Comment intégrer des Progiciels dans une Architecture d'Entreprise ?

Juin 2011

Pour leur contribution active à la fabrication de ce livre blanc, nous tenons à remercier tout particulièrement :

- **Afdel** : Loïc Rivière et Stéphanie Jullien
- **Air France** : Laurent Mondemé, Francis Bailly
- **Amadeus** : Stéphane Tabary
- **Axa** : Hervé le Hen, Luc Morin
- **Cegid** : Florence Desprets, Jean-Louis Decosse
- **Club des Pilotes de Processus** : Didier Lejeal, Henri Chelli, Henri-Paul Soulodre, Pierre Dautet, Michel Verdun
- **Conseils**-Plus : Pierre Hugot
- **ITN** : Alain Dubois
- **Logica** : Emmanuel Pesenti
- **Oracle** : Didier Medal
- **OR System** : Anis Bada
- **Renault** (RCI Banque): Olivier Chagnou
- **SAP** : Thierry Pierre
- **SFEIR :** Didier Girard
- **Stallergènes :** Jean-Pierre Bouillaguet, Jacques Lelièvre
- **Total** : Bruno Bosquette
- **W4** : François Bonnet
- **Weave :** Julien Soyer, Olivier Grandjean
- **Wyde** : Laurent Pytel

Note : les termes avec une première lettre majuscule correspondent aux termes définis dans le glossaire du CEISAR. Exemple : Entreprise, Progiciel, Processus, …

Table des matières

1 Objet du livre blanc et méthode suivie

1.1 Pourquoi lire ce livre blanc

Depuis deux décennies, le « Progiciel» est devenu la réponse privilégiée dans la stratégie informatique des Entreprises pour répondre à leurs enjeux d'excellence opérationnelle, devenant dans le même temps le leitmotiv, la panacée ou le tourment de tous les managers en charge de leur mise en place.

Les dernières années ont ainsi vu progresser le « Progiciel » d'une banale Solution de Commodité fonctionnelle à une véritable alternative Métier au cœur d'une Architecture d'Entreprise historiquement fondée et structurée sur le développement interne de « logiciels sur mesure ».

Mais ce recours constaté aux progiciels pour des besoins « Métiers » cruciaux fait naître de nouvelles interrogations particulièrement à l'heure où les architectes modernes ne prophétisent que par les Architectures Orientées Services (SOA) et le Cloud computing.

- Quand faut-il avoir recours à un progiciel et pour quel type de besoin?
- Comment sélectionner LE bon progiciel Métier dans une offre foisonnante et selon quels critères (Éditeur, modèle, déploiement, évolution, migration, ROI…)? Quelle gouvernance pour éviter de choisir des Progiciels indépendants ?
- Quelles sont les bonnes pratiques pour mettre en œuvre un Progiciel?
- Quelles précautions doit-on prendre pour garantir la cohérence de son système d'information (Interface, plate-forme de développement, standard, évolution, services associés…) ? Quelle grille d'analyse des Progiciels pour comprendre leur Architecture interne et l'intégrer au mieux à l'existant de l'Entreprise ?
- Comment garantir la coexistence efficace de progiciels indépendants dans une Architecture unifiée et harmonieuse? Comment faire cohabiter une Architecture orientée service (SOA) avec des Progiciels ?
- Comment maitriser les coûts d'installation, de paramétrage et d'évolution du progiciel retenu?
- Quelles exigences doit avoir un client vis-à-vis des éditeurs de Progiciels ? (accès à l'Architecture interne, services ouverts, respect des standards du marché et du client)
- Quelle est la vision des éditeurs de logiciels sur l'intégrabilité des produits futurs
- Le Cloud esquisse-t-il une métamorphose de cette tendance progicielle ou participe-t-il à son expansion? Le Cloud change-t-il les règles du jeu ?
- Quelle formation doit-on donner aux Architectes d'Entreprise qui œuvrent dans un contexte composé de Progiciels ?

Si l'une de ces interrogations vous préoccupe, nous vous invitons à découvrir ce Livre Blanc, qui s'appuie sur les approches appliquées par vos confrères dans le choix, l'installation et le déploiement de progiciels.
Recueil de cas concrets, ce livre s'appuie sur les expériences, les réalités d'usage et de déploiement de véritables projets pour éclairer et orienter sereinement les managers confrontés à l'intégration de progiciels dans leur Architecture.

1.2 Objectifs

Tous les six mois les Sponsors du CEISAR (Air France, Axa et Total) définissent un nouveau thème qui doit se traduire par un Livre Blanc. Le thème défini en janvier 2011 est : « comment prendre en compte la dimension **Architecture Globale** du SI de l'Entreprise dans les **choix de Progiciels** », c'est-à-dire comment les aider à trouver le **bon compromis** entre :

- **l'offre** foisonnante de Progiciels et de Services-logiciels qui semblent prêts à les satisfaire immédiatement
- le besoin de **cohérence globale** du Modèle de l'Entreprise: référentiels partagés, pas de double saisie, usage homogène pour les utilisateurs, exploitation informatique cohérente…
- les **contraintes de l'existant** qui ne permettent pas de conduire une démarche d'Architecture partant de la page blanche

Le CEISAR a donc décidé de se concentrer sur la problématique de l'intégration des Progiciels dans une Architecture d'Entreprise. Nous ne traiterons donc pas de thèmes qui ont parus trop éloignés de la question posée tels que :

- Quelles compétences développer en interne sur le pilotage de projet et le Progiciel lui-même ?
- Comment réaliser une étude d'écart fonctionnelle?
- Comment gérer le changement ?
- Quid des tests de non-régression lors de montées de version, de la qualité de bout en bout ?
- Quelle documentation attendre du Progiciel ?

1.3 La démarche suivie

A notre grande surprise, nous n'avons **pas trouvé d'étude générale** qui traite de ce thème et sur laquelle nous aurions pu nous appuyer pour démarrer nos travaux.

Nous avons donc:

- Réalisé des **études de cas** : une par sponsor (Air France, Axa, Total) et quelques études complémentaires dans d'autres Entreprises (La Poste, Renault, Stallergènes)
- Interrogé des **éditeurs de Progiciels** métier (Amadeus, CEGID, Oracle, SAP, W4, Wyde)
- Interrogé des **intégrateurs** (Logica) ou sociétés de conseil (Conseil-Plus)

Les études de cas présentées sont toutes « positives » : elles ont abouti avec succès à l'installation du Progiciel ; on n'a donc pas pris en compte de projets en échec. Par contre nous avons essayé d'analyser les difficultés rencontrées dans ces projets qui ont abouti.

Pour nous aider à comparer les différentes **études de cas**, nous avons construit un questionnaire type (voir annexe 1).

Entreprise	**Etude de cas**
Air France	Mise en place d'un nouveau CRM pour la division cargo dans le cadre de la fusion avec KLM / Solution à base de Salesforce.com
AXA France	Mise en place de la gestion des sinistres dommages avec le Progiciel Guidewire
La Poste	Mise en place d'un bureau de poste virtuel sur internet avec IBM Websphere Commerce
RCI Banque	Renault crée une compagnie d'Assurance avec le Progiciel Wynsure
Stallergènes	Mise en place d'un worflow Achats avec W4
TOTAL	Mise en place d'une solution Groupe de gestion des Ressources Humaines avec SAP HCM

Nous nous sommes aussi appuyés sur un questionnaire destiné aux **éditeurs** de logiciel (voir annexe 2) : Amadeus, CEGID, Oracle, SAP, W4, Wyde .

L'AFDEL (http://www.afdel.fr/), l'Association Française des Editeurs de Logiciels, s'est associée à cette étude sous forme de recommandations sur le contenu de ce document.
Le **Club des Pilotes de Processus** (http://www.pilotesdeProcessus.org/) s'est aussi associé à la démarche dans le cadre de son atelier « Processus et ERP ».
Nous proposons d'aborder cette problématique de la façon suivante :

- comment évoluent les **besoins** de l'Entreprise
- comment évolue **l'offre** des éditeurs
- quelle est le niveau d'**Architecture d'Entreprise** cible utile pour les Entreprises ayant recours aux Progiciels ?
- Quelles sont les bonnes pratiques aux différentes étapes du cycle de vie du Progiciel :
  - comment **choisir** un Progiciel ?
  - comment **installer** un Progiciel ?
  - comment faire **évoluer** un Progiciel ?
- quelle **organisation** et quelles **compétences** sont nécessaires (en rapport avec l'Architecture d'Entreprise) ?

2 Comment définir et caractériser les Progiciels ?

RÉSUMÉ

- Un **Progiciel** est une Solution développée, supportée et commercialisée par un **Editeur de Logiciel**. Elle s'oppose donc aux Solutions Spécifiques développées en interne : elle est destinée à plusieurs clients.
- Un **Progiciel-Solution** modélise un ensemble de Processus d'un Domaine Fonctionnel : Progiciel de CRM, Progiciel Comptable, Progiciel de « Supply chain ».
- Un **Progiciel-Composant** Modélise des Fonctions réutilisables par différents Processus aussi bien au niveau technique qu'au niveau métier: fonctions techniques telles que EAI, fonctions de sécurité, fonctions de GED, … et les Services-logiciels utilisables dans une approche SOA qui peuvent être utilisés par les autres Solutions.
- Les Progiciels-Solutions sont décomposés en **Progiciels de Commodité** (Progiciel de comptabilité, de gestion de projet, de gestion de ressources humaines, de messagerie, de pilotage) et **Progiciels Verticaux** offerts à chaque secteur économique (Progiciel pour le secteur bancaire, Progiciel pour le secteur des télécoms, Progiciel pour le secteur de la distribution).
- Les Progiciels-Solutions peuvent être **par domaine** (paie, comptabilité) ou **Intégrés** on parle alors respectivement de Progiciels de Commodité Intégré (ou **ERP**) ou de Progiciel **Vertical Intégré**.
- Un Progiciel peut avoir un **couplage fort ou faible** avec les autres Solutions de l'Entreprise
- Un Progiciel peut être conçu pour un usage **international**
- Un Progiciel peut ou non être personnalisé (par **configuration** ou par **développement complémentaire**)
- Un Progiciel peut être fourni avec des données de configuration **préremplies** pour accélérer son déploiement
- Un Progiciel peut être facturé sous forme de **licences** ou à **l'usage** (Cloud Computing)
- In Progiciel s'intègre avec les autres Solutions de l'Entreprise par des interfaces qui peuvent être **synchrones, asynchrones ou au fil de l'eau**

