Tendances technologiques

Développement d'applications programmation schématisée

Architecture d'entreprise, Direction générale du dirigeant principal de la technologie

Version 0.1
Date : 2019-7-18

Services partagés Canada
Shared Services Canada
Canada

Table des matières

Sommaire opérationnel

Une *plate-forme de développement p*rogrammation schématisée (ou simplement la programmation schématisée[1]) est un *environnement* et une *plate-forme* de développement logiciel qui permet aux programmeurs modernes de développer des *logiciels d'application* (applications) au moyen d'interfaces utilisateurs graphiques plutôt qu'en rédigeant du code dans un langage de programmation traditionnel. Destiné à l'origine aux applications liées aux bases de données, aux processus opérationnels, aux systèmes de gestion de contenu et de documents et aux interfaces Web, la programmation schématisée est maintenant suffisamment mature pour permettre le développement de presque tous les types d'applications, à l'exception des applications qui sont profondément intégrées ou à très haut rendement (comme les applications de jeu ou de calcul scientifique). La programmation schématisée tire son origine sur le plan technique du *développement rapide d'applications* (rapid application development ou RAD) et des *langages de programmation de la quatrième génération* (de 1992 au début des années 2000).

La programmation schématisée est fondé sur le concept de *hauts niveaux d'abstraction* selon lequel le développeur peut exprimer directement des processus et des exigences opérationnels sans se préoccuper des détails relatifs au codage. Pour certaines applications développées en Low-Code, une petite quantité de code simple est tout de même généralement rédigée manuellement, particulièrement dans le cas des applications qui ne respectent pas l'un des multiples modèles intégrés. (Les applications qui ne requièrent absolument aucun code rédigé manuellement sont ce qu'on appelle les *applications sans code* ou *No-Code*.) La réduction de la nécessité de rédiger du code manuellement a des répercussions importantes :

- Le développement d'applications (y compris la saisie des fonctionnalités, la mise à l'essai, la validation et le déploiement) se déroule beaucoup plus rapidement qu'à l'habitude, permettant ainsi de recourir au *développement selon la méthode Agile*, de réduire les coûts et d'améliorer les délais de mise sur le marché. Cela permet également de réduire le nombre d'erreurs et de mauvaise correspondance avec les exigences.
- Le fait d'avoir moins de code (et moins de code complexe) permet à un plus grand nombre de personnes de prendre part au développement d'applications, qui n'est alors plus limité aux programmeurs très spécialisés (qui sont généralement rares et dont les services sont dispendieux en raison de leurs talents).

[1] Certains analystes et fournisseurs de l'industrie dans le domaine des systèmes de la programmation schématisée utilisent une terminologie très différente, p. ex. la firme Gartner utilise le terme plate-forme comme service d'applications à haute productivité ou *high-productivity application Platform-as-a-Service* (hpaPaaS).

- La programmation schématisée était à l'origine utilisé pour développer des applications à partir de zéro, mais permet également maintenant d'intégrer les systèmes courants ou des systèmes tiers (p. ex., les systèmes de planification des ressources organisationnelles et les bases de données comme SAP, Oracle, DB2 et SQL Server) pour concevoir des applications complètes encore plus rapidement.

La programmation schématisée présente toutefois certains défis :

- Trouver des développeurs : Malgré le fait que les exigences techniques de cette méthode sont plutôt faibles, la plupart des systèmes de programmation schématisée sont des systèmes exclusifs et requièrent au minimum une formation de base spécifique au système en cause.
- Les développeurs traditionnels hautement qualifiés sont souvent sceptiques et sur la défensive en ce qui concerne la programmation schématisée, notamment en raison de la lente dévalorisation des ensembles de compétences traditionnels.
- Les coûts de licences sont souvent peu clairs et généralement plus élevés que les coûts de licences associés aux environnements et outils de développement traditionnels (dont bon nombre sont de source ouverte).

Selon le rapport Forrester Wave de mars 2019, les systèmes de programmation schématisée chefs de file à l'heure actuelle (sur un total de 13 systèmes évalués dans ce rapport) sont les systèmes Microsoft PowerApps, OutSystems et Mendix (maintenant propriété de Siemens), suivis de près par les systèmes Kony et Salesforce. Des classements de fournisseurs semblables sont également établis par les firmes Gartner, Ovum et IDC.

