Cigref
RÉUSSIR
LE NUMÉRIQUE

Nouvelles pratiques de développement Low Code / No Code

Libérer la valeur en maîtrisant les risques

Décembre 2022

Cigref

Nouvelles pratiques de développement Low Code / No Code

Libérer la valeur en maîtrisant les risques

Décembre 2022

SYNTHÈSE

L'utilisation des plateformes « *Low code / No code* » progresse depuis quelques années. Ne nécessitant *a priori* aucune connaissance en développement informatique, celles-ci questionnent les fonctions des DSI ainsi que leurs liens avec les métiers. Ces plateformes donnent-elles aux métiers de l'entreprise la possibilité de se passer de l'expertise technique des DSI pour développer des applications et transformer leurs propres processus ? Ou faut-il y voir une opportunité pour les DSI de travailler en collaboration avec les métiers, en leur apportant des solutions propices à leur développement avec plus d'autonomie et en limitant ainsi le *shadow IT* (c'est-à-dire le développement de solutions numériques par les métiers sans contrôle de la DSI) ?

Le constat partagé au fil des réflexions menées dans le cadre du groupe de travail Cigref est que ces approches démocratisent le développement informatique et introduisent une notion de « *citizen* » développeur (celui qui n'est pas un expert IT). En effet, la prise en main de telles plateformes peut se faire sans connaissance approfondie du développement informatique ce qui accroît l'autonomie et l'agilité des métiers et apporte ainsi des réponses au besoin d'un *time-to-market* rapide. Ces plateformes *Low code* ou *No code* peuvent aussi répondre aux besoins des DSI elles-mêmes sur certains types d'activité, leur permettant d'augmenter la productivité des développeurs « professionnels » et ponctuellement de pallier le déficit de compétences de plus en plus prégnant dans le développement informatique. Cependant, pour tirer la véritable valeur de ces nouvelles pratiques *Low code / No code*, il est primordial de définir un cadre en partenariat avec les métiers. Pour cela, il est nécessaire de mettre en place une gouvernance pour suivre et administrer l'usage de ces plateformes, de définir les critères d'éligibilité et les bonnes pratiques d'usage auprès des "*citizen developers*", d'adapter les architectures techniques mais aussi de mettre en place un centre d'expertise dédié aux pratiques *Low code / No code* au sein des équipes IT.

Voici les enseignements majeurs tirés des réflexions du groupe de travail :

- **Les approches *Low code / No code* sont de véritables opportunités, y compris pour la DSI elle-même** : dans un contexte de contraintes financières, elles peuvent permettre de faire du développement à faible coût et de poser un cadre de fonctionnement formel avec les métiers, ce qui est un manque, par définition, dans les pratiques *shadow IT*.
- **Les règles pour bénéficier des meilleures conditions de réussite consistent à standardiser ces pratiques dans une dynamique de créativité avec les métiers** : avec par exemple l'élaboration d'un arbre de décision qui structure la démarche et permet une analyse des critères d'éligibilité des plateformes, ou encore avec un fort accompagnement du métier autour d'enjeux de sécurité et de qualité du code, qui soit adapté aux populations cibles (*citizen developers* ou métiers ayant une expertise de développement).
- **Les questions d'urbanisme se posent également dans ce cadre** : la DSI a une responsabilité partagée avec les métiers en choisissant d'ouvrir les APIs et les bases de données de manière sécurisée et optimisée.

Suivant les stratégies adoptées, les plateformes *Low code / No code* pourront se révéler être une opportunité de performance business et d'innovation, mais aussi être le vecteur d'une transformation culturelle des organisations.

REMERCIEMENTS

Nos remerciements vont à **Claire WAAST-RICHARD**, Directrice Data et Numérique à **ENEDIS**, qui a piloté cette réflexion, ainsi qu'à toutes les personnes qui ont participé et contribué à ce groupe de travail Cigref :

Christophe AUBIGNAC– RENAULT
Philippe AUBOURG – BNP PARIBAS
Laurent AVET – BNP PARIBAS
Emmanuel BATT – Groupe SEB
Grégory BERTHELOT – MAIF
Thierry BEY – STELLANTIS
Éric BLAWAT – CEA
Jérôme BOULLOT – HAGER GROUP
Claude BOUQUET – ENEDIS
Mohamed Ikbal BOUZGAROU – Air France KLM
Rémi BUISSON – FONDATION DE FRANCE
Sylvain CIONNEAU – SAVENCIA
Jérémy D'AURIA – ENEDIS
Yann DAUDIN – PÔLE EMPLOI
Jérôme DE GALLÉ – AXA
Joël DIEHI – MALAKOFF HUMANIS
My-Phuong DULMAN – BANQUE DE FRANCE
Stanislas DUTHIER – GROUPE ROCHER
Mariam EL-FANIDI – EDF
Pierre-Xavier FOUILLÉ– NAVAL GROUP
Vincent GALINIER – AIRBUS
Matthieu GARCIA – Laboratoires PIERRE FABRE
Jeremy GIBBONS – AIR LIQUIDE
Frédéric GIGAN – PÔLE EMPLOI
Guillaume GIROU – Groupe BPCE
Christophe GORGEOT – GPT LES MOUSQUETAIRES
Stéphane GRAS – AIR FRANCE KLM
Abdelnour GUEMACHE – SCOR
Stéphane JACQUOT – SOCOTEC
Jean-Christophe LALANNE – AIR FRANCE KLM
Hervé LAGASSIE – GROUPE BPCE
Laurent LE SAOUT – ENEDIS
Olivier LÉAL – BANQUE DE FRANCE
Stéphane LOPEZ – iMSA
Dominique MEUNIER – MAIF
Amancio MONTEIRO – MALAKOFF HUMANIS
TuChu NGO – EDENRED
Sébastien PEANNE – SCOR
Cédric PETIT – EDF
Frédéric RIGA – GETLINK
Marc-Michel STACK – BNP PARIBAS
Tristan TARBÉ – GROUPE SAVENCIA
Caroline TRAMALLONI – AIR LIQUIDE
Francis VALLET – AIR FRANCE KLM
Laurent VERHOEST – TRANSDEV GROUP
Samia VERMOET – PÔLE EMPLOI
Julien VERRIER – SCOR
Karen WILLMS – Groupe SEB
Chris WOODROW – MAIF

Ce document a été construit et rédigé par **Flora FISCHER**, Directrice de mission au Cigref, avec la contribution de la pilote et des participants aux travaux.