2.1 L'Entreprise

Pour assurer une bonne cohérence entre nos livres blancs, nous réutilisons le glossaire du CEISAR dont nous rappelons les termes essentiels (voir le livre blanc sur le « Langage de la Transformation » dans www.ceisar.org):

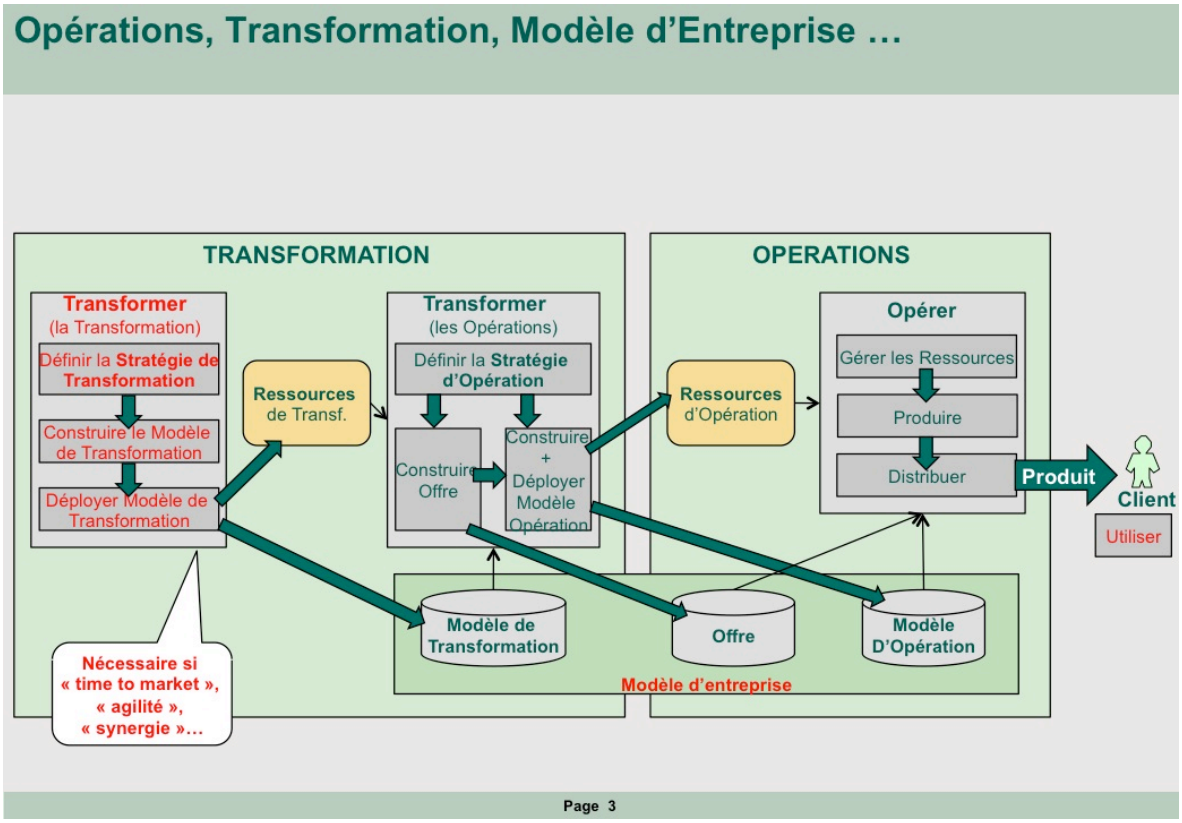
L'Entreprise :

- la Mission d'une Entreprise est d'apporter de la **Valeur** à ses **Clients** en lui délivrant des **Produits.**
- Pour finir par délivrer ce Produit, l'Entreprise doit agir : elle exécute des **Processus**. Un Processus Métier est un enchainement de **Fonctions**, déclenché par un événement indépendant et qui produit un livrable pour le Client du Processus.
- Elle agit grâce à des **Ressources** : Acteurs Humains, Acteurs-Ordinateurs, Ressources financières, Informations, Locaux, composants Matériels.
- Les **Acteurs** d'une Entreprise (Acteurs-Humains et Acteurs-Ordinateurs) **Opèrent** (produire, vendre, administrer les ressources, piloter…) selon un **Modèle d'Opération** qui formalise comment bien Opérer : Modèle documentaire pour les Acteurs humains et Modèle-Logiciel pour les Acteurs-Ordinateurs (un logiciel est un Modèle).

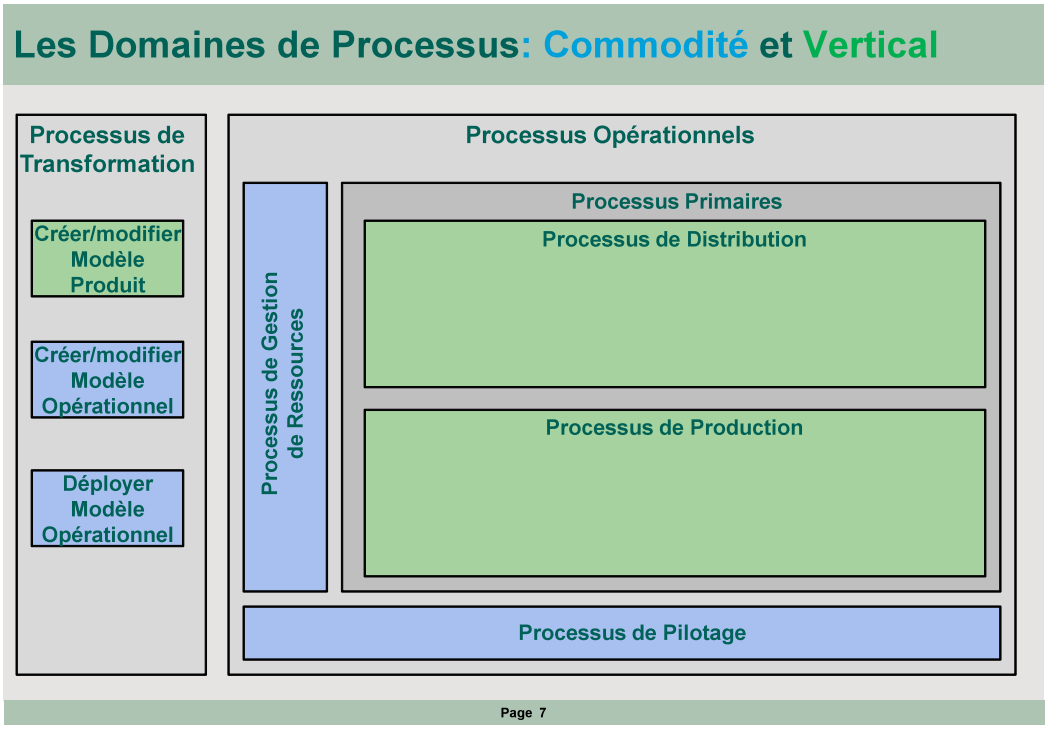
Le Modèle d'Entreprise et la Transformation

- les Processus du **Modèle d'Opération** sont regroupés en Domaines Fonctionnels : Domaines Ressources humaines, Domaines Distribution, Domaine Production…
- Les Produits (Biens ou Service ou Information) délivrés aux Clients par l'Entreprise respectent un **Modèle-Produit**.

- **Transformer** une Entreprise consiste à créer/modifier son Modèle d'Opération et/ou son Modèle Produit et à le Déployer.
- Pour bien Transformer, les Acteurs de la transformation (stratèges, chef de projet, maitrise d'ouvrage, développeurs,…) respectent un **Modèle de Transformation** (méthodologie, outils, rôles de la Transformation).
- Le **Modèle d'Entreprise** est la somme de Modèle Produit + Modèle d'Opération + Modèle de Transformation.
- Le Modèle d'Opération est décomposé en **Solutions** : une Solution peut Modéliser un ou plusieurs Processus (Solution CRM, Solution comptable, Solution gestion des ressources humaines) ou un sous-ensemble de Processus appelé **Fonction** (Solution « Tarification », Solution « Sécurité », …)
- Chaque **Solution** couvre un besoin fonctionnel et inclut à la fois les Modèles de Processus, les Modèles d'information et les Logiciels correspondants. Une Solution est caractérisée par le fait qu'elle est bâtie sur une seule **Fondation** : ensemble cohérent d'outils et de composants réutilisés pour Modéliser les fonctionnalités de la Solution (voir le Livre Blanc du CEISAR sur la Fondation).
- Elle peut représenter un ou plusieurs **Processus** (Solution « CRM », Solution « RH ») ou des **Fonctions** réutilisables par un Processus (Solution de Sécurité ou Solution de Tarification).
- Le **Modèle** est composé de logiciels, mais aussi d'informations et de procédures pour les utilisateurs.