Sommaire technique

En bref, la programmation schématisée est un environnement de développement et une plate-forme d'exécution et permet de concevoir des applications de pratiquement tous les types (à l'exception des applications profondément intégrées ou à très haut rendement) tout en ne rédigeant qu'une petite quantité de code simple. Dans bien des cas, aucun code n'a à être rédigé, ce qui fait de ces applications des applications « sans code ».

Pour le *développeur spécialisé en* programmation schématisée (souvent une personne qui est davantage spécialisée dans le domaine de l'application en cause que dans le codage), l'outil de programmation schématisée est semblable à la plupart des *environnements de développement intégrés*. Plutôt que de saisir du code dans une fenêtre, la fonctionnalité de l'application est conçue visuellement. Le développeur commence habituellement par dessiner les fenêtres de l'application et joint les actions (une fois de plus visuellement) aux divers éléments de l'interface utilisateur graphique comme les boutons, les menus déroulants et les boîtes de listes. Généralement, un bon nombre d'applications appuient un *processus opérationnel* qui est ensuite dessiné schématiquement en programmation schématisée. Chaque étape du processus opérationnel peut activer d'autres fenêtres ou entraîner de plus amples actions.

De telles actions peuvent comprendre la communication avec d'autres applications (comme une application de courriel), l'extraction de documents, la consultation d'une base de données ou une action à distance. La création d'une interaction avec une autre application est généralement une simple connexion visuelle schématique, la plupart des systèmes de programmation schématisée prennent en charge les interfaces de programmation d'applications (qui demeurent uniquement *visuelles* en programmation schématisée), à un grand nombre d'autres applications du fournisseur. De même, l'extraction de documents et de fichiers est effectuée visuellement ainsi que toute action de traitement de ces fichiers, y compris leur partage à distance.

L'un des avantages les plus notables de la programmation schématisée est l'intégration de bases de données. Le développeur n'a qu'à glisser et déposer une base de données, qui est alors généralement *automatiquement découverte* par l'environnement programmation schématisée, rendant ainsi la structure de la base de données visuellement évidente. Le développeur peut ensuite créer visuellement des requêtes de base de données (sous la forme de *langage SQL visuel*) et utiliser ces résultats aux fins de traitement additionnel ou d'affichage dans une fenêtre. Bien entendu, diverses opérations relatives aux bases de données sont également prises en charge. Les principaux fournisseurs de produits programmation schématisée prennent en charge toutes les principales bases de données et leurs différentes versions. Dans un nombre limité de cas, une requête de base de données très complexe doit être rédigée manuellement en langage SQL; ce processus est toutefois encadré par des *assistants intelligents* intégrés à l'outil et effectué visuellement.

La majeure partie du développement réel d'applications nécessite des interactions avec les systèmes existants, ce qui est également pris en charge en développement programmation schématisée. Les systèmes existants (p. ex., SAP, le système de planification des ressources organisationnelles ou la base de données) s'affichent en tant que *connecteurs* dans une palette de systèmes pris en charge, qui peuvent être ensuite utilisés visuellement, alors que le système programmation schématisée gère l'utilisation réelle de l'interface de programmation d'applications.

Tous les systèmes programmation schématisée ont des palettes détaillées de composants additionnels, souvent rédigés par le fournisseur du produit programmation schématisée e en d'autres langages aux fins de performance. Voici certains de ces composants préétablis :

- Services de localisation à l'aide d'un système GPS sur des appareils mobiles.
- Soutien relatif au nuage pour tous les principaux fournisseurs, ainsi qu'aux nuages privés.
- Caméras, y compris la reconnaissance des gestes et faciale.
- Soutien audio, y compris la génération sonore et la reconnaissance ainsi que la synthèse vocale.
- Soutien multilingue.
- Interfaces *Internet des objets* et traitement d'événements complexes.

- Soutien relatif aux fichiers journaux et pistes de vérification pour les applications nécessitant une gouvernance.
- Primitives de sécurité, comme le chiffrement, les signatures et l'authentification.
- Moteurs d'intelligence artificielle et d'apprentissage machine.
- Visualisation pour des données à grande échelle.
- Interfaces dorsales pour les systèmes de données massives comme Hadoop.

Il existe également des communautés d'utilisateurs pour les outils de programmation schématisée qui offrent de nouveaux composants ou des fonctions *wrapper* pour les systèmes existants et les programmeurs expérimentés (dans d'autres langages comme Java, C++ et C#) peuvent facilement produire leurs propres composants au besoin. Les principaux fournisseurs de produits de programmation schématisée offrent un soutien exhaustif pour les applications mobiles et bureautiques, y compris en ce qui concerne les claviers, les stylets et les écrans tactiles. Dans la plupart des cas, ce soutien est intégré au sein d'un même projet, ce qui signifie qu'une version mobile et une version bureautique d'une application peuvent être développées de façon simultanée.

