

Second changement de taille : le métier des Opérationnels. Alors qu'ils étaient auparavant en charge de récolter les besoins des produits en termes de qualité et déploiement, puis de réaliser des actions manuelles ou semi-automatisées afin d'amener le produit vers le client, ils sont maintenant en charge de définir l'automatisation du processus de test et de livraison et de le mettre en œuvre. De même, la notion de monitoring a changé, puisque l'on passe de métriques purement dédiées aux opérations (charge CPU/RAM/Disque, index de bases de données…) à des métriques applicatives qui ont une sémantique bien plus riche et pertinente. Bien entendu, les métriques opérationnelles restent valables, mais sont désormais doublées par celles dédiées au métier et aux développeurs.

Troisième changement : Le déploiement est désormais entièrement automatisé. Bien que des validations manuelles demeurent afin de permettre les tests de non-régression, les opérations de packaging des binaires, de configuration des serveurs et de déploiement des binaires font partie d'un processus automatisé, supprimant toute possibilité d'erreur humaine dans ces étapes.

Dernier changement enfin, le plus visible de tous : les livraisons sont beaucoup plus rapides ! Dans le cadre de Contoso, le choix d'une livraison indépendante de chaque fonctionnalité conduit à avoir, plutôt que 3 ou 4 grosses livraisons annuelles, une myriade de petites livraisons, chacune n'apportant, certes, qu'une amélioration réduite, mais l'apportant rapidement. Le métier voit ainsi ses idées plus rapidement mises en œuvre et les équipes ont un feedback plus rapide.

Cycle de vie d'une livraison, après DevOps

Il faut bien garder en tête, cependant, que cette évolution a un coût. Contoso a dû, en effet, investir énormément afin de rendre cela possible. Faire évoluer les métiers des opérationnels, mettre en place les outils, tout cela demande du temps et des efforts. Cependant, le résultat, offrant un produit livré plus vite, avec un niveau de qualité supérieur, montre que cet investissement est rentable.

4 transformations majeures grâce à DevOps : L'organisation du Département Opérations, le métier des Opérationnels, l'automatisation des déploiements et les délais de livraison plus courts.

AUTEURS

Mick Philippon

Idriss Selhoum

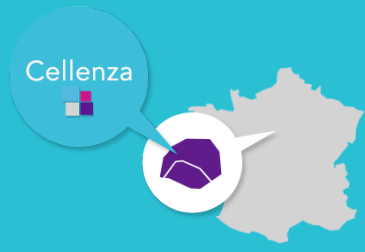
REMERCIEMENTS

Anne Pedro (Notre illustratrice de talent),
Jason De Oliveira, Nicholas Suter,
Michel Perfetti, Arnaud Hego,
Michel Hubert, Emilie Fruh,
Radoine Douhou et Fathi Bellacene.
Un grand merci également à toutes
les équipes Microsoft pour leur soutien
dans la réalisation de ce projet.

Cellenza

Does IT better

Cellenza est un cabinet de conseil IT dédié aux technologies Microsoft et aux Méthodes Agiles.



Cellenza
156 boulevard Haussmann,
75008 PARIS
Tel : 01 84 17 33 67

http://blog.cellenza.com/ www.cellenza.com