Nous remercions également vivement tous les intervenants qui ont apporté de la matière à notre réflexion :

- **Alain FAURÉ** et **Sylvain FAGNENT**, Manager, référent sur le No/Low Code à **OCTO TECHNOLOGY**
- **Christophe GORGEOT**, Responsable Domaine DEV FACTORY à la **STIME** (**GROUPEMENT LES MOUSQUETAIRES**)
- **Éric BLAWAT**, Chef de projet MOE au **CEA**
- **Matthieu GARCIA**, Architecte d'Entreprise chez **PIERRE FABRE**
- **Benoit LOCU**, Head of Enedis Innovation Labs chez **ENEDIS**
- **Caroline TRAMALLONI**, Group CIO Office, Enterprise Architect chez **AIR LIQUIDE**
- **Jean-Pierre PITRAU**, IT4IT and FastTrack Dev Manager et **Laurent LETELLIER**, IT Product Manager for Low Code / No Code solutions chez **AIR FRANCE KLM**

TABLE DES MATIÈRES

TABLE DES FIGURES

TABLE DES ENCARTS

INTRODUCTION

Si les plateformes *Low code* et *No code* ne sont pas une nouveauté dans les grandes organisations, leur attractivité auprès des équipes ne cesse de grandir ces dernières années, notamment car certaines d'entre-elles sont proposées « gratuitement » et parfois sans contrainte d'accès avec des offres en *bundle*. Le marché des offres se développe et les couvertures fonctionnelles sont de plus en plus complètes allant de la mise en place de *workflow* jusqu'au traitement de données en passant par le développement d'applications ou d'applications web. Avec peu de code ("*low code*") ou pas de code ("*no code*") ces offres séduisent de plus en plus les métiers qui les utilisent pour répondre à un besoin spécifique d'autonomie ou de rapidité d'exécution. Elles séduisent aussi des développeurs « professionnels » qui, grâce à l'automatisation ou l'abstraction des tâches purement techniques que permettent notamment les plateformes *Low code,* vont pouvoir se concentrer davantage sur l'aspect fonctionnel des applications. Entre souplesse, rapidité de développement et de mise en production et un coût qui (en première approximation) parait réduit, ces plateformes semblent offrir une réponse à l'engorgement des demandes de conception de petites applications, auquel font face de nombreuses organisations, faute de temps et de moyens.

Force est de constater, cependant, que ces plateformes, malgré toutes les opportunités qu'elles promettent, viennent parfois renforcer le « *shadow IT* », c'est-à-dire le développement de solutions numériques sans implication de la DSI, ce qui est à la fois problématique sur le plan de la gouvernance et risqué en termes de sécurité des systèmes informatiques. Ces approches *shadow IT* poseront immanquablement des problèmes d'exploitabilité, de maintenabilité, de sécurité, de redondance applicative, et même de coût sur des applications métiers « simples » développées en *Low code / No code*. Ces dernières peuvent devenir très complexes voire critiques au fil du temps, et *in fine* coûter plus cher qu'une application développée plus classiquement.

Le *shadow IT* est une réalité que le *Low code / No code* peut en effet accentuer s'il n'est pas encadré. La mise en place d'une gouvernance claire et d'un accompagnement au plus près des métiers sont des étapes qui paraissent incontournables pour réussir un déploiement maitrisé de ces plateformes, d'autant que selon une étude de Gartner, « *il y aura quatre fois plus de citizen developers que de développeurs professionnels en 2023 […] et les professionnels (de l'IT) ne seront plus assez nombreux pour répondre aux besoins des entreprises*. »[1]

Les DSI ont pris conscience de ces éléments. Mais, en pratique, comment partager avec les métiers les **biais ou les zones d'ombre des pratiques *Low code / No code*** qui ne nécessitent, à première vue, aucune connaissance en développement ? **Comment organiser, animer et accompagner** le déploiement de ces pratiques dans les métiers **en garantissant le bon niveau de sécurité et un support technique optimum** ?

Dans un contexte où la demande de conception d'applications en interne ne cesse de croître, la démocratisation de ces usages *Low code* ou *No code* serait-elle une réponse à la transformation numérique des métiers ?

[1] https://www.lemagit.fr/actualites/252490733/Low-code-no-code-les-geants-du-cloud-bousculent-les-lignes-du-marche

1 CONSTATS

1.1 POSITIONNEMENT DES DSI VIS-À-VIS DE L'APPROCHE LOW CODE / NO CODE

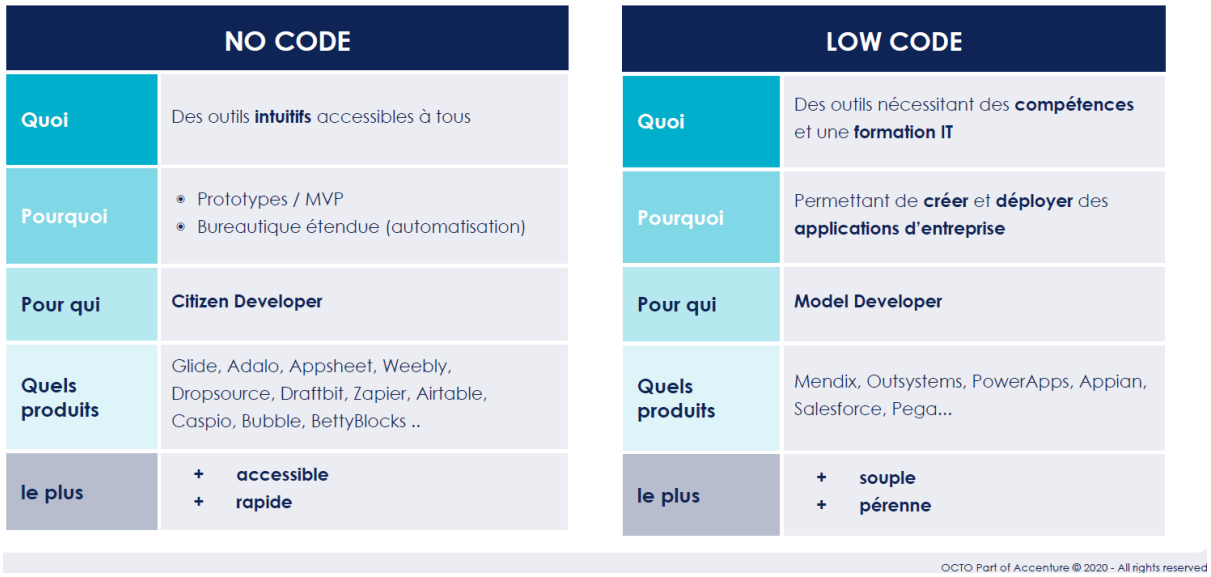
De plus en plus de DSI se questionnent sur la conduite à tenir vis-à-vis des plateformes *Low code / No code*. Faut-il bloquer leur accès le temps d'étudier les meilleures façons de faire pour éviter le *shadow IT* et les risques associés ? Ou faut-il expérimenter et dans ce cas s'interroger sur : comment définir les usages, comment les segmenter, comment accompagner les métiers, quelles plateformes choisir, comment suivre dans la durée les développements des métiers ? De nombreux DSI soutiennent qu'il n'est pas tant question ici de contrôler et restreindre les usages que de savoir les orchestrer pour en tirer la valeur et en maîtriser les risques.