2.2 Les Domaines Fonctionnels d'une Entreprise



Pour produire, vendre, gérer ses ressources ou piloter, les Acteurs de l'Entreprise exécutent des **Processus Opérationnels**.
Comme ces Processus sont très nombreux, on les regroupe dans des **Domaines Fonctionnels** : gérer les ressources humaines, comptabiliser, distribuer, gérer des partenaires.. .
Ces Domaines Fonctionnels sont structurés hiérarchiquement ; le plus haut niveau est constitué de :

- **Domaines Primaires** : produire et distribuer ; ces Processus sont généralement assez diversifiés selon les types de **produits** ou de clientèle, ce sont les plus nombreux.
- **Domaines de gestion de Ressources** : gérer le personnel, gérer les partenaires, gérer l'informatique, gérer les locaux, gérer la finance
- **Domaine de pilotage** : suivre l'activité, comparer au budget, produire des statistiques…

Dans l'industrie, les Processus Primaires utilisent des machines-outils, des chaines de production.
Dans les Secteurs qui ne traitent que d'information comme l'Administration, l'Assurance ou la Banque, les Processus Primaires sont exécutés par les outils informatiques. C'est ce qui explique que parmi les sponsors du CEISAR qui représentent de grandes Entreprises d'environ 100.000 à 150.000 employés on trouve des effectifs informatiques extrêmement différents : 2000 chez Michelin, 4000 chez Total, 15.000 à BNP-Paribas et 15.000 chez Axa.
Une grande partie des Processus informatisés dans l'industrie sont des Processus de gestion de ressources ou de pilotage, alors que la majorité des Processus informatisés dans le service sont des Processus Primaires.

Les Processus de pilotage (ou décisionnel ou Business intelligence) sont toujours difficile à classifier dans le mesure où l'on peut identifier

- des Processus de pilotage Primaires tels que suivi des ventes que l'on peut aussi ranger avec les Processus opérationnels
- des Processus de pilotage des Ressources tels que suivi des coûts que l'on peut ranger avec les Processus de gestion de ressources
- des Processus de pilotage de synthèse tels que comptabilité générale que l'on peut isoler

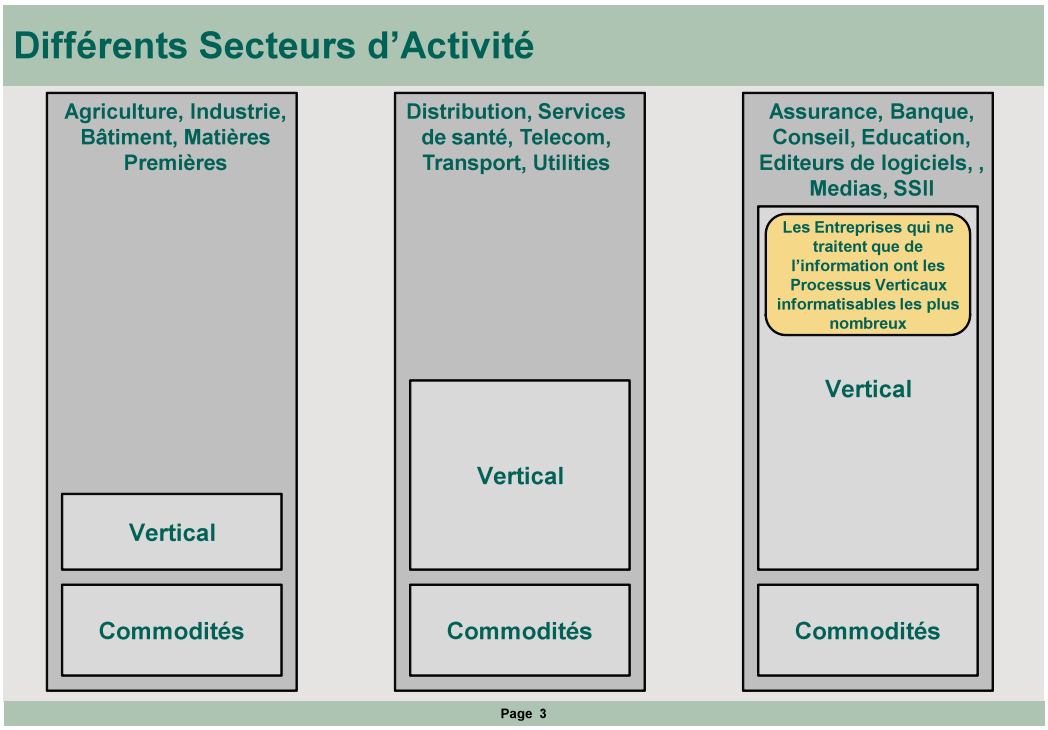
En outre, les évolutions technologiques qui permettent de mettre en mémoire des bases de données importantes vont gommer la différenciation entre base de données opérationnelle et base de données décisionnelle. La tendance sera donc à ne conserver dans les Processus de pilotage que le pilotage de synthèse.

Il existe d'autres façons de décomposer les Processus que nous utiliserons dans la suite de ce document :

- Front-Office, Middle-Office, Back Office
- Commodité ou Vertical
- Opération ou Transformation

Processus de Front-Office, Middle-Office et Back-Office :
Une autre façon de décomposer les Processus est de distinguer les **Processus de Front-Office** qui sont exécutés avec le Client et les **Processus de Back-Office** qui ne sont pas exécutés avec le Client. Les Processus de Front Office passent généralement par des **canaux multiples**.
Certains distinguent aussi les **Processus de Middle-Office** pour nommer les Processus de Back Office qui prolongent les Processus initiés au Front office, que ce soit pour la Distribution ou la Production. Par exemple, la négociation avec le client est du Front-Office. L'enregistrement de la commande qui en résulte peut être effectuée en dehors de la présence du client : c'est du Middle-Office. Les Processus de gestion de Ressources restent du Back-Office.
Cette distinction est intéressante parce que les Processus de Back-Office peuvent être similaires pour des Entreprises très différentes (la gestion des ressources humaines a des points communs entre différents secteurs d'activité), alors que les Processus de Middle-Office sont propres à chaque industrie (on ne distribue pas de la même façon des billets de train ou des abonnements télécoms).

Processus de Commodité ou Processus Verticaux
Les Processus de Commodité sont similaires d'un secteur à l'autre donc peu différenciant (Processus de gestion comptable, Processus de gestion des locaux, Processus de gestion de la paye…), alors que les Processus Verticaux sont propres à chaque industrie et source d'avantage concurrentiel (Processus de distribution ou de production de chaque industrie) : on sera donc amené à différencier des Solutions de Commodité et des Solutions Verticales.



Selon le secteur d'activité, le Vertical a un poids différent.

Processus Opérationnels et Processus de Transformation
A coté de ces Processus Opérationnels existent des **Processus de Transformation** qui servent à créer de **nouveaux Modèles** Opérationnels ou de nouveaux Modèles de Produits, et à **adapter les Ressources** pour qu'elles puissent exécuter ces nouveaux Modèles, ce que l'on appelle généralement « **Déploiement** ».

2.3 Qu'est ce qu'un Progiciel ?

2.3.1 Le Progiciel

Définitions existantes
D'après **Wikipedia** (*http://fr.wikipedia.org/wiki/Progiciel*) : « *Un **Progiciel,** contraction de **pro**duit et lo**giciel**, est un logiciel applicatif commercial « prêt-à-porter », standardisé et générique, prévu pour répondre à des besoins ordinaires. Ce terme s'oppose aux « logiciels sur mesure » développés en interne par une Entreprise et conçus pour répondre à des besoins spécifiques*. »

En 1979, la **Commission de terminologie de l'informatique** créée par le ministre français de l'Industrie a donné la définition suivante : « Un ensemble complet et documenté de programmes conçus pour être utilisés par différents utilisateurs et capable de remplir la même application ou fonction. »

*Le **Dictionnaire de l'informatique*** donne la définition suivante: « Un ensemble cohérent et indépendant constitué de programmes, de services, de supports de manipulation d'informations (bordereaux, langages, etc.) et d'une documentation, conçu pour réaliser des traitements informatiques standard, dont la diffusion revêt un caractère commercial et qu'un utilisateur peut utiliser de façon autonome après une mise en place et une formation complète[4]. »

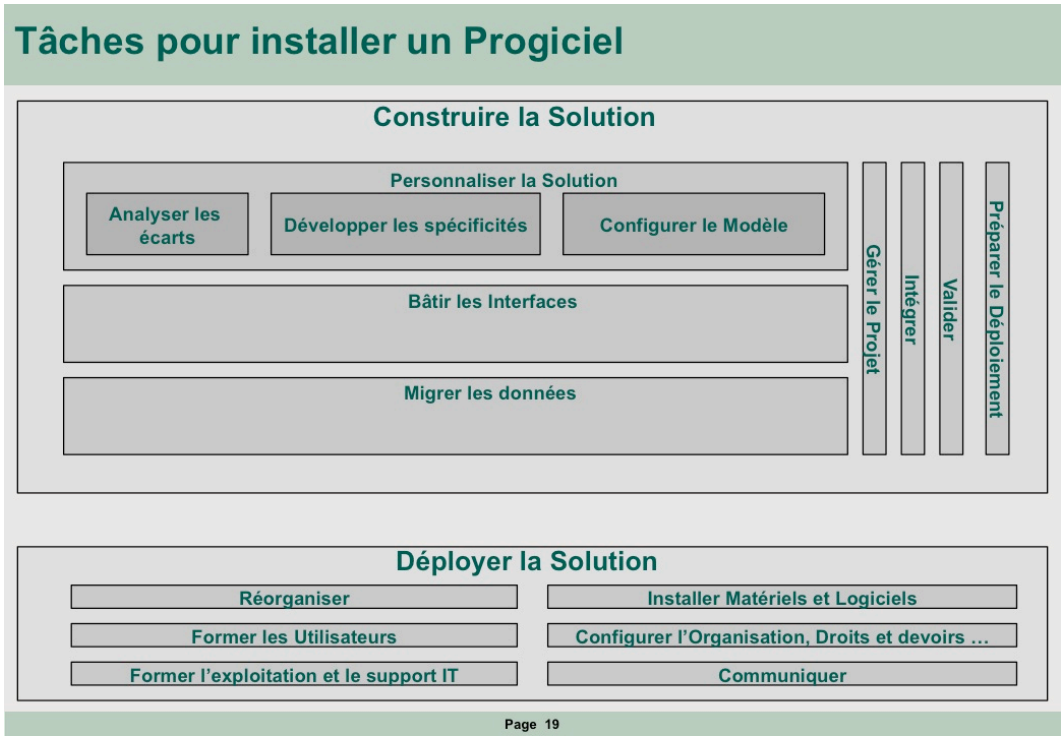
Définition proposée
Un **Progiciel** est une Solution développée, supportée et commercialisée par un **Editeur de Logiciel**. Elle s'oppose donc aux **Solutions Spécifiques** développées en interne : elle est destinée à plusieurs clients.
Le Modèle d'Opération d'une Entreprise est donc constitué d'un ensemble de Solutions Spécifiques et de Progiciels qui coexistent et dont le pourcentage respectif dépend de la politique de l'Entreprise.

