Stratégies de déploiement d'un projet de GCL

Jean-François Castaing, consultant

Dans les grandes organisations, la mise en œuvre d'un projet de Gestion de Configuration Logicielle (GCL) représente un effort important, tant sur le plan budgétaire que sur le plan humain, et se déroule en général sur plusieurs mois. Il est donc important, pour la réussite du projet, de se poser assez tôt la question de la stratégie de déploiement. Cet article présente quelques stratégies possibles et recense les questions que doivent se poser les responsables d'un tel projet.

L'importance de la stratégie de déploiement du projet de GCL

Plusieurs facteurs peuvent expliquer les difficultés inhérentes à la mise en œuvre d'un projet de GCL dans les grandes organisations :

- la variété des acteurs impliqués dans un tel projet ; pratiquement tous les « métiers » de la DSI sont concernés par la GCL : les équipes de développement, bien sûr, mais aussi les équipes d'exploitation informatique, les ingénieurs sécurité, système et réseaux, les administrateurs de données, etc. De plus en plus, les représentants des directions utilisatrices sont également impliqués puisque la GCL s'intègre désormais dans le processus plus large de la Gestion des Changements Applicatifs. Cette multiplicité d'acteurs conduit souvent à l'expression de besoins très divers et parfois contradictoires ;
- dans la majorité des cas, il existe dans l'organisation une « solution » de GCL déjà en place, quelles que soient sa nature – développement interne ou solution commerciale - et la richesse des fonctionnalités offertes aux utilisateurs ; dans tous les cas, une action de migration devra être envisagée ; cette migration offre en général peu de difficultés techniques mais elle peut induire des complications organisationnelles et psychologiques ;
- le patrimoine applicatif est, en général, très hétérogène ; c'est l'héritage d'un historique complexe de restructurations de l'entreprise, de développements logiciels et de maintenances. Ces travaux informatiques ont été réalisés au fil du temps aux différents stades des évolutions technologiques. La GCL ne remplissant vraiment son office que si elle supporte l'intégralité du Système d'Information, cette hétérogénéité nécessitera la prise en compte de nombreuses spécificités ;
- le Système d'Information des grandes organisations s'appuie de moins en moins sur des applications spécifiques développées par des équipes internes ; le recours aux progiciels se généralise, le recours à des prestataires externes – sous forme d'outsourcing, de délocalisation, de tierce maintenance – devient la règle plutôt que l'exception ; ces nouvelles « pratiques » ne doivent pas être exclues du champ d'action de la GCL mais nécessitent une approche particulière ;
- enfin, dans la période récente, les projets de déploiement de la GCL sont souvent perçus comme des projets d'infrastructure internes aux DSI. N'ayant pas de visibilité de la part des « clients » ils ne sont pas prioritaires. Les restrictions risquent de frapper immédiatement ces projets de mise en œuvre, y compris dans le cas où un effort budgétaire d'achat d'une solution logicielle avait été consenti. Cette situation évolue lentement.

Face à cet ensemble de difficultés, il est donc très important d'appréhender le projet de déploiement avec toutes ses composantes : planification, budget, formation, aspects techniques, communication, etc.

Avant le début du déploiement

Entre le moment où la solution logicielle de GCL a été choisie et celui où le déploiement peut commencer, un certain nombre d'étapes doivent avoir été réalisées :

- définition des processus d'entreprise GCL ;
- prise en main technique de l'outil de GCL ;
- formation des utilisateurs ;
- développement et tests des procédures de chargement des applications ;
- définition des processus transitoires ;
- développement et tests des procédures transitoires ;
- organisation du support interne.

Déploiement « Big Bang » ou déploiement progressif ?

Il est clair qu'un déploiement « Big Bang » présente un certain nombre d'avantages indéniables qui poussent à envisager d'emblée cette solution :

- le projet de déploiement est mené d'un seul jet et donc avec une bonne visibilité sur la date de fin prévue ;
- il n'y a pas de période temporaire de « cohabitation » entre les anciennes procédures de GCL et les nouvelles, ni de procédure technique particulière à mettre en place pour gérer cette cohabitation ;

- par voie de conséquence, le budget consacré au déploiement s'en trouve réduit, même s'il est concentré sur une période courte.

Malheureusement, un démarrage « Big Bang » n'est pas toujours envisageable, pour des raisons de budget, d'organisation, de formation, de disponibilité des équipes. Il faut alors choisir une stratégie de déploiement progressif, définir le nombre de « vagues successives » de déploiement, les critères de définition du périmètre de chacune d'elles, le calendrier des opérations, les modalités de traitement des périodes transitoires.