1.2 DÉFINITIONS ET PREMIÈRES APPROCHES

La différence majeure entre le **développement classique** et les **approches *Low code* ou *No code*** repose sur la mise à disposition par les plateformes *Low code / No code* de composants unitaires de haut niveau, prêts à l'emploi, que le *citizen developer* assemble selon les fonctionnalités qu'il veut mettre en œuvre. L'approche est très fonctionnelle et la complexité technique du développement est cachée dans les composants réutilisables. Ces composants sont donc accessibles à des non spécialistes, ce qui limite les coûts, baisse le niveau de technicité requis pour développer une application, contribue à l'agilité métier et répond au manque de développeurs grâce à d'autres profils (qu'il faudra cependant former pour que les applications restent simples). Le développement classique quant à lui, nécessite d'écrire du code explicitement. Il permet d'appréhender des besoins complexes mais aussi d'optimiser le code pour qu'il soit, par exemple, plus efficace dans sa consommation de disques ou de mémoire (et donc d'énergie), plus facilement exploitable ou maintenable dans la durée, y compris quand il faut faire face à des failles de sécurité.

La promesse du ***No code*** est de fournir des interfaces intuitives accessibles à toute personne sans connaissance du code. Le *No code* est particulièrement adapté pour faire du prototypage, du MVP (*minimum viable product*), car il permet de concevoir rapidement de petites applications, pour créer par exemple des pages web très simples, intégrer des données de manière dynamique ou des applications externes via des APIs ou encore de publier rapidement sur un site web ou une application.

Le ***Low code*** est bien différent. Il permet de concevoir des applications d'entreprise qui ont un objectif de pérennité. Ses fonctionnalités sont plus riches et complexes, mais non critiques pour l'entreprise. Elles permettent par exemple la construction de pages ou la configuration de *workflows*, l'intégration des applications au SI de l'entreprise, la gestion de configuration d'environnement. Le *Low code* nécessite beaucoup plus de compétences et de formation IT, même si la frontière *Low code / No code* s'estompe car les outils *No code* proposent des options de conception de plus en plus poussées.



NO CODE	
Quoi	Des outils **intuitifs** accessibles à tous
Pourquoi	• Prototypes / MVP • Bureautique étendue (automatisation)
Pour qui	**Citizen Developer**
Quels produits	Glide, Adalo, Appsheet, Weebly, Dropsource, Draftbit, Zapier, Airtable, Caspio, Bubble, BettyBlocks ..
le plus	+ **accessible** + **rapide**

LOW CODE	
Quoi	Des outils nécessitant des **compétences** et une **formation IT**
Pourquoi	Permettant de **créer** et **déployer** des **applications d'entreprise**
Pour qui	**Model Developer**
Quels produits	Mendix, Outsystems, PowerApps, Appian, Salesforce, Pega...
le plus	+ **souple** + **pérenne**

Figure 1 : Octo Technology, « Deux types d'outils pour des usages et des utilisateurs-développeurs différents », intervention au groupe de travail Cigref "Low code, No code", Janvier 2022

Il est important de souligner que **plus les outils sont simples, plus ils vont contraindre les usages** et les types d'applications possibles. À partir de ce constat, il est donc essentiel de savoir adresser les solutions *Low code / No code* aux populations adéquates, selon leurs différents besoins :

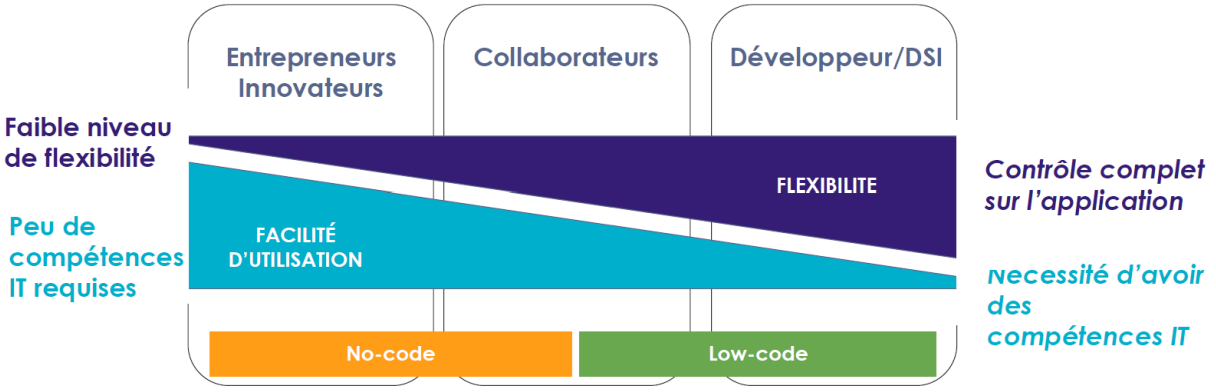


Figure 2 : Octo Technology, « Des compromis différents pour des populations différentes », présentation dans le cadre du GT Cigref « Low code, No code », Janvier 2022.

Selon cette modélisation d'Octo Technology, trois types de populations sont concernées par ces solutions : L'entrepreneur/ innovateur, les collaborateurs et les développeurs et équipes de la DSI.

L'entrepreneur - innovateur souhaite par exemple créer un **prototype** pour **valider des hypothèses sur un produit**, développer un MVP pour **débuter une activité, ou encore créer une application pour un événement spécifique**. Avec le *No code,* ce profil d'entrepreneur/innovateur disposera d'outils simples à manier, focalisés sur un problème spécifique (site web ou mobile, gestion de données), avec la possibilité d'assembler différentes briques par spécialité. Certes, cet usage donne peu de visibilité sur le modèle et les recettes générées, mais il offre en contrepartie une grande simplicité et une

rapidité de mise en œuvre. De plus, aller au plus court peut permettre d'acquérir la maturité nécessaire pour pouvoir plus tard mettre à l'échelle une solution plus riche, et couvrir des besoins plus larges.