Note : il existe des synonymes du mot Progiciel, d'origine anglophone, comme « Package », et des synonymes du Progiciel-Solution comme « Application Package » ou « Commercial Package » (aussi appelé « Commercial Off The Shelf ou COTS, signifiant « logiciel commercial sur étagère »).

2.3.2 Le Cycle de vie du Progiciel

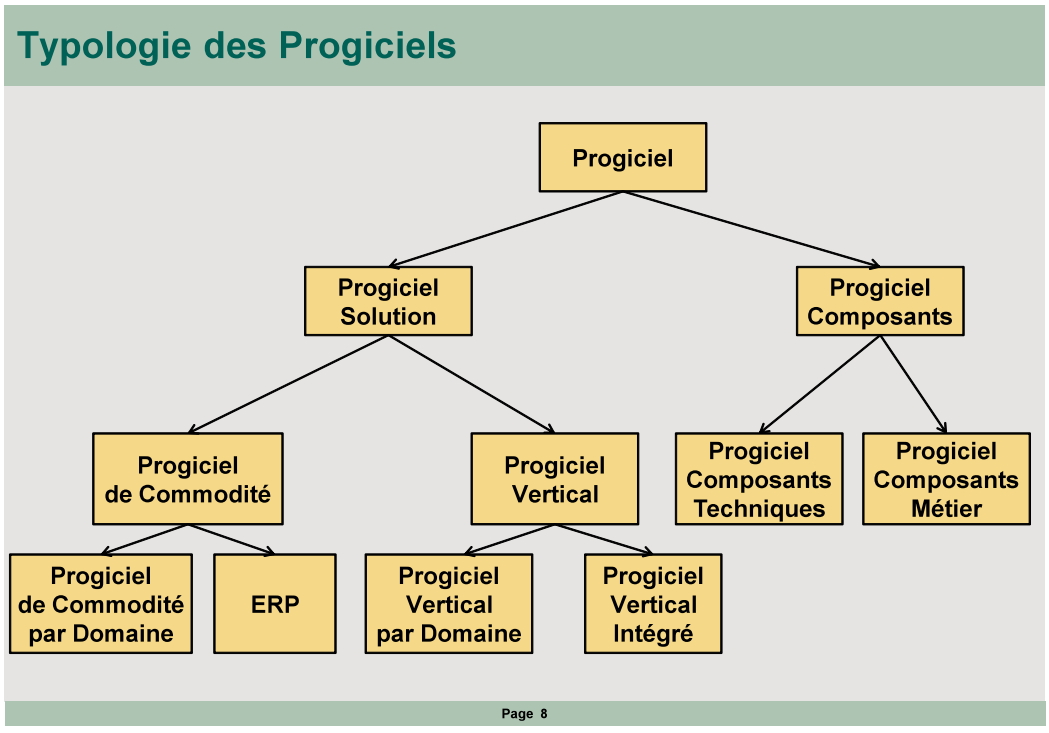
On distingue
- le **Progiciel** livré par l'éditeur
- la **Solution** qui s'exécute chez le Client qui est la somme du Progiciel et de la partie personnalisée pour le client.

Le Progiciel est livré sous forme de **versions** successives : généralement tous les six mois à deux ans.
L'adaptation de la Solution à la nouvelle version du Progiciel s'appelle « **montée de version** ».
Installer un Progiciel est l'action qui consiste à
- **personnaliser** le Progiciel en fonction du client : que ce soit
  - par **configuration,** c'est-à-dire en utilisant du paramétrage ou un moteur de règles
  - par **développement** complémentaire : spécifique ou communautaire
- construire les **Interfaces** avec les autres Solutions en place, que ces Solutions soient elles mêmes des Progiciels ou des Solutions spécifiques développées en interne
- **migrer** les informations de l'ancienne Solution au Progiciel
- **déployer** le Progiciel (ou « gérer le changement »)
  - réorganiser : structure et Processus
  - former les utilisateurs
  - former l'exploitation informatique et le support aux utilisateurs
  - installer matériel et logiciel



Dans les grands groupes qui choisissent un même Progiciel pour différentes filiales on passe souvent par un intermédiaire : le **Progiciel-matrice** est la somme du Progiciel fourni par l'Editeur et d'un premier niveau de personnalisation commun à toutes les filiales
Une bonne façon de faire converger les Solutions des filiales est de regrouper leur exploitation dans un même centre.

2.4 Typologie des Progiciels



On distingue Progiciel-Solution et Progiciel-Composant.

Un **Progiciel-Solution** modélise un ensemble de Processus d'un Domaine Fonctionnel : Progiciel de CRM, Progiciel Comptable, Progiciel de « Supply chain ».

Un **Progiciel-Composant** modélise des Fonctions réutilisables par différents Processus aussi bien au niveau technique qu'au niveau métier: fonctions techniques telles que EAI, système de gestion de données, middleware, fonctions d'éditique, fonctions de sécurité, fonctions de GED, fonctions de traitement de texte, fonctions de datawarehouse, Services-logiciels utilisables dans une approche SOA qui peuvent être utilisés par les autres Solutions.

Les Progiciels-Solutions sont décomposés en **Progiciels de Commodité** (Progiciel de comptabilité, de gestion de projet, de gestion de ressources humaines, de messagerie, de pilotage) et **Progiciels Verticaux** offerts à chaque secteur économique (Progiciel pour le secteur bancaire, Progiciel pour le secteur des télécoms, Progiciel pour le secteur de la distribution).

Les Progiciels de Commodité sont offert pour **chaque Domaine Fonctionnel** (Progiciel de paye, Progiciel de Comptabilité) ou intègrent différents Domaines Fonctionnels dans un Modèle unique que l'on appelle **ERP** (ou le synonyme français que nous n'utiliserons pas : PGI pour Progiciel de Gestion Intégré).

De la même façon les Progiciels Verticaux sont offerts pour des Domaines Fonctionnels distincts (par exemple pour l'assurance : Progiciel de gestion des contrats auto, Progiciel de gestion de contrat vie, Progiciel de gestion de sinistre auto…) ou intègrent différents Domaines dans un Modèle unique ; on parle alors de **Vertical Intégré**.

2.5 Caractéristiques des Progiciels

2.5.1 Couplage fort ou couplage faible

Si un Progiciel fonctionne indépendamment des autres Solutions de l'Entreprise, il a un couplage faible.
Si un Progiciel interopère avec de nombreuses autres Solutions, il a un couplage fort.
Il y a bien sûr tous les degrés intermédiaires.

L'interopérabilité peut être :

- **synchrone** (c'est l'approche SOA) : le Progiciel offre des Services appelables par les autres Solutions ou le Progiciel appelle des Services offerts par les autres Solutions.
- ou **asynchrone** : des flux sont échangés entre Solutions aussi bien en entrée qu'en sortie (le Progiciel comptable reçoit des flux d'écritures comptables, le Progiciel de pilotage reçoit des flux d'évènements) ; une distinction plus fine permet de distinguer
    - le transfert de fichier
    - le transfert d'évènements au fil de l'eau

L'utilisation de Services synchrones a pour avantages :

- pour les données :
    - de ne pas répliquer les données dans différentes Solutions
    - d'offrir la même fraicheur d'information à toutes les Solutions
- pour la logique
    - d'alléger le Modèle de la Solution appelante qui utilise le Modèle du Service appelé
    - de garantir que chacun utilise la même version de logique (par exemple un Service de tarification réutilisé par des Solutions qui correspondent à des canaux de distribution différents).

Par contre, elle nécessite

- de définir des **Interfaces** d'appel les plus stables et les plus génériques possibles
- de **modifier les Solutions appelantes** pour qu'elles tirent parti des Services disponibles
- de bien **synchroniser** les évolutions de Service
- d'utiliser des **mécanismes d'échanges** compatibles (tels que Web Services et XML)

Les Solutions deviennent alors plus **imbriquées**.

2.5.2 Périmètre national ou international

L'homogénéité des Solutions entre pays permet :

- d'appliquer une stratégie de croissance sans ruptures
- d'offrir des services mondiaux aux Clients
- d'échanger les produits entre pays
- d'échanger les bons Processus entre pays
- de créer des centres de services partagés, internes ou externes (« BPO »)
- de réduire les coûts de Transformation : les effectifs d'étude diminuent
- de mettre plus facilement en commun l'exploitation des centres informatiques

Un Progiciel est international s'il peut être utilisé dans plusieurs pays.

- Pour certains, il ne s'agit que d'offrir des mécanismes multilangues, multidevises ou de changer certains paramétrages (comme les villes et codes postaux)
- Pour d'autres il faut aussi personnaliser des Fonctions propres à chaque pays : fiscalité, réglementation, modes de distribution…

Les Progiciels de Commodité sont aujourd'hui essentiellement internationaux.

Dans l'industrie, les Progiciels Verticaux sont rapidement devenus internationaux (Ex : SAP chimie).

Dans le service, les Progiciels Verticaux ont été essentiellement nationaux, mais le rapprochement des cultures et des marchés, la nécessité de faire des économies d'échelle dans les grands groupes, précipitent la convergence de ces Progiciels Verticaux.