Il vient alors spontanément à l'esprit d'envisager de déployer le projet de GCL par Applications ou regroupements d'Applications. Il s'agit en effet de périmètres assez homogènes sur le plan des liens entre composants applicatifs, sur le plan des équipes en charge du développement et de la maintenance, sur le plan des technologies utilisées (langages de programmation, SGBD). Mais si on examine d'un peu plus près les solutions possibles, on s'aperçoit assez rapidement que ce n'est pas le seul critère disponible, ou en tout cas, que ce critère peut être combiné avec d'autres pour définir une démarche de déploiement optimale.

Dans la suite de cet article, nous allons examiner plusieurs autres « axes de déploiement » possibles ; et préciser pour chacun d'eux, les phases possibles de déploiement et les raisons pour lesquelles une organisation peut avoir à s'y intéresser.

Tout scénario de déploiement consistera alors à choisir le ou les axes selon lesquels le déploiement se fera de manière globale (déploiement « en largeur ») et les axes selon lesquels il se fera de manière progressive (déploiement « en profondeur »).

On peut envisager alors dix axes principaux de déploiement possibles de la GCL, ceux-ci étant éventuellement combinables entre eux :
- le déploiement par domaine applicatif ;
- le déploiement par étape du cycle de vie ;
- le déploiement par plate-forme technique ;
- le déploiement par fonctionnalité GCL ;
- le déploiement par degré d'utilisation des composants ;
- le déploiement par projet ;
- le déploiement par technologie de développement ;
- le déploiement par type de composants ;
- le déploiement par mode d'organisation des développements ;
- le déploiement par type de processus.

Axe 1 : déploiement par domaine applicatif

Comme nous l'avons écrit plus haut, c'est l'axe de déploiement qui est le plus fréquemment envisagé comme hypothèse de départ. Plusieurs raisons plaident en sa faveur : le clivage en domaines applicatifs recouvre souvent d'autres clivages : organisationnels (les équipes de développement sont souvent organisées par domaines fonctionnels) ou techniques (les plates-formes qui supportent les applications, les langages de programmation, les systèmes de gestion de bases de données mis en œuvre au travers du temps, l'ont en général été de manière homogène au sein de chaque domaine applicatif). Il semble donc plus facile de créer des ensembles cohérents d'Applications qui seront placés par étapes successives sous le contrôle de la GCL. En outre cela pourra simplifier l'organisation et la planification des formations des équipes de développement.

En revanche, si on retient ce critère de déploiement, il faut réaliser des actions préparatoires :
- on doit être capable de rattacher chacun des composants applicatifs existant à une Application et une seule, ce qui impose un travail de recensement et de recherche de « propriétaires » qui peut s'avérer important s'il n'a jamais été clairement effectué auparavant ;
- on doit être capable d'identifier les composants « partagés » qui ont une utilisation dépassant le contexte de leur Application d'appartenance : ce sont des composants développés dans le contexte d'une Application mais utilisés par d'autres (définition des structures de données des fichiers ou bases de données, modules fonctionnels ou techniques implémentant des règles de gestion, définition des zones de communications entre modules) ; il est intéressant d'observer, dans chaque couple d'applications, dans quels sens se font ces échanges (plus de modules de A1 utilisés dans A2 ou l'inverse) ;
- on doit éventuellement identifier une Application spécifique à laquelle on rattachera tous les composants d'usage général, souvent techniques, qui n'ont pas de contenu métier (modules de calculs, modules de communication, mise en forme de messages, contrôle de sécurité).

On peut ensuite envisager de regrouper les Applications en lots homogènes qui seront placés en une seule étape sous contrôle de l'outil de GCL. On essaiera de regrouper dans un même lot des Applications qui ont beaucoup d'interactions (composants partagés évoqués ci-dessus).

En parallèle, il convient d'étudier le calendrier prévisionnel des projets de développement et des maintenances majeures afin d'en tenir compte : il peut ne pas être opportun de migrer une Application sous le contrôle de la GCL avant la livraison en production d'un important projet ; a contrario, il peut

être intéressant de le faire au début d'un projet de développement.

Lorsqu'on a une vision claire des Applications qui constituent chaque lot de migration et de la séquence de migration des différents lots, il convient d'étudier les modalités de gestion des situations transitoires. La question peut s'exprimer ainsi : quelle procédure devra-t-on appliquer lorsqu'on doit modifier un composant applicatif « partagé » entre une application migrée et une application non-migrée et que cette modification a un impact sur les deux applications ? La réponse peut être technique et/ou organisationnelle. Dans tous les cas, elle doit garantir contre une régression ou une désynchronisation au moment de la livraison en production.

Axe 2 : déploiement par étape du cycle de vie

Cela semble une évidence : la mise en place d'un projet de GCL doit couvrir la totalité du cycle de vie des développements et des maintenances, c'est-à-dire :

- la prise de contrôle d'un composant en vue de maintenance (check out) ;
- la modification des composants source (stage) ;
- la fabrication des exécutables (build) ;
- la promotion (ou la rétrogradation) vers (ou depuis) les environnements de tests ;
- la livraison des composants modifiés et testés vers le référentiel des composants (Check In) ;
- le déploiement effectif sur les plates-formes d'exécution (Installation).