Les collaborateurs ont, pour leur part, des usages peut-être plus occasionnels, mais avec des **exigences de qualité et de pérennité** qui nécessitent une utilisation plus fine de ces plateformes. Les cas d'usage les plus partagés portent sur :

- L'automatisation de tâches opérationnelles à faible valeur ajoutée et consommatrices de temps (effectuées actuellement souvent via la bureautique) ;
- La création d'outils pour gérer les processus métiers permettant de fluidifier et simplifier la contribution des différents acteurs ;
- La création d'outils qui facilitent les opérations (comme des applications de support ou de saisie en mobilité).

Pour les équipes de la DSI enfin, l'usage du *Low code* est aussi une nouvelle façon de répondre aux besoins des métiers ou à leurs propres besoins : il permet par exemple de réduire le temps d'un projet en disposant d'une *factory* à applications, de modéliser les données ou d'alléger l'exploitation en utilisant des solutions plateformisées.

Une fois les besoins identifiés et les populations cibles définies, quel cadre la DSI peut-elle mettre en place pour optimiser la réalisation et la sécurisation de ces initiatives ?

2 GOUVERNANCE ET ORGANISATION POUR SUIVRE ET ADMINISTRER LES PLATEFORMES LOW CODE / NO CODE

Mettre en place une gouvernance dédiée aux pratiques du *Low code* ou du *No code* dans l'entreprise est un prérequis nécessaire au bon déploiement de ces plateformes. Cette gouvernance doit permettre d'agir à différents niveaux, tant organisationnel que technique et managérial. Cela recouvre par exemple la définition de critères de sélection des solutions *Low code / No code* avec les métiers, l'élaboration de prérequis techniques et d'architecture, la prise en compte des besoins en compétence, des risques de maintenabilité et de sécurité des applications, ou encore de l'accompagnement des métiers.

2.1 LES OBJECTIFS D'UNE GOUVERNANCE LOW CODE / NO CODE

Côté DSI, la gouvernance doit permettre de maîtriser les solutions qui sont produites sur ces plateformes, d'harmoniser les pratiques et d'optimiser la gestion des développements *Low code / No code*, et ce en :

- Contrôlant les coûts, notamment à cause des modèles type *freemium*[2] ;
- Évitant la multiplication des produits et en rationalisant l'information ;
- Proposant un service de support structuré (PQSR, *templates*, formation, tutoriels…) ;
- Maîtrisant les déploiements, notamment en identifiant les applications à succès et en les mettant sous contrôle ;
- Maîtrisant l'urbanisation des solutions ;
- Assurant la sélection puis la mise en commun et en visibilité des cas d'usages développés ;
- Mettant en place des règles pour répondre aux besoins *business* d'autonomie ou de rapidité d'exécution tout en satisfaisant aux exigences IT (maintenabilité, sécurité, conformité RGPD, etc.) ;
- Accompagnant les métiers et en communiquant sur le périmètre d'autonomie possible du métier, en particulier concernant les aspects techniques.

Cette gouvernance nécessite souvent la mise en place d'un **centre d'expertise** pour adresser la question de l'usage des plateformes *Low code / No code*. Cela permet de bien **préparer les appels d'offres** pour choisir de manière éclairée les intégrateurs ou éditeurs. Plusieurs points de vigilance pourront être alors qualifiés tels que la qualité intrinsèque du code, les types de connecteurs proposés et leur possible réversibilité.

2.2 PILOTAGE DE LA DÉMARCHE

En termes de pilotage, plusieurs positions peuvent être défendues : il peut être opportun de laisser davantage d'autonomie aux métiers tant que les projets restent à petite échelle et non structurants, mais si l'ambition est de systématiser le recours à ces plateformes, il est important de bien définir les rôles dans chaque métier et entité, de développer un co-pilotage DSI-métier, même si la DSI peut prendre la maîtrise de la gouvernance, en la structurant par métiers ou entités.

[2] *Freemium* : Modèle commercial proposant un service gratuit visant à attirer le plus grand nombre d'utilisateurs mais assorti de services complémentaires plus évolués payants.

Avoir une entité dédiée, côté développement, ou un référent faisant l'interface avec les métiers semble essentiel pour rester au plus près des enjeux et des attentes métiers. Par ailleurs, décliner une pratique d'analyse de la valeur de ces initiatives et rentrer dans une logique de capitalisation et de mutualisation des cas d'usage et des réussites, via une "*market place*" par exemple, rendra la démarche visible et attractive.

Dans l'exemple d'une organisation (Pierre Fabre), trois rôles ont été définis pour travailler de concert afin de gérer la demande des métiers :

- **Le *Business process owner* (BPO)** : référent métier faisant déjà partie des métiers, il a pour mission de mieux qualifier les demandes, de gérer l'urgence métier et son besoin. Il est le référent transverse des processus et des données.
- **Le *Coordinateur SI Business Unit* (CSIB)** : garant de la capacité à mettre en œuvre les projets côté métier, il assure un *reporting* consolidé des activités SI pour la BU.
- **Les *Business Relationship Manager*** (BRM) : répartis par domaines (R&D, marketing & commerce, opérations, RH / Finances …), ils gèrent la demande métier et cadrent les besoins SI pour leur domaine fonctionnel. Des comités cross-BRM peuvent être organisés tous les mois pour renforcer la cohérence d'ensemble.

Retour d'expérience de Pierre Fabre
L'accompagnement métier pour les projets small apps

Pierre Fabre s'est doté d'une entité spécifique, *nommée Digital acceleration and information system* (DAIS), fruit de la collaboration d'une entité process & organisation, d'une entité digitale et d'une DSI classique. Son objectif est de couvrir une offre de services totale intégrant à la fois la gouvernance des process métiers, le *data management*, la mise en œuvre et la maintenance des solutions notamment *Low code*.

L'un des modes de solutions qui a été ouvert aux métiers en 2018 concerne les *small Apps.* Sur ce sujet, l'autonomie est laissée au métier en termes de *delivery* mais dans un cadre maîtrisé. La DAIS va alors :

- S'assurer que le service n'est pas déjà existant ou en construction dans le catalogue DAIS ;
- Référencer la solution ;
- Assurer la conformité aux règles de sécurité, à la réglementation et aux standards du groupe ;
- Définir et maintenir les architectures et les solutions de référence ;
- Mettre à disposition des plateformes sécurisées, supervisées et sauvegardées ;
- Gérer l'intégration avec le SI (APIs / Interfaces) ;
- Apporter des *guidelines* et une expertise technique.