Les environnements de programmation schématisée prennent directement en charge le contrôle des versions au moyen de différentes interfaces comme Git, Mercurial et Subversion. Cela offre non seulement la possibilité d'effectuer des retours en arrière relativement aux changements, mais également de réaliser des projets avec de multiples développeurs.

À la suite de l'assemblage de l'application, l'environnement de programmation schématisée permet immédiatement la mise à l'essai ou le débogage de celle-ci sans devoir passer par un long cycle de compilation et de conception. Cela permet d'appuyer directement le développement Agile et le perfectionnement conjoint des exigences lors desquels les spécialistes et les utilisateurs du domaine peuvent apporter des ajustements à l'application rapidement. Ces essais peuvent être réalisés par exemple sur une plate-forme de mise à l'essai conçue à cette fin avec des bases de données d'essai.

Enfin, le déploiement correspond généralement à un processus d'une seule étape au sein du *gestionnaire du cycle de vie* de l'application lors duquel l'environnement de programmation schématisée est conscient des paramètres de la plate-forme de déploiement et l'application peut être lancée directement en production et utilisée. Si quelque chose ne se déroule pas comme prévu, le retour en arrière à une version antérieure est généralement simple à exécuter.

Bien entendu, la programmation schématisée utilise également les langages de programmation et les outils traditionnels, quoique cela ne soit généralement pas visible pour le développeur. La programmation visuelle est directement transposée en langage C#, Java ou en langage semblable (certaines plates-formes comme OutSystems prennent en charge les deux) qui est ensuite compilé à l'aide de la chaîne d'outils de développement normale. L'environnement de la programmation schématisée assure le suivi de la quantité minimale de code généré et recompile les éléments, permettant ainsi

une expérience de développement très interactive axée sur l'expérimentation et l'établissement de prototypes. Les extrants C# ou Java peuvent également être utilisés de façon autonome si l'on délaisse la programmation schématisée, bien que le code en soi ne soit souvent pas suffisamment élégant pour permettre une compréhensibilité réelle.

Utilisation par l'industrie

Tous les principaux analystes techniques de l'industrie (Forrester, Gartner, Ovum et IDC) produisent des rapports sur la programmation schématisée depuis un certain nombre d'années, à la suite de la désignation de cette approche par le terme « programmation schématisée» par Gartner en 2014. Ces rapports ont parfois relevé jusqu'à cinquante fournisseurs d'environnements de programmation schématisée au cours de la dernière décennie. Certains de ces fournisseurs ne sont plus actifs alors que trois sont devenus les chefs de file des fournisseurs de solutions de programmation schématisée très générales[2] : Mendix (qui fait maintenant partie de Siemens), Microsoft (avec PowerApps) et OutSystems. Le diagramme Forrester Wave de mars 2019 est illustré à la Figure 1, alors que le diagramme Gartner Magic Quadrant de 2018 est illustré à la Figure 2.

[2] Une solution *générale* est une solution qui est en mesure d'interagir avec bon nombre d'autres types de composants (composants existants et composants rédigés dans un autre langage de programmation) et qui peut être exécutée sur plusieurs plates-formes et produire des applications de tous types.