On peut toutefois envisager d'autres stratégies consistant à déployer la GCL sur les Applications, au moment où les changements applicatifs atteignent un stade donné de leur cycle de vie. La migration des composants sous le contrôle de la GCL se fait alors « au fil de l'eau », toutes applications confondues.

Cette stratégie se justifie lorsque, globalement, le projet de mise en place de la GCL répond à un besoin clairement identifié dans l'organisation et vise à remédier à un dysfonctionnement majeur. Trois situations caractéristiques peuvent illustrer ce fait :

- si on a identifié une lacune dans l'industrialisation du développement et de la maintenance, on mettra en place la GCL de manière prioritaire sur les activités de début du cycle de vie : développement, fabrication des exécutables et tests unitaires ;
- si on doit résoudre un problème d'intégration (plusieurs équipes travaillent sur des sous-ensemble de l'application mais peinent à se coordonner) ou un problème d'articulation entre maîtrise d'ouvrage et maîtrise d'œuvre (difficulté à relier ce qui est livré en recette avec les demandes fonctionnelles émanant des utilisateurs, difficulté à organiser et à exécuter les tests de recette utilisateurs), on mettra en place la GCL sur la partie centrale du cycle de vie (Tests et Assurance Qualité) ;
- si on veut résoudre un problème de communication entre Études et Production, on mettra en place la GCL sur la partie aval du cycle de vie (livraison dans un référentiel de composants, suivie du déploiement effectif en production).

On peut ensuite étendre la couverture de la GCL aux étapes amont et aval.
Cette approche présente l'avantage d'induire un retour rapide sur investissement puisqu'elle traite de façon précise une priorité identifiée ; de manière corollaire, elle est plus facile « à vendre » au sein de l'organisation : ce n'est plus seulement un projet d'infrastructure technique propre à la DSI, c'est un projet d'entreprise.

Mais elle présente un inconvénient : on doit également gérer des situations transitoires pour les activités à la frontière de ce qui est déjà couvert par la GCL et ce qui n'y est pas encore. En outre, la migration étant liée à la dynamique des projets en cours, il faudra une action volontariste pour migrer, en fin de projet, les composants applicatifs rarement touchés par des évolutions.

Axe 3 : déploiement par plate-forme technique

Les grandes organisations ont pratiquement toutes désormais un Système d'Information supporté par des plates-formes techniques multiples : mainframes, serveurs départementaux, postes de travail. Elles ont néanmoins jusqu'ici rarement choisi l'approche consistant à mettre en place la GCL sur toutes leurs plates-formes techniques simultanément.

On peut l'expliquer par des raisons historiques (hétérogénéité des équipes, culture GCL différente selon les équipes, plus ou moins grande stabilité des applications selon les plates-formes). Mais, surtout, cette solution présente l'inconvénient d'exiger en amont qu'un grand nombre d'aspects aient été intégralement couverts avant que le déploiement puisse commencer :

- achats des licences des outils de GCL pour l'ensemble des plates-formes ;
- résolution des questions techniques de mise en œuvre pour toutes les plates-formes (paramétrage de la GCL, paramétrage des procédures de « build », interfaçages éventuels avec les outils de développement et de test, recensement des procédures de mise en exploitation effective et paramétrage de celles-ci dans la GCL) ;
- travail de recensement des composants par application ;
- mise au point des supports de formation.

De plus, rares sont les organisations dont la majorité des applications est construite sur une architecture mixte Mainframe + Distribué (on peut estimer que

moins de 25 % des composants correspondent à des applications mixtes Mainframe + Distribué) ; il est donc rarement justifié de soumettre le déploiement de la GCL à autant de contraintes techniques ou financières préalables pour une justification reposant sur une part aussi faible du S.I.).

Toutefois, cette approche est justifiée lorsque la GCL est mise en place au titre d'un projet stratégique sur le plan métier ou sur celui du choix d'une nouvelle architecture technique destinée à devenir le standard de tous les nouveaux développements.

Axe 4 : déploiement par fonctionnalité GCL

En surplus d'un noyau commun de fonctionnalités attendues de la GCL, les outils modernes de GCL offrent aux utilisateurs un grand nombre de fonctionnalités avancées, destinées à couvrir des besoins extrêmement larges. En contrepartie, certaines fonctionnalités avancées peuvent s'accompagner d'une mise en œuvre complexe, nécessitant une formation poussée des utilisateurs. D'autres peuvent exiger un surcroît de ressources matérielles (puissance des machines, espace de stockage) ou un paramétrage approfondi.