2.5.3 Progiciel personnalisable ou non

Certains Progiciels ne sont **pas personnalisables** : voir les exemples de Progiciels proposés par l'Apple-store. Mais la majorité des Progiciels sont personnalisables : soit par configuration, soit par développement complémentaire, soit par les deux.
Certains Progiciels sont **personnalisables par configuration :** on peut paramétrer le Progiciel ou utiliser un Moteur de règles dans un cadre précis qui permet de bien isoler la personnalisation, sans qu'elle perturbe la bonne Architecture de la Solution. C'est le cas de la majorité des Progiciels de Commodité tels que traitement de texte, Progiciel Comptable…
Certains Progiciels sont **personnalisables par développement complémentaire** dans l'environnement de Transformation du Progiciel: c'est le cas de Progiciels Verticaux. Ce développement complémentaire peut être **communautaire** (pris en compte par l'éditeur dans le cadre d'une future version de son Progiciel) ou **spécifique** à un client.
La tendance actuelle consiste à enrichir les possibilités de configuration pour réduire les développements complémentaires au minimum.

2.5.4 Progiciel avec ou sans données de configuration

Un Progiciel est avant tout un Modèle : logiciels et Modèles de Processus. Comme un Progiciel importe souvent une organisation sous-jacente, on dit qu'il est « structurant » pour l'Entreprise.
Pour accélérer l'intégration du Progiciel, certains éditeurs fournissent aussi le Progiciel « rempli » : ils proposent une configuration standard qui représente les Processus, les produits, les données de base et les modes d'organisation les plus courants. Le client travaille alors par différence et ne modifie que ce qu'il souhaite sans passer par tout le Processus de configuration.

2.5.5 Licence ou facturation à l'usage : le Cloud ?

Un Client peut acquérir une **licence** du Progiciel pour un périmètre d'utilisation donné.
Il souscrit en outre un contrat de **maintenance** s'il veut bénéficier des corrections et des évolutions contenues dans les futures versions.

Un autre mode se développe qui permet au client de payer à l'usage. C'est ce que l'on observe dans l'ASP ou le **Cloud**.

Les Progiciels peuvent être exécutés sous « SaaS » qui virtualise les ressources exécutant le Progiciel, en général pour plusieurs clients (Cloud Public), et parfois pour un seul client (Cloud Privé). La généralisation d'Internet permet d'utiliser le Progiciel sans avoir besoin de l'installer sur ses propres machines.
Les intérêts sont évidents : le coût est lié à l'usage, donc à la valeur que l'on en retire, on peut faire face à des pointes de consommation, on n'a pas à gérer l'exploitation informatique.

Par contre, la maîtrise des interfaces entre Progiciels exécutés dans des « Clouds » différents est problématique.
Il en est de même entre un Progiciel exécuté dans le Cloud et des Solutions externes qui s'exécutent dans des environnements classiques.

2.6 Typologie des interfaces entre progiciel et autres Solutions

Remarque :
Le terme « Interface » est parfois ambigu : il signifie « Modèle d'échange entre 2 Solutions », mais il peut aussi signifier « Modèle d'usage de la Solution ».
*Pour lever l'ambiguïté, nous réservons « **Interface** » pour le **Modèle d'échange entre Solutions** et nous utilisons « Mode d'usage » ou « Interface Utilisateur » pour le Modèle qui définit les interactions entre l'utilisateur et la Solution.*

Une **Solution** est un logiciel construit sur la même Fondation : mêmes outils de développements, mêmes composants techniques et métiers.
Les **relations internes** aux Solutions ne nécessitent pas d'Interfaces.
Le terme d'Interface est généralement réservé aux **échanges entre Solutions distinctes**.
Lorsque plusieurs instances de la même Solution doivent échanger, par exemple dans différents pays, elles peuvent avoir besoin d'interfaces, mais elles seront plus faciles à construire compte tenu de l'homogénéité de la source et de la destination.

Une Interface est **synchrone** si la Solution appelante attend une réponse de la Solution appelée ou **asynchrone** dans l'autre cas.
L'Asynchrone peut prendre deux formes : soit du **transfert de fichiers**, soit du « **fil de l'eau** » qui permet une meilleure fluidité et une meilleure fraîcheur d'information.

Une interface se décompose en deux demi-interfaces : l'une sur la Solution appelante, l'autre sur la Solution appelée.
La **demi-interface de la Solution appelante** s'exécute en trois étapes :
- préparer les données à transmettre
- convertir les données à transmettre au bon format
- envoyer

La **demi-interface de la Solution appelée** s'exécute en quatre étapes :
- recevoir
- convertir les données reçues au bon format
- contrôler les données reçues
- ranger les données reçues

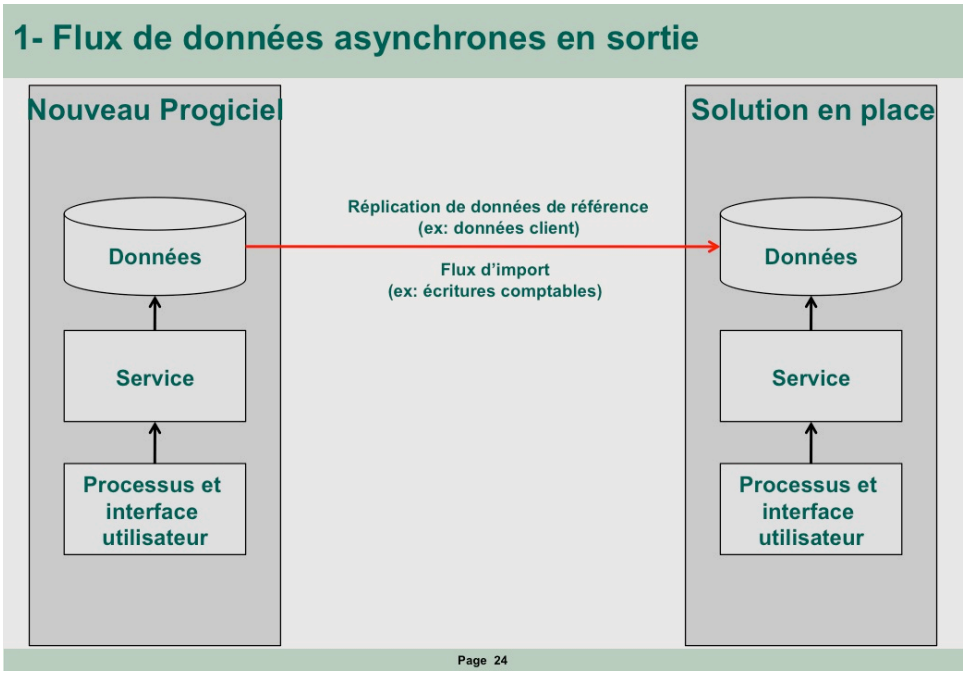
L'étape de contrôle peut s'avérer complexe : elle est nécessaire pour protéger la Solution appelée au cas où les données reçues ne sont pas de bonne qualité, ce qui explique pourquoi il est plus coûteux de construire la demi-interface côté Solution appelée.
La préparation et le rangement des données peuvent être complexes : il ne suffit pas de lire une table, il peut s'agir
- de rassembler les données dispersées d'un **même objet** métier (toutes données d'un même client),
- d'y adjoindre les **objets liés**,
- de transférer des **listes** d'objets
- de n'envoyer que les **données modifiées**
- de récupérer les données nécessaires à l'exécution d'un Service externe (comme calcul de tarif)

Il existe différentes formes d'Interfaces que nous allons maintenant décrire.

2.6.1 Le Progiciel envoie un flot de données asynchrone

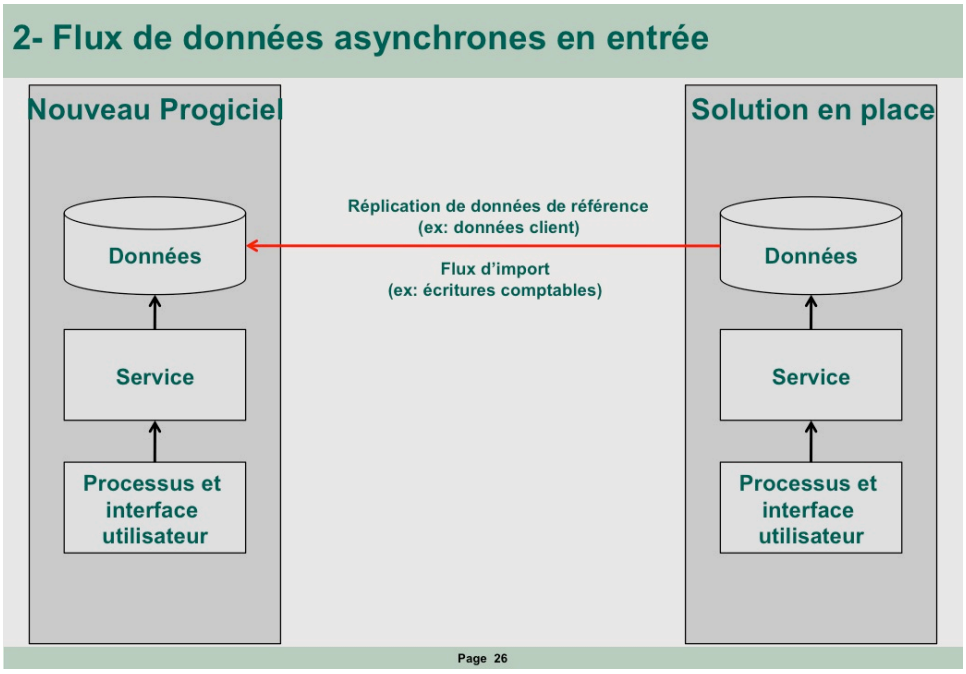
pour :

- répliquer des données dans la Solution Couplée
- générer des entrées destinées à une Solution Couplée