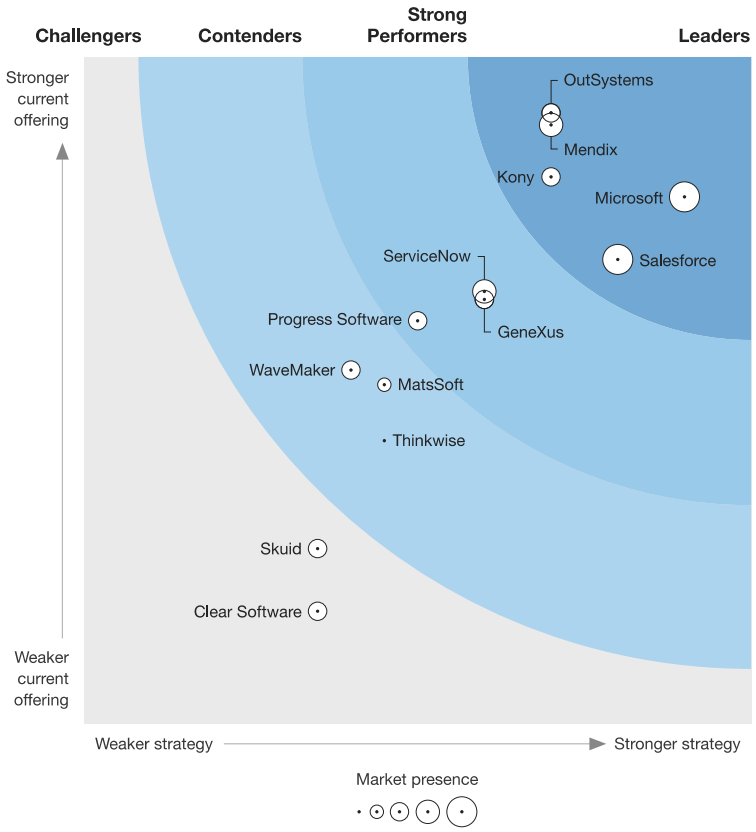

Figure 1 Diagramme Forrester Wave : Plates-formes de développement en programmation schématisée pour les professionnels du développement et de la livraison d'applications, T1 2019. Utilisé sans autorisation.

English	Français
Challengers	Nouveaux fournisseurs
Contenders	Fournisseurs concurrents
Strong Performers	Fournisseurs performants
Leaders	Chefs de file
Stronger current offering	Offres actuelles les plus intéressantes
Weaker current offering	Offres actuelles les moins intéressantes
Weaker strategy	Stratégie plus faible
Stronger strategy	Stratégie plus solide
Market presence	Présence sur le marché
OutSystems	OutSystems
Mendix	Mendix
Kony	Kony
Microsoft	Microsoft
Salesforce	Salesforce
ServiceNow	ServiceNow
GeneXus	GeneXus
Progress Software	Progress Software
WaveMaker	WaveMaker
MatsSoft	MatsSoft

Thinkwise	Thinkwise
Skuid	Skuid
Clear Software	Clear Software

Parmi les différentes exigences, les systèmes figurant dans le diagramme Forrester Wave ont été notamment sélectionnés puisque chacun d'entre eux :

1. Offre une approche de développement déclaratif exhaustive : le niveau d'abstraction correspond à celui du client, ce qui est essentiel pour exprimer les exigences.
2. Offre un modèle commercial à faible coût : les fournisseurs offrent des périodes d'essai sans frais et du matériel de formation en ligne.
3. Prend en charge l'établissement de nombreux cas d'utilisation opérationnelle, relatifs aux applications Web et mobiles, aux bases de données, au traitement d'événements, à l'Internet des objets et aux applications de processus opérationnels.
4. Cible principalement les grandes entreprises : revenus de plus d'un milliard de dollars américains et équipes dispersées géographiquement.

Les sites Web des fournisseurs d'outils de programmation schématisée chefs de file présentent généralement des clients en référence et ceux-ci comprennent bon nombre de banques, de compagnies d'assurance et de compagnies aériennes de renom ainsi que des ministères gouvernementaux et l'Armée américaine. Toutefois, dans la plupart des cas, aucun renseignement n'est fourni en ce qui concerne les domaines d'application en particulier. Selon la plupart des rapports produits par les analystes et les rapports produits par OutSystems et Mendix, 88 % des entreprises adoptent la programmation schématisée et 74 % de ces entreprises procèdent à l'intégration du volet affaires au développement de programmation schématisée, incluant ainsi la participation directe des clients qui dictent les exigences[3].

[3] Pour obtenir un exemple, consultez le site suivant : https://www.mendix.com/why-developers-should-embrace-low-code/ (en anglais seulement)