À l'occasion du déploiement d'un tel outil, les organisations peuvent choisir d'offrir immédiatement aux utilisateurs, l'accès à l'ensemble des fonctionnalités proposées par l'outil ou au contraire, de commencer par les fonctions de base en remettant à une extension ultérieure, post-déploiement, l'ouverture à des fonctionnalités avancées.

Les critères de choix à prendre en compte sont les suivants :

- on ne peut raisonnablement proposer une solution de GCL qui ne soit pas au moins équivalente aux services dont disposaient les utilisateurs au préalable ; ils y verraient une régression ; on doit même aller au-delà et proposer « un bonus » ; il ne faut pas oublier qu'un projet de GCL n'est pas toujours très « vendeur » à l'intérieur d'une DSI, en particulier auprès des équipes de développement qui y voient une restriction de leur autonomie ou une charge de travail supplémentaire. La capacité à proposer des fonctionnalités qui allègent ou facilitent certains aspects de leur travail quotidien aidera à éliminer cet écueil ;
- on doit également s'appuyer sur les « facteurs déclencheurs » du projet de GCL lui-même : si l'organisation a souhaité conduire ce projet, c'est en fonction des critiques adressées à la situation existante ou en fonction d'évolutions stratégiques souhaitées ; il faut donc proposer un jeu de fonctionnalités de départ, cohérent avec ces objectifs globaux ;
- un point important à prendre en compte est la formation initiale ainsi que la nécessaire assistance aux utilisateurs après le démarrage : toutes les fonctionnalités offertes doivent avoir été abordées au cours des séminaires de formation – si possible accompagnées d'exercices pratiques. Les personnes chargées du support interne doivent également avoir acquis suffisamment d'expérience par elles-mêmes pour être en situation d'apporter des réponses précises sur toutes les fonctionnalités offertes.

Parmi les fonctionnalités dont la mise en œuvre peut être différée à une étape ultérieure, citons en particulier tout ce qui touche au développement parallèle. Par développement parallèle on entend toute situation dans laquelle, pendant un laps de temps donné, plusieurs personnes ou équipes sont amenées à travailler en modification sur les mêmes composants ou sur des composants fonctionnellement ou techniquement liés les uns aux autres. Au-delà de cette définition générale, les cas de développement parallèle peuvent recouvrir des situations très diverses (selon la durée de cette période de parallélisme, selon le volume de composants concernés, selon qu'il s'agit de modifications ponctuelles ou de Releases complètes, etc.).

Une gestion complète du Développement parallèle exige la mise en œuvre d'un grand nombre de mécanismes :

- possibilité de travailler dans des environnements de développement, séparés ;
- information réciproque des différents intervenants (chacun doit être informé que quelqu'un d'autre travaille en parallèle sur le même jeu de composants ; il doit également être tenu informé de la progression de ses collègues dans leur propre cycle de vie de maintenance) ;
- possibilité de tester des modifications parallèles dans des environnements de tests, séparés, utilisant si nécessaire des jeux de données de tests différents ;
- fourniture de moyens automatiques ou semi-automatiques de comparaison, de réconciliation ou de report des modifications ;
- moyens d'assurer la traçabilité des modifications effectuées en parallèle ou des installations dans des environnements de tests ;
- possibilité de décider, même très tardivement, quelle branche parallèle arrivera en production la première et dans quel sens les reports de maintenance devront s'effectuer.

On voit, à partir de cet exemple, qu'il peut être préférable de ne pas ouvrir immédiatement une telle fonctionnalité si on n'a pas les moyens de l'assurer avec tout le niveau de fiabilité et d'intégrité requis.

Axe 5 : déploiement par degré de réutilisation des composants

Il est couramment admis que l'on peut classer les composants d'une Application par degré de réutilisation ; on distinguera par exemple :

- (type 1) les composants développés pour l'ensemble du Système d'Information de l'organisation et qui ne sont pas rattachés à telle ou telle Application (ex. : modules de calcul ou de contrôle, module de gestion de la sécurité, etc.) ;
- (type 2) les composants communs développés dans le cadre d'une Application mais dont l'utilisation est limitée à cette Application (ex. : règles de gestion communes) ;
- (type 3) les composants communs développés dans le contexte d'une Application mais destinés à être intégrés par appel dans d'autres Applications (ex. : encapsulation des accès aux données physiques) ;
- (type 4) les composants spécifiques à une fonction utilisateur au sein d'une Application et qui ne sont jamais utilisés en dehors de ce contexte (ex. : Interface Homme Machine).

On imagine assez bien qu'il est possible de migrer progressivement les composants du S.I. sous le contrôle de la GCL, dans l'ordre ci-dessus (types 1 et 2 d'abord, type 3 ensuite, type 4 enfin - en une ou plusieurs vagues) : on sécurise ainsi les composants les plus centraux d'abord et on termine par ceux qui sont les plus « périphériques ».