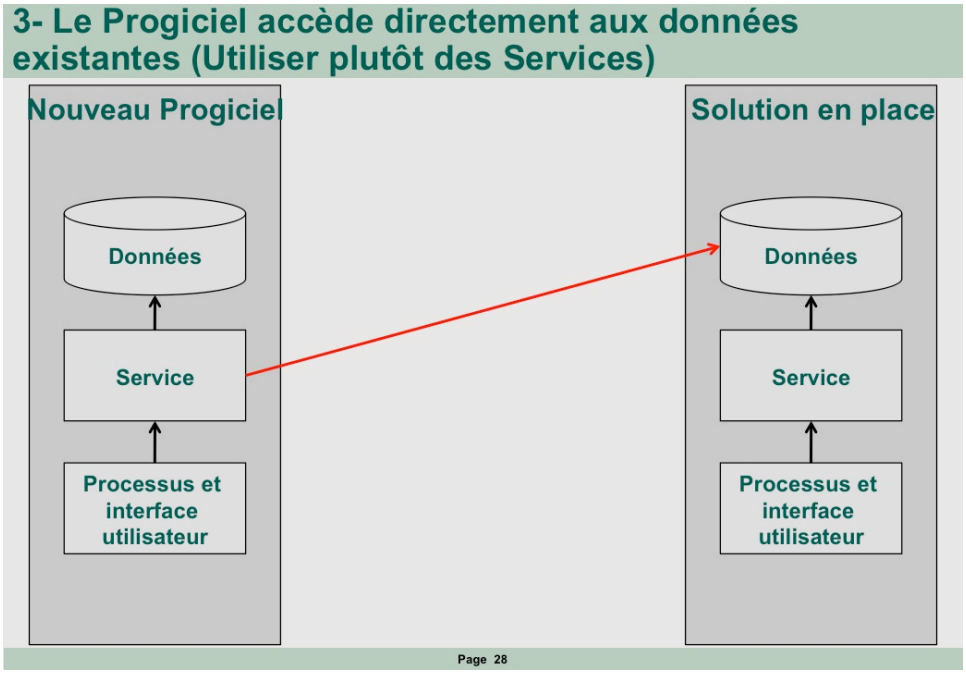
2.6.2 Le Progiciel reçoit un flot de données asynchrones

Beaucoup plus complexe que le cas précédent car le Progiciel doit contrôler les données du flux avant de les prendre en compte.
Attention, les flux entrants dans le Progiciel sont plus difficiles à gérer que les flux sortants : le Progiciel considère que les données transmises sont les bonnes, alors qu'il doit contrôler les données qui rentrent dans le Progiciel (ratio de 1 à 3 en difficulté)



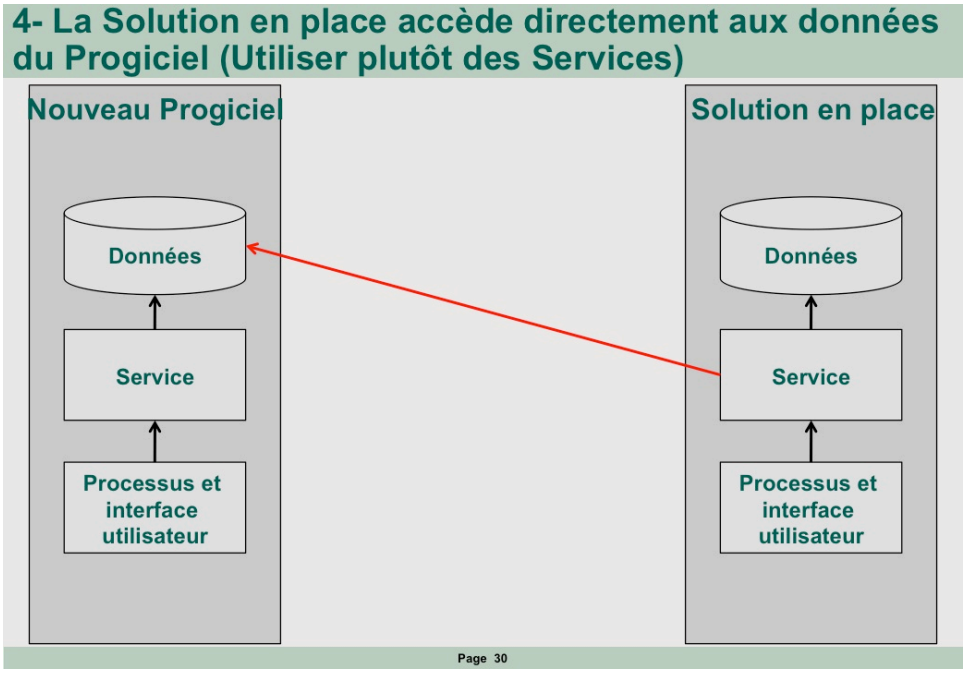
2.6.3 Le Progiciel accède à des Données appartenant à une Legacy Solution

Plutôt passer par des Services offerts par la Legacy Solution, s'ils existent, ce qui permet de désynchroniser l'évolution des données legacy des appels provenant du Progiciel.
Exemple : Progiciel de Product Factory accède aux informations décrivant les Produits dans les Legacy Solutions de Bouygues Télécoms.



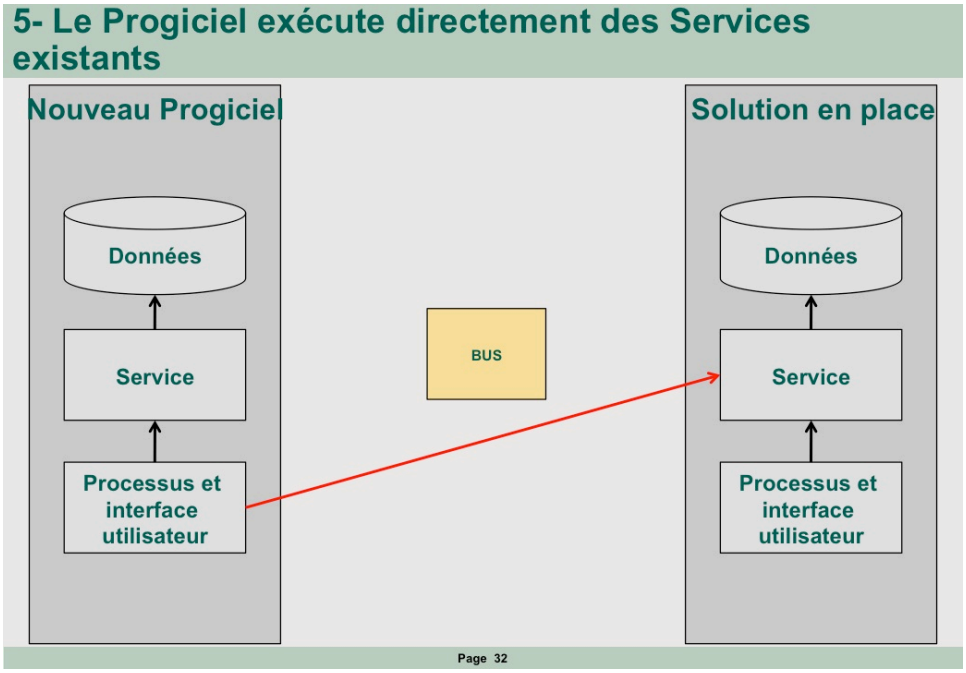
2.6.4 La Legacy Solution accède à des données du nouveau Progiciel

Là encore, il est préférable de passer par des Services.
Exemple : Infocentre Legacy qui souhaite accéder directement aux données du Progiciel



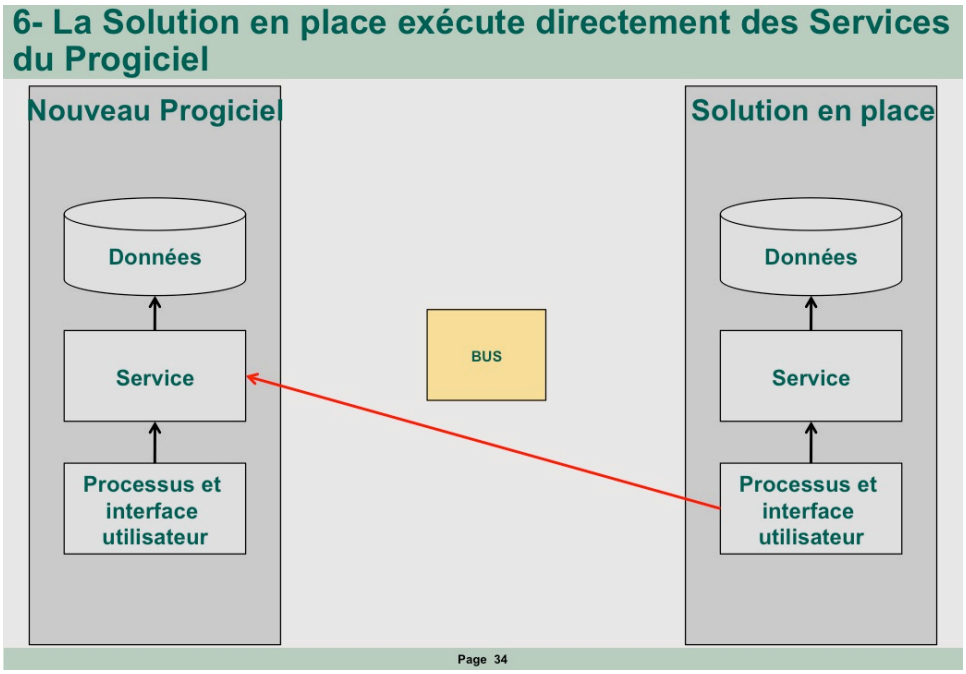
2.6.5 Le Progiciel appelle des Services offerts par une Legacy Solution

C'est une méthode propre souvent utilisée pour réutiliser des Services tels que authentification de l'utilisateur (LDAP), Services offerts par le CRM, l'Editique, la GED…



2.6.6 Le Progiciel offre des Services appelables par une Legacy Solution

C'est une solution propre pour réutiliser des Services offerts par le Progiciel tels que tarificateur, gestion de corbeille, création d'Entités Métier (client, contrat…).