Le socle de cette démarche repose sur la gestion de la demande (et donc des dialogues et échanges entre les 3 acteurs : BRM, CSIB, BPO) qui permet de guider les métiers vers ce type de solution.

Matthieu Garcia, *Architecte d'Entreprise, Pierre Fabre*

2.3 LES CRITÈRES D'ÉLIGIBILITÉ

L'enjeu de gouvernance se pose dès la phase de sélection des plateformes à partir de critères d'éligibilité qui seront définis en fonction du périmètre utilisateur et de ses besoins. Il peut être judicieux de commencer par définir un **périmètre d'usage** et **d'exclusion** via un **arbre de décision.** Celui-ci peut être construit avec les métiers, comme dans l'exemple ci-dessous :

Retour d'expérience de la STIME (Groupement des Mousquetaires)
Arbre de décision Low code / No code

Les équipes de la Dev Factory (pôle gérant les applications DotNet/React et AppSheet/PowerApps) ont travaillé sur un arbre de décision permettant de valider qu'une application est réalisable avec AppSheet ou avec PowerApps, les deux solutions les plus utilisées dans le groupe (Si elle ne l'est avec aucune des deux, un développement classique sera nécessaire). Une première étape repose sur la validation de la "*Design Authority*", organe de décision réunissant développeurs, utilisateurs, architectes et les équipes de sécurité. Son rôle est d'assurer :

- La revue des applications créées sur la période dans le but d'éviter les doublons et de détecter les applications sur lesquelles on peut communiquer ;
- La revue et l'acceptation de *patterns* d'architecture applicative mêlant plateforme *No code* et interaction avec le SI ;
- La revue des demandes de mise à disposition de sources de données issues du SI via API.

Un questionnaire est également délivré aux métiers. Il comprend des questions types comme :

- L'application est-elle stratégique, quel est le nombre d'utilisateurs visés, quels sont les besoins de performance ?
- Quel est le niveau de complexité UI/UX ?
- Quelle est la durée de vie de l'application, etc. ?

À partir des réponses, des recommandations seront formulées. Si une application est sélectionnée, suivant le type de projet et de public, différentes instances seront mobilisées : un comité d'architecture et/ou un comité de gouvernance. Ensuite, il sera décidé si un plan de formation doit être lancé, ou si le déploiement peut être débuté directement. La maintenance et le suivi de l'application sont ensuite assurés par l'équipe Dev Factory, qui pourra en faire état lors des comités de gouvernance.

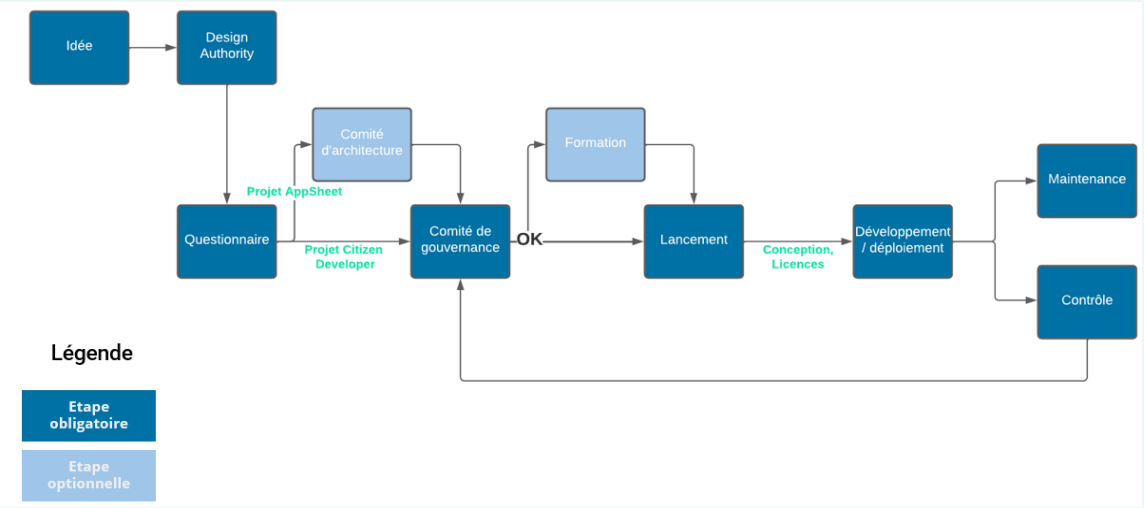


Figure 3 : Arbre de décision Low code / No code (STIME)

Christophe Gorgeot, *Responsable Domaine DEV FACTORY, STIME (Groupement des Mousquetaires)*

Élaborer un **arbre de décision** permet d'évaluer la pertinence du choix d'une plateforme *Low code / No code* ou d'un développement spécifique, et de traiter par exemple les questions qui se posent sur l'aspect stratégique de l'application, le nombre d'utilisateurs concernés, les besoins de performance, le niveau de complexité de l'*UI/UX*[3], ou encore la durée de vie de l'application.

Les critères d'éligibilité pourront se concentrer par ordre de priorité sur :

- L**e coût des licences** et leur **modèle :** la prédictibilité des coûts vis-à-vis des usagers visés n'est pas toujours aisée. Il faut de plus bien délimiter le périmètre des usages et s'interroger sur le choix de licence fait en fonction du nombre d'utilisateurs ou du nombre de développeurs. La capacité d'administration du portefeuille par l'IT est à prendre en compte également.
- La **sécurité** de la plateforme : une exigence de certification SecNumCloud concernant l'hébergeur, la garantie d'un accès structuré et sécurisé aux données, la possibilité d'audit du SI ou des hébergeurs, la réactivité du support éditeur, et la stabilité de l'environnement seront des critères clés à prendre en compte. Le **niveau de risque** doit aussi être évalué en interne, en fonction de la criticité des données des applications concernées ou de leur niveau d'exposition aux données stratégiques de l'entreprise.
- La **montée en compétence rapide** : il est essentiel que la solution puisse être prise en main rapidement par les utilisateurs et que le délai de formation soit court et efficace. L'*UX design* est un critère essentiel pour que l'approche soit appréciée des métiers.
- La **pérennité et l'évolutivité** de la solution : il est important que les utilisateurs qui développent des applications via les plateformes *Low code / No code* assurent leur pérennité dans le temps en documentant le plus possible leur travail. Un accompagnement et une sensibilisation spécifiques sur ce sujet sont essentiels.