Cette solution présente l'avantage de démarrer la GCL sur un nombre limité de composants, donc sans investissement initial lourd, tout en apportant des bénéfices immédiat ; c'est aussi l'occasion de faire naître auprès de tous les intervenants une « culture GCL » là où il n'y en avait peut-être pas.

En revanche, elle présente l'inconvénient de nécessiter une formation de presque tous les utilisateurs en même temps, si la responsabilité de développer et tester les composants communs n'est pas concentrée sur un petit nombre de personnes.

Cette approche est particulièrement intéressante dans les organisations qui ont déjà développé une culture de « génie logiciel », à base de démarche méthodologique et/ou d'AGL.

Axe 6 : déploiement par projet

Cette stratégie consiste à attendre qu'on soit effectivement amené à développer de nouveaux composants ou à modifier des composants existants pour les placer sous contrôle de la GCL.

On considère alors le « référentiel » des composants comme un stock de composants qui se vide au fur et à mesure des modifications, les composants impactés par une maintenance étant réinjectés sous le contrôle de l'outil de GCL comme s'il s'agissait de nouveaux développements.

Elle présente l'intérêt d'avoir un impact faible en termes de charge ; en quelque sorte, l'effort à faire pour passer sous contrôle de la GCL est indolore, « noyé » dans la charge des projets eux-mêmes. En revanche, elle risque d'être totalement inefficace si on n'a pas la certitude que des projets réels d'évolution concerneront une part significative du S.I. dans les deux années à venir !

Elle est envisageable dans les organisations qui ont vraiment très peu de moyens à consacrer expressément au déploiement de la GCL. Elle n'a de sens également que s'il n'y avait en place aucune solution réelle de GCL mais tout au plus un simple stockage des composants de référence.

En contrepartie, elle cumule plusieurs inconvénients :

- avant toute maintenance, les développeurs doivent identifier quels composants seront à rechercher dans l'ancien référentiel et lesquels dans le nouveau ;
- cette double gestion rend difficile les fonctions d'analyse d'impact ;
- il faudra de toutes façons migrer de manière autoritaire les composants restant (jamais modifiés depuis le début du projet GCL) après un délai raisonnable ;
- elle nécessite d'entretenir en permanence les compétences techniques spécifiques à un déploiement GCL (recensement de composants, automatismes d'introduction dans la GCL, paramétrages) ;
- associer démarrage de projet applicatif et passage sous la GCL peut mettre une pression supplémentaire sur des équipes de développement et de production à un moment où ce n'est pas utile (de toutes façons, si on choisit cette technique, il faut bien le faire au démarrage du projet, pas en phase de recette ni de livraison en production !) ;
- psychologiquement, il y a également un risque de faire porter, à la mise en œuvre de la GCL, la responsabilité des dérapages de délais ou de budgets du projet applicatif…

Axe 7 : déploiement par technologie de développement

Dans les grandes organisations qui ont construit leur Système d'Information par strates successives au fil du temps, on rencontre fréquemment une très grande diversité de technologies : langages de programmation, Ateliers de Génie Logiciel ou générateurs de code, outils de design de l'interface homme machine, outils de gestion des données – système de fichiers ou SGBD, outils d'automatisation des tests, serveurs de transactions, outils d'automatisation de la production.

La majorité des outils de GCL du marché sont capables de s'adapter à ces différents types d'outils pour automatiser au maximum les actions de modification des composants, de fabrication des exécutables, d'installation des exécutables dans les différents environnements de tests et de production.

Il n'en reste pas moins que chaque technologie doit être étudiée de manière particulière en vue de son intégration dans le processus de GCL sous plusieurs angles :

- mode d'utilisation dans l'organisation ;
- mode de stockage des composants (fichiers séparés, base de données ouverte, référentiel propriétaire) ;
- identification des composants et règles de nommage ;
- règles de fabrication des exécutables (paramètres, constitution des dépendances, priorité de recherche en cas d'existence multiple dans des environnements différents) ;
- modalités techniques d'installation dans les environnements d'exécution ;
- interfaces, API, fonctions d'import-export avec les référentiels utilisés par les outils ;
- identification des différents profils d'utilisateurs habilités à travailler sur les composants.

Chaque type de composant nécessitera une charge de paramétrage, de développement spécifique dans- ou autour- de l'outil de GCL, de documentation et de tests afin de garantir l'intégrité des informations et le support du cycle de vie complet. Sur la somme des outils présents dans l'entreprise, il ne sera peut-être pas possible de réaliser l'ensemble de ces travaux pour le déploiement initial.

On commencera éventuellement par identifier les technologies cantonnées à des Applications en fin de vie ou peu évolutives pour se demander jusqu'à quel point une intégration poussée de ces technologies dans l'outil de GCL est justifiée.