2.6.7 Récapitulatif

Le tableau ci-dessous explique les points forts (en vert) et les points faibles (en orange ou rouge) de différentes approches possibles pour les échanges entre les Solutions :

- Solutions indépendantes sans échange
- Echanges asynchrones
- Echange au fil de l'eau (file de messages ou d'évènements)
- Echanges synchrones (appels de services)
- Fondation partagée : les Solutions échangent nativement grâce à une même Fondation de Construction

Les critères d'évaluation identifiés sont :

- L'homogénéité de **l'interface utilisateur** : si les modes d'usage des différentes Solutions sont homogènes, les utilisateurs adoptent plus facilement les nouvelles Solutions (coûts de formation moindres et meilleure acceptation), ils sont plus polyvalents et plus mobiles au sein de l'Entreprise.
- Possibilité d'offrir des **Produits transverses** : par exemple, dans le domaine de l'assurance, créer une offre qui combine deux produits existants (par exemple un produit d'épargne et un produit de prévoyance)
- Possibilité de créer des **Processus Transverses** : à partir d'un moteur de workflow interne ou externe, on peut créer plus ou moins facilement des Processus transverses à plusieurs Solutions
- **Duplication** d'information et ressaisies : pour les Solutions qui doivent échanger de l'information et qui ne sont pas interfacées, cela implique des duplications manuelles d'information donc une surcharge de travail et un risque d'incohérence des données
- **Fraîcheur** d'information : les différentes Solutions sont à jour plus ou moins en temps réel
- **Coûts d'exploitation** de la Solution
- Complexité de la Solution et **coûts projet**
- **Agilité** et time to market : capacité maximale à évoluer vite et bien
- **Synchronisation** des mises à jour
- **Cohérence** du modèle d'information : garantir que toutes les Solutions portent la même définition des Informations
- Niveau **d'autonomie** de chaque domaine fonctionnel : soit le domaine est libre de ses choix et peut adopter la Solution qui couvre le mieux ses exigences fonctionnelles (best of breed), soit il est contraint par l'existence d'une Fondation qu'il doit prendre en compte
- Nécessité d'avoir une **Fondation** ou non (et les investissements nécessaires associés)
- Nécessité d'avoir une **équipe-Fondation** pour développer et supporter la Fondation : en fonction de la nécessité, l'équipe peut ne pas exister, être limitée ou être importante.

Type d'échange entre Solutions	**Pas d'échange**	**Asynchrone**		**Synchrone**	**Même Fondation**
	Solutions indépendantes	Transfert de fichier	Au fil de l'eau	Appel de service	Echanges au sein de la Solution
Homogénéité de l'interface utilisateur	Non	Non	Non	Non	Oui
Possibilité d'offrir des Produits transverses	Non	Non	Non	Non	Oui
Possibilité de créer des Processus transverses	Difficile	Difficile	Difficile	Difficile	Possible
Duplication d'information et double saisie	Oui	Non	Non	Non	Non
Fraîcheur d'information	Non	Non	Moyenne	Oui	Oui
Coûts d'exploitation IT	Elevés	Elevés		Elevés	Faible
Taille du modèle et coût de Transformation	Elevés	Elevés	Elevés	Moyenne	Réduits
Agilité, time to market	Non	Non	Non	Non	Oui
Synchronisation des services	Non	Non	Non	Oui	Oui
Cohérence du Modèle d'Information	Non	Non	Non	Non	Oui
Niveau d'autonomie de chaque domaine fonctionnel	Best of Breed	Best of Breed	Best of Breed	Chaque Solution propose des Services	Gouvernance forte
Nécessité de la Fondation	Non	Non	Non	Non	Oui
Nécessité d'une équipe Fondation	Non	Faible	Faible	Moyenne	Oui

3 Evolutions de la demande des Entreprises

A travers les étu**d**es de cas nous avons cherché à comprendre quelles étaient les attentes de fond du marché.

RÉSUMÉ

- les Entreprises doivent progressivement **renouveler** leur parc de Solutions et en créer de nouvelles
- les Entreprises font un **appel croissant aux Progiciels** aux dépens des développements internes
- les Entreprises ne veulent plus de Solutions isolées (l'approche « silo ») : elles doivent s'interconnecter et **échanger**
- les Entreprises recherchent non seulement des Progiciels qui informatisent des Processus globalement mais aussi des Progiciels qui informatisent des Fonctions sous forme de **Services-Logiciels** utilisables par leurs propres Solutions
- les Entreprises recherchent des Solutions à **périmètre plus large** pour réduire le nombre de Solutions et la complexité globale : on recherche des ERP plutôt que des Progiciels de Commodité indépendants, on recherche des Verticaux Intégrés plutôt que des Verticaux par Domaine Fonctionnel, on recherche des offres de Services-Logiciels groupés plutôt que des Services indépendants
- Après les Progiciels de Commodité, les Entreprises recherchent des **Progiciels Verticaux** qui peuvent leur donner un avantage concurrentiel
- Pour accélérer l'installation d'un Progiciel et permettre au Métier d'adapter directement le Progiciel, les Entreprises souhaitent utiliser des **mécanismes de configuration puissants** plutôt que des développements spécifiques.
- Les Entreprises **internationales** cherchent à développer des **synergies** pour économies d'échelle, offrir des produits internationaux, gérer des clients internationaux et transférer de bonnes pratiques ou de bons produits entre pays.
- Les Entreprises souhaitent mieux **comprendre l'Architecture du Progiciel** pour juger des possibilités de personnalisation, de sa capacité d'intégration avec les autres Solutions et de son évolutivité.

3.1 Les Entreprises doivent renouveler et enrichir leur parc de Solutions

Le parc informatique d'une Entreprise est composé d'une multitude de Solutions progressivement ajoutées au fur et à mesure de l'informatisation de nouveaux domaines.
On retrouve généralement un mix de Solutions vieillissantes et de Solutions plus modernes qui coexistent tant bien que mal.
Les Entreprises doivent faire évoluer ce parc pour différentes raisons :

- Informatiser des **domaines fonctionnels** qui ne l'étaient pas.
- Accompagner des **initiatives mé**tier : ouverture d'un nouveau pays, sortie d'une nouvelle gamme de produits, gestion d'un nouveau réseau de distributeurs.
- Remplacer des **Solutions anciennes** qui ont des difficultés à évoluer ou ne sont même plus maintenables, La lourdeur d'évolution des Solutions anciennes se traduit par le fait que pour une dépense de 10€, on dépense 1€ en évolutions fonctionnelles et 9€ en interfaçage avec les autres Solutions, en qualification ou tests de non régression.
- **Réduire la complexité d'ensemble** de leur parc : plus le parc de Solutions est foisonnant, plus il est coûteux de le faire évoluer. A titre d'exemple, La division Fret du groupe Air France a dû procéder à la refonte de ses Solutions de CRM afin d'automatiser le nouveau Processus métier défini dans le cadre de la fusion Air France / KLM / Martinair tout en réduisant les redondances sur ce domaine entre les applications issues des différentes Entreprises composant le Groupe.

Cette informatisation croissante permet aux ordinateurs de remplacer les Acteurs-Humains dans les tâches simples, ce qui permet de les libérer progressivement pour des tâches à valeur ajoutée plus

importante. On parle de « gestion par exception » : les ordinateurs gèrent les cas courants, les Acteurs-Humains gèrent les exceptions qui sont les cas complexes.

3.2 Les Entreprises demandent de plus en plus de Progiciels

Compte tenu de la croissance des Domaines informatisés et de la complexité croissante des métiers, il devient impossible de faire face à la demande croissante d'informatisation par des développements spécifiques : les Entreprises doivent recourir de plus en plus fréquemment à des **Progiciels** pour informatiser leurs activités. Même les grandes Entreprises qui se différencient par le système d'information (banque, assurance, utilities, télécoms…), et qui disposent d'équipes informatiques très importantes, font désormais un appel croissant aux Progiciels.

Les avantages annoncés sont nombreux.

- **Récupération** de Processus et de Fonctions métier déjà formalisés : pas besoin de bâtir en interne un cahier des charges complet
- **Accélération** du projet et **réduction des coûts** de développement, par la mise en place d'une Solution déjà existante
- **Concentration** des capacités internes de développement sur le cœur de métier lorsque l'on utilise des Progiciels pour les Solutions de Commodité
- Gestion du **cycle de vie du logiciel** externalisé chez l'éditeur, un professionnel de la conception et de la maintenance de solutions informatiques
- **Echanges standardisés** avec d'autres clients de l'éditeur

Mais l'introduction d'un Progiciel est aussi un bon moyen pour entrer rapidement dans un **nouveau métier.**
A ce titre, l'exemple de RCI-Assurance est intéressant. RCI Banque, filiale de Renault, propose des crédits auto aux clients de Renault, Nissan et Samsung Motors. Un tiers des ventes de véhicules sont ainsi financées. Chaque crédit auto s'accompagne d'une assurance-crédit fournie par des partenaires assurance dans chaque pays. Renault a décidé de créer RCI Assurance pour reprendre cette activité. Ne connaissant pas le métier de l'assurance, il était impossible de définir des spécifications détaillées pour développement interne. L'acquisition d'un Progiciel est alors le bon moyen non seulement pour se doter d'une Solution informatique, mais aussi pour acquérir la compétence métier-assurance.

Quelles limites à l'usage des Progiciels ?

Autrement dit, une Entreprise peut-elle constituer son Système d'Information uniquement avec des Progiciels ?
Concevoir une Solution pour plusieurs clients est plus difficile que concevoir une Solution pour un seul client : à ce titre, les Progiciels sont plus difficiles à construire que les Solutions internes. Mais la pression des clients, la concurrence entre Editeurs font émerger progressivement des Progiciels de plus en plus mûrs. Progressivement les Progiciels vont donc se substituer aux Solutions spécifiques pour finir par les remplacer, sauf dans des domaines :

- où il n'existe **pas suffisamment de clients potentiels** pour justifier la construction d'un progiciel ; c'est le cas de l'Administration qui peut utiliser des progiciels de Commodité mais qui est obligée de construire ses propres Solutions primaires, tant que les administrations de différents Etats définissent des lois et réglementations spécifiques
- où **l'avantage concurrentiel** est prépondérant : par exemple, les Solutions d'exploitation pétrolière (géosciences …) sont considérées comme des Solutions spécifiques par l'industrie du pétrole

3.3 Les Entreprises demandent non seulement des Progiciels-Solutions mais aussi des Progiciels-Composants

Les Entreprises les plus matures souhaitent homogénéiser le mode d'usage (interface utilisateur). Plutôt que d'utiliser un nouveau Progiciel qui inclut une nouvelle interface utilisateur à laquelle ils vont devoir former leur personnel, ils préfèrent conserver leur interface utilisateur et faire appel à des Progiciels-Composants qui offrent des Services-logiciels appelables par l'interface utilisateur en place.