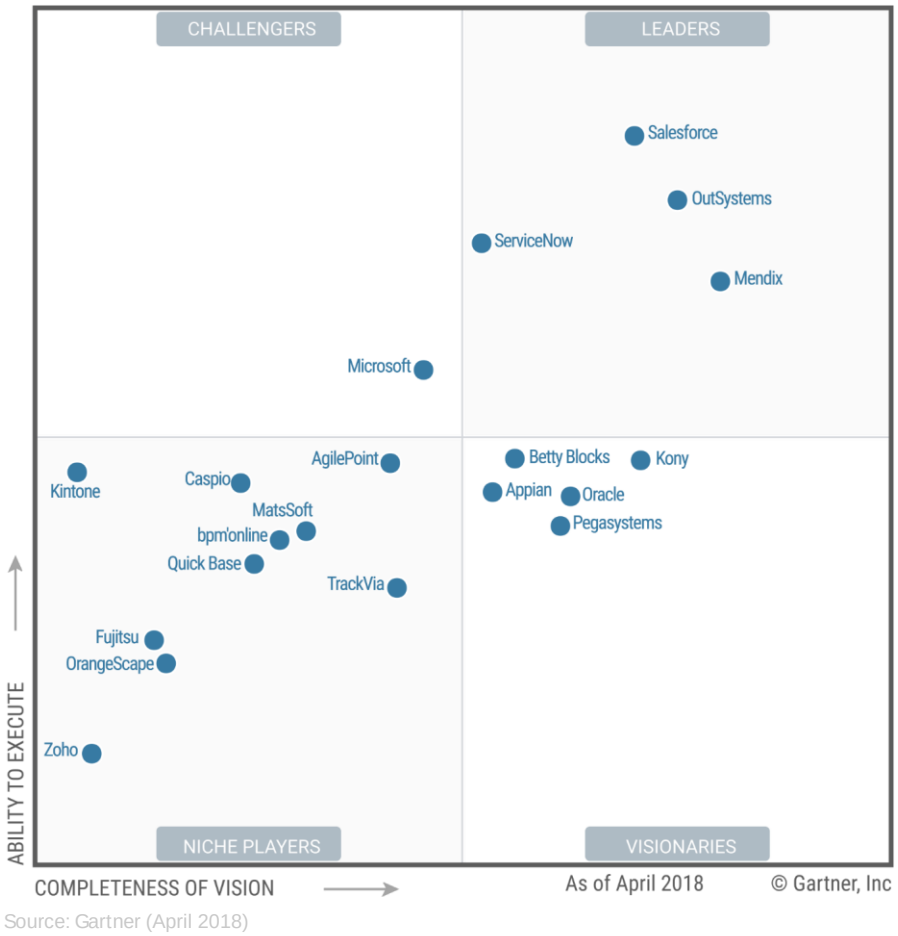

Figure 2 : Magic Quadrant de Gartner – Plate-forme d'application d'entreprise comme service à haute productivité. Utilisé sans autorisation.

English	Français
CHALLENGERS	NOUVEAUX FOURNISSEURS
LEADERS	CHEFS DE FILE
ABILITY TO EXECUTE	CAPACITÉ D'EXÉCUTION
COMPLETENESS OF VISION	EXHAUSTIVITÉ DE LA VISION
NICHE PLAYERS	FOURNISSEURS DE NICHE
VISIONARIES	VISIONNAIRES
As of April 2018	En date d'avril 2018
Source: Gartner (April 2018)	Source : Gartner (avril 2018)
Salesforce	Salesforce
OutSystems	OutSystems
Mendix	Mendix
ServiceNow	ServiceNow
Microsoft	Microsoft
Kintone	Kintone
Caspio	Caspio
AgilePoint	AgilePoint
MatsSoft	MatsSoft

Bpm'online	Bpm'online
Quick Base	Quick Base
TrackVia	TrackVia
Fujitsu	Fujitsu
OrangeScape	OrangeScape
Zoho	Zoho
Betty Blocks	Betty Blocks
Kony	Kony
Appian	Appian
Oracle	Oracle
Pegasystems	Pegasystems

Utilisation par le gouvernement du Canada

La programmation schématisée moderne peut être utilisé littéralement dans *tous les domaines d'applications* à l'exception des systèmes qui sont profondément intégrés ou à très haut rendement. Toutefois, même pour ces systèmes, la programmation schématisée peut fournir la majeure partie d'une application tout en assurant des interactions avec des composants logiciels rédigés dans d'autres langages de programmation.

Compte tenu de cette portée, littéralement *tous* les domaines d'applications de technologie de l'information (TI) du gouvernement du Canada sont des candidats potentiels pour la programmation schématisée. Par exemple :

- Applications composées d'un processus opérationnel, incluant les interfaces relatives à la gestion de contenu et de documents, aux courriels de rappels, etc.
- Applications de collecte de données et de base de données ayant une interface avec des dépôts de stockage de données à grande échelle ou répartis, incluant la gestion de l'identité et la protection des renseignements personnels.
- Applications de communication et de gestion des tâches permettant des fonctions de messagerie instantanée, de courriel, de vidéoconférence et de gestion de calendrier.
- Applications de science des données et de visualisation des données pouvant possiblement contenir des fonctions d'apprentissage machine ou d'intelligence artificielle pour traiter et présenter d'importantes quantités de données.