On passera ensuite en revue l'ensemble des technologies à intégrer pour répondre aux questions évoquées ci-dessus. L'objectif étant d'essayer de trouver des réponses qui soient homogènes à la fois vis-à-vis des besoins de la GCL et des modes opératoires proposés aux utilisateurs : ainsi, si on doit gérer des structures de données relatives à trois SGBD différents, on essaiera de trouver un mode technique d'intégration qui soit comparable pour chacun d'eux. Un chiffrage du coût d'intégration de chaque technologie est ensuite élaboré.

Il convient ensuite d'établir les priorités d'intégration de ces différentes technologies en tenant compte de ces coûts estimés ainsi que des besoins des Applications qui s'appuient sur ces technologies : l'idée sous-jacente est que, si une Application très stratégique pour l'entreprise doit connaître d'importants projets d'évolution, les différentes technologies qu'elle utilise doivent être intégrées en priorité dans la solution de GCL ; les choses ne sont, bien sûr, pas toujours aussi simples…

Axe 8 : déploiement par type de composants

Dans l'esprit de tous les utilisateurs, la GCL s'applique en premier lieu aux composants source et exécutables des Applications. Ce n'est pourtant pas son seul champ d'application.

On peut regrouper les composants applicatifs en trois sous-ensembles :

- les composants « Études » : sources de programmes, description de structures de données communes, définition de l'interface homme machine, exécutables associés, documentation techniques, modèles de conception ;
- les composants « Production » : JCL ou scripts de commandes, paramètres d'exploitation, définitions d'enchaînements de tâches (Ordonnanceurs) ;
- les composants « d'administration de données » : scripts de définition de structures de bases, objets dérivés (ex. : définition de Vues relationnelles ; autorisations d'accès).

Cette segmentation se justifie par la nécessité de processus techniques très différents.

Ainsi, les types de « rôles » concernés ne sont pas les mêmes :

- chefs de projets et développeurs dans un cas ;
- ingénieurs ou analystes de production dans l'autre ;
- administrateurs de données ou de bases de données dans le dernier cas.

Par ailleurs, les règles d'évolution dans le cycle de vie sont différentes et nécessitent des réponses différentes :

- une fois qu'un programme a été modifié en développement et que l'exécutable a été fabriqué, celui-ci sera installé tel quel (ou désinstallé) dans les environnements successifs de tests ou de production ;
- un JCL est adapté à chaque environnement (Tests Unitaires, Recette, Production) ; il n'y a pas d'ordonnanceur en tests ! ;
- un script de création de base de données n'est pas « installé » dans un environnement mais exécuté ; on ne revient pas en arrière par une simple désinstallation.

Pour ces raisons, on met souvent la GCL en place sur ces typologies de composants en autant de vagues successives ; par ailleurs, la couverture des objets de production (JCL, Ordonnanceurs) par la GCL est en général précédée d'une refonte profonde des normes en ce domaine.

Axe 9 : déploiement par mode d'organisation des développements

Souvent, la mise en place de la GCL s'envisage sur les seules activités de développement ou de maintenance des applications internes de l'organisation. C'est une erreur, car les besoins de sécurisation, de traçabilité, d'automatisation sont les mêmes dans toutes les situations où un « changement applicatif » doit être géré, quel que soit le mode d'organisation que l'on adopte pour ce faire.

À titre d'exemple, on citera trois modes d'organisation différents qui peuvent poser des problèmes spécifiques et donc exiger des réponses adaptées.

La Tierce Maintenance sous-traitée à un partenaire mais s'exerçant sur le site de l'organisation.

Dans cette situation, la totalité du cycle de vie des changements de composants s'exécute sous le contrôle de la GCL, mais il peut être scindé en deux phases :

- l'une recouvre l'activité de développement, de fabrication des exécutables et de tests informatiques (tests unitaire et d'intégration),
- l'autre recouvre la recette utilisateurs, la livraison dans le référentiel de composants et la mise en production.

Il faut donc prévoir une séparation assez nette entre les environnements de composants et de données de tests impliqués dans ces deux phases ainsi qu'un mécanisme assez formalisé de passage de relais à la frontière de ces deux phases.

« L'outsourcing » intégral.

Dans ce cas, on suppose que le partenaire livre à son client des collections de composants développés, fabriqués et testés, sans préjuger du processus de GCL qu'il a pu mettre en place pour ses besoins propres. Du point de vue du client, le cycle de vie commence par l'installation dans un environnement de recette utilisateurs, suivi ou précédé d'un test de non-régression. Il faut tenir compte aussi du fait que le partenaire peut livrer soit des composants source (avec refabrication en masse des exécutables au moment de la réception par le client) soit des exécutables seuls.

Le recours à des progiciels applicatifs ou techniques.