[3] *User Interface* (interface utilisateur) / *User Experience* (expérience utilisateur)

- Les **connecteurs natifs** : ceux-ci doivent permettre une intégration rapide et optimale des applications *Low code / No code* au SI de l'entreprise, mais aussi de fonctionner en DevOps, ou en mode « déconnecté ».
- La facilité d'application des règles de **l'accessibilité numérique** : ces règles doivent être prises en compte le plus en amont possible de la conception. Les plateformes *Low code / No code* ne prennent pas forcément en compte dans leurs offres cet enjeu sociétal fort, encore trop méconnu. De plus, la réglementation évolue en la matière. L'accessibilité est déjà obligatoire pour un certain type d'organismes (article 47 de la Loi du 11 février 2005 pour l'égalité des droits et des chances, la participation et la citoyenneté des personnes handicapées) et la directive européenne "Biens et services" [4] va intégrer de nouveaux services dans cette obligation.

2.4 ACCOMPAGNEMENT ET RESPONSABILITÉS MÉTIER

Enfin, l'**accompagnement des pratiques des collaborateurs** est au cœur de la gouvernance. Certaines organisations ont mis en place une stratégie spécifique au *no code* avec des *guidelines* et recommandations permettant de s'assurer que les bonnes pratiques en matière de cybersécurité, de confidentialité ou encore de maintenance technique sont bien suivies et respectées. Plusieurs entreprises négocient également avec leur intégrateur ou éditeur l'accompagnement dans le déploiement de la gouvernance et dans la mise en place de petites formations adaptées aux profils qui vont utiliser ces plateformes.

La gouvernance doit passer par **une pédagogie avec les différentes parties prenantes**. La DSI ne peut pas imposer ses règles seule, il est nécessaire de mettre les métiers en responsabilité, comme dans les processus d'agilité.

Deux positionnements peuvent être pris concernant l'enjeu d'accompagnement métier. La DSI peut choisir de se doter d'un **pool de développeurs pour les métiers** : dans ce cas, la DSI met des développeurs à la disposition des métiers et prend en charge l'urbanisation et les risques. Mais cela a un coût.

L'autre positionnement consiste à **proposer un « accompagnement » au développement** à la main des métiers, sans se doter d'une entité dédiée : dans ce cas, il est important de cibler les populations, le plus souvent en attribuant le *no code* pour les *citizen developers* et le *low code* pour les métiers ayant davantage d'expertise. L'idée de proposer des hackathons est bonne pour sensibiliser et accompagner de façon ludique.

[4] https://eur-lex.europa.eu/legal-content/FR/TXT/?uri=uriserv:OJ.L_.2019.151.01.0070.01.FRA&toc=OJ:L:2019:151:TOC

Les bonnes pratiques en synthèse

En termes de gouvernance et de pilotage des solutions *Low code / No code*, on peut retenir les bonnes pratiques suivantes :

- L'élaboration d'un **arbre de décision**, co-construit avec les métiers afin d'identifier les critères clés d'éligibilité des solutions *Low code / No code* ;
- La définition du **périmètre d'usage** de l'application afin de bien clarifier les besoins et d'anticiper les moyens et ressources nécessaires ;
- **La cartographie des applications** *Low code / No code* déjà en place et la mise à disposition de ce catalogue auprès des métiers ;
- La définition des **rôles** : être clair sur les responsabilités de chacun, concernant par exemple la **MCO** et la **sécurité**, et intégrer des **architectes** et les équipes de **cybersécurité le plus en amont possible des démarches.**
- **L'accompagnement métier** afin d'anticiper les risques, et promouvoir les bons usages :
    - **Partager** ce qui est fait, et faire **évaluer** les applications développées par les utilisateurs finaux ;
    - Mettre en place une **communauté de bonnes pratiques** pour capitaliser sur les bonnes et moins bonnes expériences ;
    - Proposer des **formations courtes** pour les *citizens developers ;*
    - Mettre en place **un dialogue permanent avec les métiers** sur : la pédagogie, les risques, la valeur de ces plateformes ;
    - **Proposer des hackathons** autour du *Low code / No code* avec le métier : ce type de challenge comporte un aspect ludique et peut rendre l'accompagnement plus efficace.
- **L'identification des compétences requises** afin d'anticiper la mobilisation de ressources et de compétences :
    - Créer une équipe de développement rapide pour supporter les besoins métiers ;
    - Ou autoriser le développement à la main des métiers, quand les applications sont peu critiques.

3 LES DEFIS ET OPPORTUNITES POUR L'IT

Dans ce contexte, les DSI doivent faire face à différents enjeux dans la phase opératoire du déploiement des plateformes *Low code / No code*, qu'il s'agisse d'urbanisme et d'architecture, de cybersécurité, de réversibilité ou de gestion des coûts, mais aussi de challenges en termes d'innovation et de valorisation des relations IT-métiers.

3.1 URBANISME ET ARCHITECTURE

Le *No code* oblige à envisager des solutions simples sur des process non critiques : les utilisateurs métiers vont donc débuter les projets avec enthousiasme mais avec un risque déceptif car l'application aura des limites, notamment sur le plan ergonomique et fonctionnel. La DSI doit prendre en compte ce risque de demandes d'évolutions fonctionnelles qui ne peuvent parfois pas être résolues. On touche ici aux limites des solutions sur étagère. Se pose alors la question du **design d'usage** : les DSI sont sollicitées pour fluidifier les interfaces afin d'améliorer l'expérience utilisateur. Il peut être intéressant dans ce cas de prendre en compte ce paramètre dans les critères de sélection des offres *Low code / No code*, ou de proposer des formations d'ergonomie et d'*UX design* aux métiers, ou encore d'y consacrer une équipe dédiée. Un autre enjeu pour la DSI repose sur l**'urbanisme d'usage** qui concerne l'organisation et le référencement des applications *Low code / No code* afin d'optimiser les performances et les ressources pour chaque initiative.

Certains DSI constatent qu'avec le *No code*, seule la phase de développement est réellement accélérée et que peu de temps est gagné sur l'ensemble du *process*, à cause de la complexité plus ou moins grande à connecter les flux, selon l'environnement choisi. Certains environnements en effet, nécessitent de passer par des *webs APIs*, et obligent à faire des développements spécifiques. Il est parfois nécessaire de définir un circuit court de livraison et d'adapter les *process* de l'entreprise pour ces applications peu sensibles.

3.2 CYBERSÉCURITÉ

Les DSI doivent faire face à différents types de risques : l'**utilisation des données** est l'un des premiers enjeux. Il faut tout d'abord veiller à ce que l'usage des données ne mette pas en péril leur intégrité, leur conformité, ni la sécurité d'un canal. Par ailleurs, certaines applications peuvent parfois être détournées de leur usage initial, ce qui oblige à revoir l'ensemble des *process*. Veiller au respect des **référentiels de sécurité et confidentialité** et aux autorisations d'accès est un point clé dans ces environnements sans cesse mouvants et évolutifs.