Les groupes d'analystes de la firme IDG et de la société OutSystems ont plus particulièrement examiné la solution OutSystems pour le gouvernement numérique en fondant leur évaluation sur les objectifs présentés dans la Figure 3. Les conclusions de cette évaluation sont toutefois valides pour tous les systèmes de programmation schématisée. Plus particulièrement, les systèmes de Programmation Schématisée peuvent permettre d'accroître la qualité tout en diminuant les coûts associés à la prestation des services.

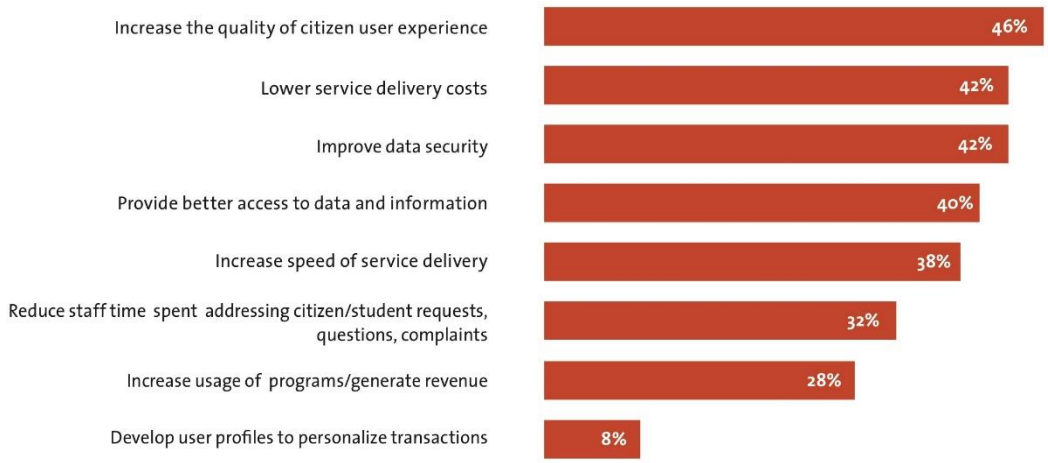

Figure 3 : Principaux objectifs en ce qui concerne l'expérience numérique offerte aux citoyens. Utilisé sans autorisation. Tiré de : Improving Digital Experience for End Users in the Public Sector (Améliorer l'expérience numérique pour les utilisateurs finaux du secteur public), IDG Research Services, décembre 2018. Utilisé sans autorisation.

English	Français
Increase the quality of citizen user experience	Accroître la qualité de l'expérience utilisateur du citoyen
Lower service delivery costs	Diminuer les coûts de la prestation des services
Improve data security	Améliorer la sécurité des données
Provide better access to data and information	Offrir un meilleur accès aux données et à l'information
Increase speed of service delivery	Accroître la rapidité de la prestation des services
Reduce staff time spent addressing citizen/student requests, questions, complaints	Réduire le temps consacré par le personnel à la résolution des demandes de renseignements, des questions et des plaintes des citoyens et des étudiants
Increase usage or programs/generate revenue	Accroître l'utilisation des programmes et générer des revenus
Develop user profiles to personalize transactions	Créer des profils d'utilisateurs pour personnaliser les transactions

Il convient de noter que l'atteinte de ces objectifs (au moyen par exemple de solutions de programmation schématisée) cadre avec les niveaux supérieurs du modèle Digital Government Maturity Model (modèle de maturité relatif au gouvernement numérique) de Gartner, comme le niveau 4 et le niveau 5.

Répercussions pour Services partagés Canada (SPC)

Proposition de valeur

Compte tenu de la participation importante de SPC en ce qui concerne les services de TI et la gestion de la TI, la proposition de valeur relative à la programmation schématisée est axée sur la production, le déploiement et la maintenance des applications. La programmation schématisée permet au développement des applications d'atteindre des niveaux d'abstraction plus élevés (p. ex., la modélisation directe des processus opérationnels) et nécessite la rédaction d'une moins grande quantité de code. Voici les propositions de valeur correspondantes :