En première analyse, cette situation peut sembler n'avoir que peu de rapports avec notre sujet. On s'aperçoit pourtant rapidement que tout progiciel connaît, au fil du temps, des versions successives et qu'il incombe assez largement à l'organisation cliente de gérer les conséquences liées à ces changements de versions : à chaque livraison d'une nouvelle version du progiciel par l'éditeur, elle doit s'assurer que son installation en environnement de production ne conduira pas à des régressions ni à des défaillances. On retrouve donc des objectifs habituels de la GCL qu'on devra atteindre par des moyens spécifiques, compte tenu du fait que cette situation présente deux particularités majeures :

- les éditeurs de progiciels livrent à intervalles réguliers des versions de leurs outils, constituées d'un « package d'installation » générique (non adapté à l'utilisation qu'en fait chaque client) ne contenant que des composants exécutables, de la documentation, des « modèles » d'utilisation (« samples » ou « templates ») ; il reste à la charge du client de déterminer quelles différences cette version inclut par rapport à la précédente et donc quel type de ré-installation est nécessaire ;
- les clients qui utilisent un progiciel doivent toujours en spécialiser le fonctionnement (« customisation ») soit par un paramétrage, soit par le développement de composants spécifiques dans l'outil lui-même (Exits, Panels, Rapports), soit à l'extérieur du progiciel en utilisant des interfaces ; chaque nouvelle version du progiciel nécessite de se poser la question de l'adaptation de ces développements spécifiques.

Les étapes à exécuter sont grossièrement les suivantes :

- installer la nouvelle version du progiciel, sans aucune customisation dans un environnement de développement séparé (NVB = « nouvelle version brute ») ;
- comparer l'ensemble de l'environnement ainsi initialisé avec le contenu de la dernière version installée sans customisation (AVB = « ancienne version brute) ; les outils de GCL contiennent tous des fonctions de comparaison qui identifient les composants supprimés, ajoutés ou modifiés entre deux Releases d'un Projet ;
- analyser en détail la nature et le contenu de ces différences pour évaluer les impacts sur la version « customisée » ; les conséquences peuvent être variées : certaines customisations seront à reporter telles quelles sur la nouvelle version, certaines seront à adapter, certaines pourront disparaître (développements spécifiques qui sont désormais proposés en standard dans la nouvelle version du progiciel) ; les fonctions de comparaison et de fusion des outils de GCL permettent là aussi d'assister l'utilisateur ;
- créer un « Change Package » ou un « Projet » qui contiendra tous les composants impactés par la nouvelle version, qu'ils proviennent du progiciel lui-même ou de la customisation faite localement ;
- exécuter les actions habituelles de codage des composants sources ou des paramètres, de fabrication des exécutables dans l'environnement de développement NVB ;
- promouvoir ce Package ou ce Projet vers un environnement de tests de non-régression et y exécuter les tests correspondants ;
- lorsque le processus de qualification est validé, on « installe » la nouvelle version customisée dans les environnements de production.

À chaque passage d'un environnement à l'autre, jusqu'à trois séries d'actions peuvent être enchaînées : installation du progiciel « brut » avec les procédures fournies par l'éditeur, exécution d'utilitaires de migration des données du progiciel – si le méta-modèle sous-jacent a changé – et installation des composants relevant de la customisation.

On voit mieux sur cet exemple, comment les principes et les fonctionnalités des outils de GCL peuvent être mis à profit dans une telle situation.

Bien entendu, si une entreprise utilise plusieurs progiciels, il y a fort à parier que chacun d'eux nécessitera une étude séparée et conduira à des réponses différentes !

Heureusement, dans leur majorité, les outils de GCL du marché sont capables de prendre en charge des situations telles que celles que nous avons décrites ; encore faut-il que l'organisation, dans sa démarche de sélection d'outil de GCL, ait pensé à inclure ce type de scénario dans ses pré-requis. Lors de la phase de déploiement, il faut clairement estimer le coût de la mise en œuvre de la GCL dans chacune de ces situations spécifiques, pour déterminer si le déploiement les intègre immédiatement ou dans une étape ultérieure.

Axe 10 : déploiement par type de processus

Il arrive assez souvent que la mise en place de la GCL s'inscrive dans un contexte plus global d'optimisation des processus internes de la DSI et s'accompagne d'un projet d'optimisation des processus d'assurance qualité ou de mise en production :

- création d'environnements de tests parallèles pour supporter une démarche de développement parallèle ;
- regroupement des demandes d'évolution sous forme de versions applicatives majeures plutôt que prise en compte « au fil de l'eau » ;
- mise en place d'une fonction d'intégration applicative située entre les équipes de développement et les équipes de production pour préparer et supporter les activités de recette utilisateurs ;
- instauration d'environnements de « pré-production » sous le contrôle des équipes de production ;
- réorganisation des CPUs et/ou partitions pour une meilleure sécurité ;
- conduite d'un projet de site de Backup et de Plan de Reprise d'Activité (PRA).

Dans un tel contexte, il est évident que la stratégie de déploiement de la GCL doit être établie pour satisfaire en priorité les objectifs globaux de la DSI.