Les équipes de cybersécurité déjà soucieuses d'améliorer la sécurité *by design* des développements classiques s'inscrivent naturellement dans les démarches *Low code / No code*. Ces équipes sont à l'aise avec certaines solutions qui sont dans des environnements déjà balisés et sécurisés. L'une des difficultés majeures rencontrées réside dans les modèles *Low code / No code* qui n'embarquent pas leur propre base de données, car cela nécessite d'exposer les données de l'entreprise et de mettre en place des APIs pour sécuriser les requêtes.

3.3 RÉVERSIBILITÉ

L'un des points qui soulève le plus de réticences à la systématisation des approches *Low code / No code* est la question de la réversibilité (ou "*vendor lock-in*") : cela signifie qu'une fois la souscription à une offre résiliée, l'application devient inutilisable. L'une des solutions est alors de pousser les équipes à **documenter le plus possible** leur travail, comme pour un développement classique, afin de pouvoir récupérer les informations s'il faut réécrire l'application dans un nouvel environnement. Une autre possibilité repose sur l'utilisation de ***frameworks* standards**, ou encore sur l'usage de plateformes ***open source***. De nombreuses entreprises ont des stratégies *open source* qui répondent à leur exigence de souveraineté et de réduction de leur dépendance technologique. Elles peuvent ainsi considérer les approches *Low code / No code* comme intéressantes en ce sens.

3.4 OPPORTUNITÉS

Les approches *Low code / No code* constituent aussi de véritables opportunités, y compris pour la DSI elle-même si l'on considère par exemple qu'elles permettent de **développer ou d'enrichir les plateformes de développement à moindre coût** dans un contexte de contraintes financières, ou encore de poser un cadre de fonctionnement avec les métiers plus sécurisé et standardisé que le *shadow IT.* L'approche *Low code / No code* peut aussi **servir la stratégie *data*** de l'organisation.

Une véritable dynamique de créativité et d'innovation avec les métiers peut en découler, avec une offre de service répondant **plus rapidement** au besoin en petites et moyennes applications, et la possibilité pour les responsables d'application de concevoir des **prototypes** plus facilement. Une autre dimension positive de ces approches est **l'acculturation des métiers aux enjeux IT**, grâce à l'appétence et la curiosité que suscitent ces environnements parfois ludiques et créatifs. L'aspect collaboratif du *Low code* a ainsi été mis en avant par l'expérience d'**Enedis** :

Retour d'expérience d'Enedis
Le Low code comme levier de transformation des projets data

Enedis a fait le choix d'une solution qui cible les *data analysts* dans les régions pour améliorer la phase amont de leur projet *data* et le faire en toute sécurité dans un environnement unique et sécurisé, limitant ainsi les initiatives locales de *shadow IT*. Cette solution est très puissante dans la préparation et le nettoyage de données, et offre les garanties d'une solution industrielle et sécurisée. Elle peut être installée *on-premise,* et c'est l'une des rares à offrir cette possibilité. De plus, elle facilite la collaboration entre les différentes équipes. Ainsi, la solution a permis d'améliorer plusieurs aspects dans la réalisation de projets *data* en :

- Rendant possible une dynamique collaborative entre les *data scientists* au niveau national et les *data analysts* en région ;
- Réduisant les délais de mise en œuvre et en optimisant l'utilisation des ressources rares ;
- Passant du stade expérimental au stade industriel de manière fluide et « *by design* » ;
- Acculturant les métiers à l'ouverture maitrisée des données : la solution est reliée au *datalake* de l'entreprise, ce qui permet de se connecter à des outils *BI* (Business

> Intelligence), de créer des fichiers de données, d'APIs ou de service de données internes.
>
> **Benoit Locu**, *Head of Enedis Innovation Labs, Enedis*

Enfin, la gouvernance *Low code / No code* permet de **proposer un cadre** à cette dynamique d'innovation, en embarquant tous les acteurs dans une logique standardisée, en mettant en place des arbres de décision, et en animant les communautés de développeurs et de *citizen developers* (non sans défis puisqu'il existe un grand *turn over* dans les métiers du développement), tout en maintenant les objectifs de sécurité et de qualité du code.

Les bonnes pratiques en synthèse

En termes d'urbanisme SI et d'architecture des solutions *Low code / No code,* les bonnes pratiques à retenir portent sur les points suivants :

- Avoir une **bonne maturité « d'APIsation »** du SI et acculturer l'ensemble des acteurs aux enjeux d'ouverture des données ;
- Exiger une **documentation des outils** au même niveau que celle d'un développement spécifique classique afin d'éviter le verrouillage de la solution (ou *vendor lock-in*), ou privilégier les **approches *open source*** ;
- Définir et mettre en place des **règles d'urbanisme d'usage** voire de **design d'usage** face à la démultiplication du *No code* : cela doit permettre de cadrer l'évolution des fonctionnalités afin d'éviter que l'application ne devienne inutilisable ou non conviviale. Ces règles doivent aussi permettre d'éviter que des données d'entreprise ne soient ingérées sans contrôle par des solutions *Low code / No code* ;
- **Vérifier annuellement le catalogue d'applications** et leur possible optimisation ;
- Articuler l'usage de ces plateformes avec la **stratégie *data*** de l'organisation.

4 *GUIDELINES* À DESTINATION DES *CITIZEN DEVELOPERS* ET DES ÉQUIPES IT

Le guide ci-dessous, issu des réflexions d'une organisation participante, a été enrichi par les membres du groupe de travail. Il présente les bonnes pratiques à destination des utilisateurs-développeurs (*citizen developer*s) d'une part, et des équipes IT d'autre part.