1. Délais de mise sur le marché plus courts, ce qui fait en sorte d'obtenir un taux de satisfaction de la clientèle plus élevé et des calendriers plus exacts et permet d'éviter d'être tentés de concevoir des solutions de rechange ou d'acquérir des solutions temporaires mal adaptées.
2. Moins grand nombre d'heures-personnes requises pour le développement, faisant en sorte de diminuer les coûts.
3. Élargissement du bassin de développeurs d'applications potentiels permettant de diminuer la pression sur le personnel et possiblement de diminuer les coûts tout en raccourcissant les délais associés aux activités de développement.
4. Intégration directe des systèmes existants permettant le déploiement graduel et l'utilisation de solutions programmation schématisée pour la gestion des coûts et des risques.
5. Saisie plus exacte des exigences des clients et conception mieux adaptée à celles-ci permettant de diminuer les coûts associés aux activités de remaniement et de maintenance.

Difficultés

Alors que les fournisseurs des solutions programmation schématisée présentent cette approche comme étant la solution à *tous les projets d'applications*, cette approche n'est pas sans difficultés ou problèmes, qui sont tous pertinents pour SPC :

- La nature de la programmation schématisée consiste à permettre à des personnes qui ne sont pas des programmeurs (idéalement des spécialistes du domaine du client en cause) de créer des applications. Toutefois, les environnements de la programmation schématisée sont en grande partie exclusifs et requièrent au minimum une formation de base. Cela s'applique également aux développeurs hautement spécialisés.
- La plupart des organisations observent une certaine résistance chez les développeurs traditionnels qui est partiellement attribuable à un scepticisme et qui découle souvent d'une attitude défensive par rapport au fait qu'un plus grand nombre de personnes deviennent des « développeurs » (et à moindre coût).

- La plupart des environnements de développement traditionnels nécessitent l'achat unique de l'environnement de développement intégré. Les licences pour les solutions de programmation schématisée sont toutefois beaucoup plus complexes et comprennent souvent des coûts par application déployée. Ces coûts sont parfois peu clairs et doivent être examinés en détail aux fins d'analyse du rendement du capital investi.
- Il existe plusieurs systèmes de programmation schématisée parmi lesquels faire un choix. Les chefs de file actuels, qui comprennent Mendix, Microsoft et OutSystems, sont toutefois constamment en tête du peloton depuis un bon moment[4]. Le choix des plates-formes est complexe et dépend largement du logiciel existant qui doit être pris en charge, des plates-formes de déploiement existantes, etc. La prise en charge de plus d'une plate-forme de programmation schématisée entraînera ses propres défis.
- La nature exclusive des solutions de programmation schématisée fait en sorte que SPC pourrait ne pas être en mesure de changer de fournisseur en raison d'un contrat exclusif ou être contraint de continuer à utiliser des solutions de programmation schématisée. Les fournisseurs présentent généralement le développement de programmation schématisée e comme n'étant pas assujetti à l'exclusivité contractuelle en raison du fait que le code généré est en langage C# ou Java et peut être mis à jour par la suite sans recourir à la solution de programmation schématisée. En pratique, cela n'est pas vrai pour les applications de grande taille.
- La facilité (ou la « démocratisation ») du développement d'applications avec la programmation schématisée peut causer des problèmes pour SPC : les clients seront tentés de développer eux-mêmes les applications si SPC n'est pas assez rapide, créant ainsi une mentalité de *TI utilisée dans l'ombre (masquée/travaux autonomes)*. Si cette tendance croit, cela pourrait entraîner des problèmes relatifs à la maintenance et remettre en question l'existence de SPC.
- La sécurité est le maillon faible chronique des solutions de programmation schématisée et les fournisseurs ne s'y attardent à l'heure actuelle que du point de vue structurel. Les primitives de sécurité sont depuis longtemps disponibles dans les palettes de développement de la programmation schématisée, mais leur utilisation n'est pas obligatoire et elles ne sont pas imbriquées à l'environnement.

Considérations

SPC doit prendre en considération de nombreux éléments afin de déterminer si le Ministère adopte ou non la programmation schématisée et de quelle façon et à quel moment il fera la transition le cas échéant. SPC tirerait bien évidemment profit des améliorations de l'efficacité que permet la programmation schématisée et les considérations principales visent à déterminer de quelle façon et à quel moment

[4] Ce n'est pas tout à fait vrai pour Microsoft, qui est relativement un nouveau joueur dans le domaine des solutions de programmation schématisée, bien que ses environnements de développement et ses plates-formes générales soient très matures et stratégiques.