Il y a là un côté apparemment paradoxal puisqu'on met en place, pour atteindre un objectif à court ou moyen terme, une démarche et des outils qui s'inscrivent dans le long terme. Le paradoxe disparaît si on considère que la GCL – destinée à gérer des changements en toute fiabilité et cohérence – doit elle-même être mise en œuvre dans l'organisation, d'une manière susceptible d'évoluer dans le temps avec les besoins et les objectifs de l'entreprise. Exprimé de manière inverse, on pourrait dire qu'une GCL, mise en œuvre initialement de manière optimale mais qui se révélerait à terme être un facteur d'immobilisme raterait complètement sa cible.

Est-ce un paradoxe ? La GCL n'est-elle pas aussi le levier nécessaire permettant non seulement d'atteindre mais aussi de maintenir sur la durée la satisfaction de ces objectifs de qualité ?

La démarche consiste alors à définir une architecture globale cible de la GCL (définition des environnements, définition des cycles de vie, définition des différents types de rôles, modalités de fabrication des exécutables, procédures d'approbation et de notification, intégration dans le processus de gestion des Demandes, etc.) puis à tracer le contour de ce qui est indispensable pour atteindre les objectifs d'amélioration fixés.

- Il pourra s'agir de mettre en place la GCL sur une partie du cycle de vie (ex : assurance qualité) ;
- ou de mettre sous contrôle de la GCL certaines fonctionnalités (ex : refabrication en masse des exécutables par une cellule d'intégration) ;
- ou de contrôler avec la GCL les composants relevant de telle ou telle technologie de développement.

La mise en place de la GCL se fera dans ce contexte limité mais avec la garantie de pouvoir en étendre le champ ultérieurement sans sortir de la cible prévue.

Constitution de scénarios détaillés de déploiement

On l'aura compris, au risque de frustrer le lecteur, ce document pose autant de questions qu'il apporte de réponses ! Au moins, les questions à se poser sont-elles ici clairement identifiées. Chaque projet de déploiement d'une solution de GCL est particulier. Nous avons donc plutôt cherché à fournir un recensement de toutes les situations à prendre en compte et à détailler les implications de chacune d'elles pour guider la recherche de scénarios de déploiement.

La démarche que nous proposons ensuite consiste à :

(1) examiner les différents axes un par un

Pour chacun d'eux, se poser les questions suivantes :

- notre organisation est-elle concernée par les questions posées relativement à cet axe ? Si ce n'est pas le cas, on ignorera définitivement cet axe dans la suite ;
- peut-on envisager un déploiement exhaustif en une seule fois, sur toutes les rubriques associées à cet axe ?
- en combien d'étapes principales pouvons-nous regrouper ces différentes rubriques et dans quel ordre ?

(2) parmi les axes pour lesquels un déploiement exhaustif est envisageable, en sélectionner deux ou trois, pas davantage

Ces axes combinés avec la première étape retenue sur chacun des autres axes, constitueront le périmètre de la première phase de déploiement.

(3) reprendre les actions (1) et (2) en ignorant les axes et étapes de déploiement déjà pris en compte pour la première phase

On recommence ce travail jusqu'à obtenir les principales phases de déploiement qui constituent un scénario 1.

(4) qualifier en détail, et phase par phase, le scénario 1 en termes de :

- recensement du parc des composants applicatifs concernés ;
- paramétrage, customisation de l'outil de GCL et tests ;
- formation des utilisateurs ;
- migration des composants existants sous le contrôle de la solution de GCL ;
- assistance technique auprès des utilisateurs, administration ;
- planification de ces différentes activités et affectation de ressources.

(5) établir si nécessaire des scénarios alternatifs en reprenant l'étape (3)

Il s'agit simplement de faire glisser certains champs d'application partiels (pas les axes de déploiement majeurs) d'une phase à l'autre.
Dans le choix final du scénario de déploiement, il faut veiller à ce que le calendrier de la première phase ne s'étende pas sur une période trop longue : il vaut mieux en limiter la portée et s'assurer que l'on obtient un résultat rapide et positif plutôt que de viser un objectif trop ambitieux.

Conclusion

Cette étude des stratégies de déploiement illustre un décalage fréquent entre deux types d'horizons :

- la durée prévisible d'utilisation de la GCL dans l'organisation - une fois en place, elle est destinée à rester en fonction pendant une dizaine ou une quinzaine d'années - ;
- et la visibilité sur les besoins auxquels elle doit répondre - on ne connaît pas les technologies qui seront le standard dans l'organisation cinq ans plus tard, ni les projets applicatifs majeurs qui seront lancés dans les deux ans.

Il est donc impératif que la solution de GCL retenue soit flexible et que la mise en œuvre que l'on en fait dans l'organisation soit également flexible ! ▲

jf.castaing@laposte.net