4.1 POUR LES UTILISATEURS-DÉVELOPPEURS (*CITIZEN DEVELOPERS*)

1. **Utiliser les outils sur des périmètres délimités** :
   - Les applications développées doivent être de faible complexité et ne pas concerner les domaines métiers critiques. Elles peuvent servir par exemple les périmètres suivants : communication, événement ponctuel, *process* interne etc.
   - Les applications doivent être développées par un seul service ou une seule équipe.
2. **Assurer la propriété de l'application** : la propriété sera soit celle de l'utilisateur-développeur, soit celle d'une autre personne qui choisit d'être responsable et de gérer tout le cycle de vie de l'application. Si personne n'est volontaire pour assurer le suivi et la maintenance, il faudra décommissionner l'application.
3. **Définir à quel moment l'application devient critique** : si le périmètre fonctionnel augmente et que la solution concerne un plus grand nombre d'utilisateurs, si l'application bloque des *process* ou qu'elle est indisponible, il est nécessaire de déléguer son entière gestion à l'IT.
4. **Utiliser les APIs existantes pour alimenter les applications qui seront développées** : L'accès sécurisé aux données se fait de préférence en utilisant les APIs existantes : cela assure un accès sécurisé aux données qualifiées (clients, contrats, etc.) et permet à l'application de garantir la qualité et la cohérence des données.
5. **Interdire la redondance des fonctionnalités déjà présentes dans une application régalienne :** Les applications développées ne doivent pas dupliquer ou remplacer les applications existantes ni créer de passerelles avec les applications déjà en place (se référer au catalogue d'applications de l'organisation). Elles doivent être développées dans le but de faciliter, de simplifier ou d'accélérer l'acquisition de données, ou de réduire la communication fastidieuse entre les équipes ou les départements.
6. **Partager des retours d'expérience** dans le cadre d'une communauté de *citizen developers* par la mise en place par exemple d'une *market place.*
7. **Arbitrer le maintien d'une application en fonction du ROI ou de l'évolution de la criticité de l'application.**

4.2 POUR LES ÉQUIPES IT

1. **Définir des zones sécurisées** : il est nécessaire d'établir un partenariat fort entre l'IT et les métiers afin d'éviter tous les impacts négatifs du *shadow IT*. Il faut définir les limites et un cadre en proposant le soutien de développeurs professionnels, en élaborant une supervision

et une gouvernance informatique strictes avant la diffusion, et en identifiant ce qui est entièrement hors du cadre des *citizen developers* et qui doit être géré dans le portefeuille informatique et par les départements informatiques.

2. **Avoir des cas d'utilisation identifiés** comme par exemple la conception de formulaires, la conception d'applications de collecte de données, d'orchestration des processus métiers ou de *workflows*. Si les métiers veulent développer d'autres applications, en dehors de cette liste ou trop complexes, il est nécessaire de consulter l'IT.
3. **Utiliser au maximum ce qui existe déjà dans le *cloud*** : les procédés d'authentification, d'autorisation et de propriété doivent être réutilisés par les *citizen developers*, afin d'éviter de développer une nouvelle fois ces parcours. L'utilisation de fonctionnalités apportées par le *cloud* (authentification de sécurité, séparation des droits des utilisateurs) donne plus de possibilités d'appliquer les règles de gouvernance et de sécurité (permissions basées sur des règles...).
4. **Utiliser des APIs** : afin d'interfacer des systèmes et de transférer des données dans leur application, les *citizen developers* doivent utiliser les APIs existantes disponibles, en consultant le catalogue des APIs de l'organisation. Si aucune API n'est disponible, il faudra contacter la direction informatique pour évaluer la meilleure façon d'accéder aux données requises.
5. **Définir les rôles et accompagner le *citizen developer* sur la sécurité** : lorsqu'il n'est pas possible de passer par une API, il est essentiel de sécuriser les données à travers la mise en œuvre des contrôles d'accès, la configuration de l'application, et la définition des rôles et responsabilités. Ces tâches requièrent l'expertise de la DSI.
6. **Respecter la classification des données qui est imposée en interne** (et qui doit être connue de tous) afin de respecter le niveau de sécurité souhaité. Des audits pourront être réalisés *a posteriori*.

CONCLUSION

Le mouvement *Low code / No code* n'en est qu'à ses débuts. Il est le symptôme positif de la numérisation croissante des métiers de l'entreprise, et d'une évolution des pratiques chez les développeurs. Ses utilisateurs, qu'ils soient *citizen developers* ou professionnels, souhaitent de plus en plus avoir accès à des environnements similaires à ceux qu'ils connaissent dans le monde de l'informatique grand public. La réponse qu'offre l'usage de ces plateformes à l'engorgement des DSI et à l'autonomisation des métiers est une aubaine, mais il est nécessaire d'orchestrer et de maîtriser ces nouveaux environnements dans la totalité de leur *process*.

La relation entre la DSI et les métiers est ici au cœur d'une gouvernance spécifique, permettant de canaliser et d'arbitrer les choix de ces plateformes suivant de nombreux critères touchant tant au besoin utilisateur, à l'urbanisation ou à la sécurité qu'à la définition des rôles et responsabilités.

Certains outils méthodologiques sont essentiels à mobiliser comme l'arbre de décision ou le questionnaire co-construit avec les métiers. Ils permettent d'aider à l'identification des critères clés d'éligibilité des solutions *Low code / No code*, mais aussi de favoriser l'acculturation des métiers aux enjeux IT et leur responsabilité en la matière.

Les années à venir nous montreront si le manque de compétences en développement perdurera et orientera les entreprises à privilégier volontairement ce type d'approche pour compenser la pénurie de développeurs, ou si le *Low code / No code* ne suscitera pas, de lui-même, de nouvelles vocations permettant de faire émerger ces compétences au sein même des organisations.

Cigref
RÉUSSIR
LE NUMÉRIQUE

Au service de la croissance économique et de la compétitivité de nos membres, grandes entreprises et administrations publiques françaises, utilisatrices de solutions et services numériques, par la réussite du numérique.

Le Cigref est un réseau de grandes entreprises et administrations publiques françaises qui a pour mission de développer la capacité de ses membres à intégrer et maîtriser le numérique. Par la qualité de sa réflexion et la représentativité de ses membres, il est un acteur fédérateur de la société numérique. Association loi 1901 créée en 1970, le Cigref n'exerce aucune activité lucrative.

Pour réussir sa mission, le Cigref s'appuie sur trois métiers, qui font sa singularité.

Appartenance
Le Cigref incarne une parole collective des grandes entreprises et administrations françaises autour du numérique. Ses membres partagent leurs expériences de l'utilisation des technologies au sein de groupes de travail afin de faire émerger les meilleures pratiques.

Intelligence
Le Cigref participe aux réflexions collectives sur les enjeux économiques et sociétaux des technologies de l'information. Fondé il y a près de 50 ans, étant l'une des plus anciennes associations numériques en France, il tire sa légitimité à la fois de son histoire et de sa maîtrise des sujets techniques, socle de compétences de savoir-faire, fondements du numérique.

Influence
Le Cigref fait connaître et respecter les intérêts légitimes de ses entreprises membres. Instance indépendante d'échange et de production entre praticiens et acteurs, Il est une référence reconnue par tout son écosystème.

www.Cigref.fr
21 av. de Messine, 75008 Paris
+33 1 56 59 70 00
Cigref@Cigref.fr