adopter la programmation schématisée et dans quelle mesure cette approche doit être déployée. Lorsque les clients prendront connaissance des plans de SPC relatifs à la programmation schématisée, ils exerceront de plus en plus de pression pour raccourcir les échéanciers et les calendriers et diminuer les coûts associés aux projets. SPC devra faire preuve de plus de souplesse dans les interactions avec les clients ainsi que solliciter une participation plus directe de leur part dans le cadre des projets fondés sur la programmation schématisée, tout en maintenant le contrôle direct du développement des applications. Peu importe la rapidité avec laquelle SPC adoptera la programmation schématisée, il existe un risque que cette approche devienne populaire auprès de programmation schématisée de type *TI dans l'ombre*. Il sera difficile d'éviter complètement que de tels projets aient lieu et SPC doit examiner de quelle façon il pourra reprendre en main ceux-ci afin de les intégrer au sein de son organisation et offrir un soutien pour assurer leur bon déroulement.

D'autres éléments doivent également être pris en considération en ce qui concerne le déploiement de la programmation schématisée au sein de SPC. Malgré sa simplicité, de la programmation schématisée nécessitera au minimum une formation de base pour les développeurs. La formation devra être jumelée à des initiatives de gestion du changement afin de rallier et non de s'aliéner les développeurs traditionnels en vue de la transition; le défaut d'adopter une telle approche pourrait faire en sorte de diviser l'effectif de développeurs en deux catégories ou que les développeurs traditionnels travaillent à contre-courant des programmeurs en programmation schématisée. SPC devra miser sur une séquence de projets qui est appropriée pour la programmation schématisée, p. ex., une séquence axée sur la gestion du risque et la criticité ou fondée sur les économies de coûts potentielles. Les besoins en matière de sécurité de SPC sont en quelque sorte unique (en raison du large éventail de clients) et il sera important de s'assurer que la programmation schématisée permet d'appuyer adéquatement la sécurité. Cette question devra être examinée de façon plus approfondie.

SPC devrait considérer la tenue d'un essai pour la programmation schématisée pour différentes raisons, notamment pour mieux connaître l'approche, évaluer les outils connexes et quantifier le rendement du capital investi (au moins pour un projet). Idéalement, tous les systèmes de programmation schématisée chefs de file devraient être évalués, mais l'échantillon pourrait être restreint (p. ex., ne retenir que Microsoft PowerApps ou OutSystems) en tenant compte de la portée du système ainsi que de la prise en charge par celui-ci des technologies existantes utilisées au sein de SPC. Plusieurs aspects devraient être mesurés pendant l'essai, notamment les *coûts associés à la formation* (qui seront ensuite *amortis* puisqu'ils seront répartis sur plusieurs projets), les coûts de licences amortis, le temps écoulé, le taux de satisfaction de la clientèle et les coûts de maintenance amortis.

Références

- *The Forrester Wave: Low-Code Development Platforms for AD&D Professionals, Q1 2019*, Forrester. https://reprints.forrester.com/#/assets/2/160/RES144387/reports (en anglais seulement)
- *Magic Quadrant for Enterprise High-Productivity Application Platform as a Service*, Gartner. https://www.gartner.com/doc/reprints?id=1-4XVPI4N&ct=180430&st=sb (en anglais seulement)
- Ovum Decision Matrix: Selecting an Enterprise Mobile Application Development Platform, Ovum, 13 avril 2018. (en anglais seulement)
- *What Is Low-Code*, OutSystems. https://www.outsystems.com/blog/what-is-low-code.html (en anglais seulement)
- *App dev has evolved: Why developers need to embrace Low-Code*, Mendix. https://www.mendix.com/why-developers-should-embrace-low-code/ (en anglais seulement)
- *Low-Code Guide*, Appian. https://www.appian.com/assets/sites/14/2016/12/low-code-guide.pdf (en anglais seulement)
- *Low-code development platform* : https://en.wikipedia.org/wiki/Low-code_development_platform (en anglais seulement)
- *App dev has evolved*, Mendix. https://www.mendix.com/why-developers-should-embrace-low-code/ (en anglais seulement)
- Accelerating Digital Government With Low-Code Development, IDG parrainé par OutSystems, décembre 2018.
- *Introducing the Gartner Digital Government Maturity Model 2.0*, Gartner. https://www.gartner.com/doc/reprints?id=1-667LEM1&ct=190130&st=sb (en anglais seulement)
- *The Best Low-Code Development Platforms for 2019* : https://www.pcmag.com/roundup/353252/the-best-low-code-development-platforms (en anglais seulement)