Point de vue du CEISAR

Homogénéiser l'interface utilisateur est un moyen efficace pour diminuer la complexité des Solutions vue par l'utilisateur final. Cela permet de réduire les charges de formation et de mutation entre les départements de l'Entreprise et contribuer à une politique de mobilité ; cela permet aussi de rendre les Acteurs plus polyvalents ce qui contribue à l'intérêt des employés et à la productivité de l'Entreprise. Pour cela, on doit maitriser la séparation des couches entre l'interface utilisateur et la logique métier. L'Editeur de Progiciels doit fournir des Services-Logiciels. L'Entreprise doit se doter d'un environnement pour créer l'interface utilisateur commune. L'ouverture du Progiciel sous forme de Services-Logiciels peut donc entrer dans la liste des critères de choix importants.

Une de nos étude de cas a démontré l'apport d'une offre de Composants-techniques intégrés pour construire des Solutions spécifiques et les interfacer à d'autres Solutions: Stallergènes a reconstruit en 2006 son Processus de gestion des demandes d'achat « hors production » et des demandes d'investissements en s'appuyant sur l'offre de W4. Le Processus a été conçu et développé dans le moteur de workflow de W4 et s'interface avec différentes Solutions dont SAP (qui fournit le référentiel des fournisseurs et des groupes de marchandises) par l'intermédiaire d'une Interface standard à SAP (fournie par iBolt). L'interface utilisateur est simple, conviviale et adaptée aux spécificités de l'Entreprise. Cette Solution s'interface non seulement à SAP mais aussi à la Solution-GMAO.

3.4 Les Entreprises ne veulent plus de Solutions isolées

Une adjonction de Progiciels indépendants conduisent à un Modèle **complexe** qui engendre des coûts de possession globaux sur le long terme mal maîtrisés et une capacité limitée à évoluer alors que les Acteurs métiers expriment des exigences **d'agilité** toujours plus grandes et cherchent à concevoir des Processus métiers hautement adaptables et configurables.

Les Entreprises souhaitent **éviter de ressaisir** les informations dans différentes Solutions.
Elles souhaitent **partager des référentiels** communs.
Elles souhaitent aussi **partager des services-logiciels** communs.

Elles mettent donc en place **l'interopérabilité entre Solutions** : les Progiciels Intégrés et isolés sont remplacés par des Solutions plus ouvertes. Les Entreprises font pression sur les Editeurs de Progiciels pour qu'ils ouvrent leurs Progiciels.

Certaines (une minorité encore aujourd'hui comme les opérateurs de téléphonie) souhaitent aussi automatiser des **Processus de bout en bout** qui sont traités par différentes unités d'organisation pourvues de Solutions différentes.

Point de vue du CEISAR

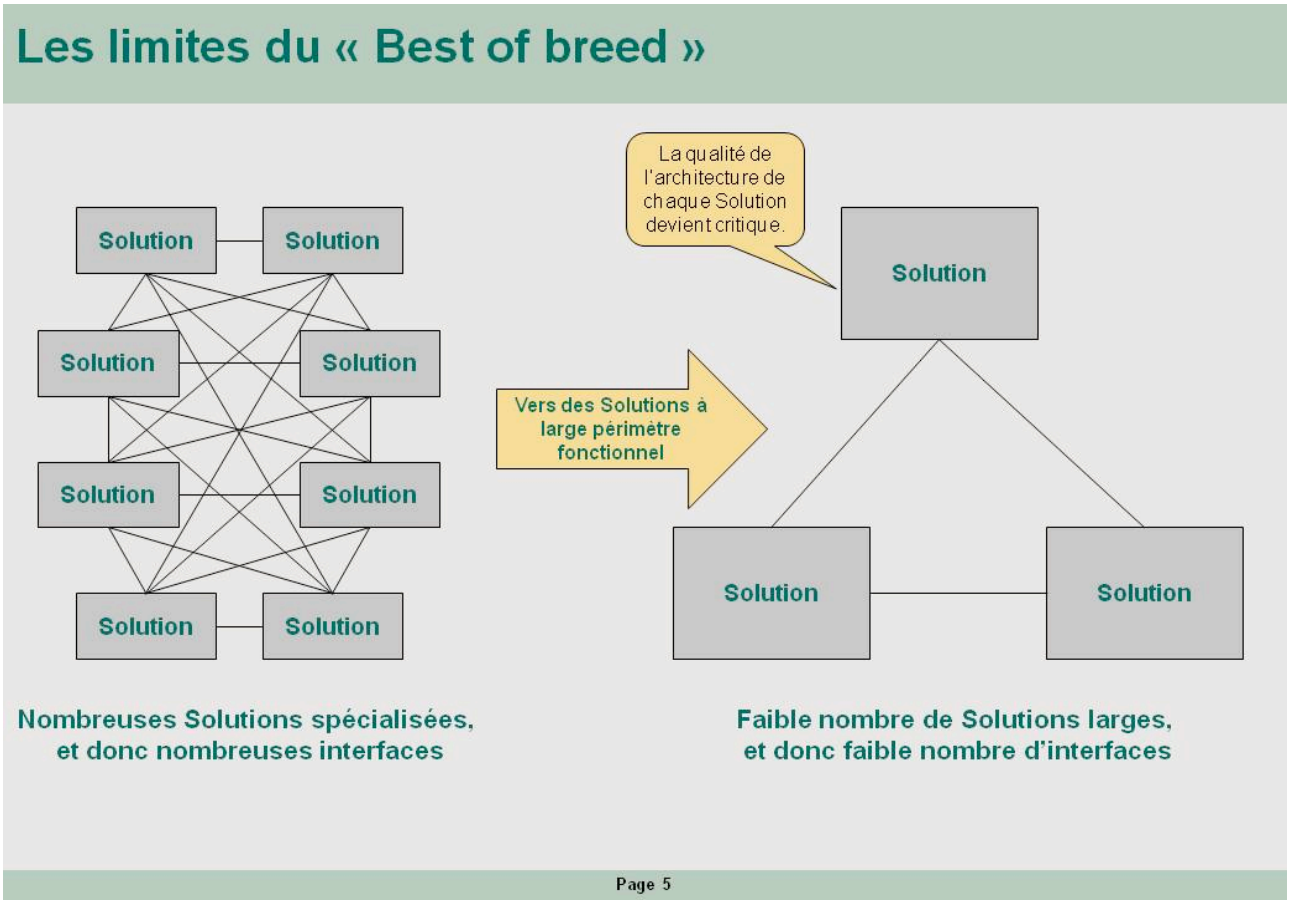
Cet appel à davantage d'interopérabilité se traduit par le développement d'Architectures orientées services (SOA) dans de plus en plus d'Entreprises. Ces dernières se dotent de Services-Logiciels à partir de Progiciels-Composants ou de développements spécifiques ou de Services ouverts des Progiciels-Solutions. L'ensemble des Services-Logiciels est géré dans un référentiel de Services. Ce portefeuille est complexe à gérer et pose la question de la responsabilité du bon fonctionnement du système complet ainsi construit. Notre conviction est que les Entreprises auront tendance à réduire progressivement le nombre de fournisseurs de Services-Logiciels, à s'appuyer de manière croissante sur des offres de Services groupés plutôt que par des Services unitaires de sources disparates qu'il faut ensuite intégrer.

3.5 Les Entreprises recherchent des Solutions à périmètre plus large

L'interopérabilité est difficile à gérer lorsque les Solutions sont nombreuses et évolutives.
Axa France estime faire un énorme travail de « plomberie » à chaque modification fonctionnelle, compte tenu de la multitude de Solutions en place qui sont le résultat des fusions et rachats successifs.
Axa France comme la majorité des grandes Entreprises du secteur tertiaire, recherche des **Solutions à périmètre large** pour réduire le nombre de Solutions, ce qui a pour effet de :

- **réduire le nombre d'interfaces** entre Solutions : l'éditeur a déjà interrelié des fonctionnalités à l'intérieur de sa Solution, ce qui limite le travail d'intégration postérieur de l'Entreprise
- de **simplifier les évolutions** : coordinations plus simples, gestion du logiciel plus simple
- d'offrir une **interface utilisateur** moins hétéroclite : permet davantage de polyvalence des collaborateurs qui peuvent passer d'une application à une autre plus facilement

Les exigences sont alors plus fortes vis-à-vis des Editeurs de logiciels : la couverture fonctionnelle du Progiciel est plus large et on a transféré une partie de l'Architecture interSolutions au sein du Progiciel à large périmètre. Il faut donc que son Architecture soit plus robuste pour bénéficier d'un rythme d'évolution régulier. L'Editeur doit être transparent vis-à-vis des Entreprises pour qu'elles puissent juger de la qualité de l'Architecture du Progiciel : il ne suffit pas de comprendre les fonctionnalités offertes, il faut aussi comprendre les grandes lignes de l'Architecture du progiciel.



Vaut-il mieux dépendre d'un nombre restreint de fournisseurs ou d'un nombre important de fournisseurs ?
A vrai dire, on dépend toujours de quelqu'un : soit des équipes internes, soit des éditeurs de Progiciels.
Si l'on reprend le schéma ci-dessus, il nous semble plus facile de gérer trois fournisseurs plutôt que huit.
Mais quelque soit le nombre de fournisseurs, la meilleure protection est liée au nombre de clients du Progiciel, et à la modernité de l'Architecture du Progiciel qu'il vaut mieux choisir au début de son cycle de vie.

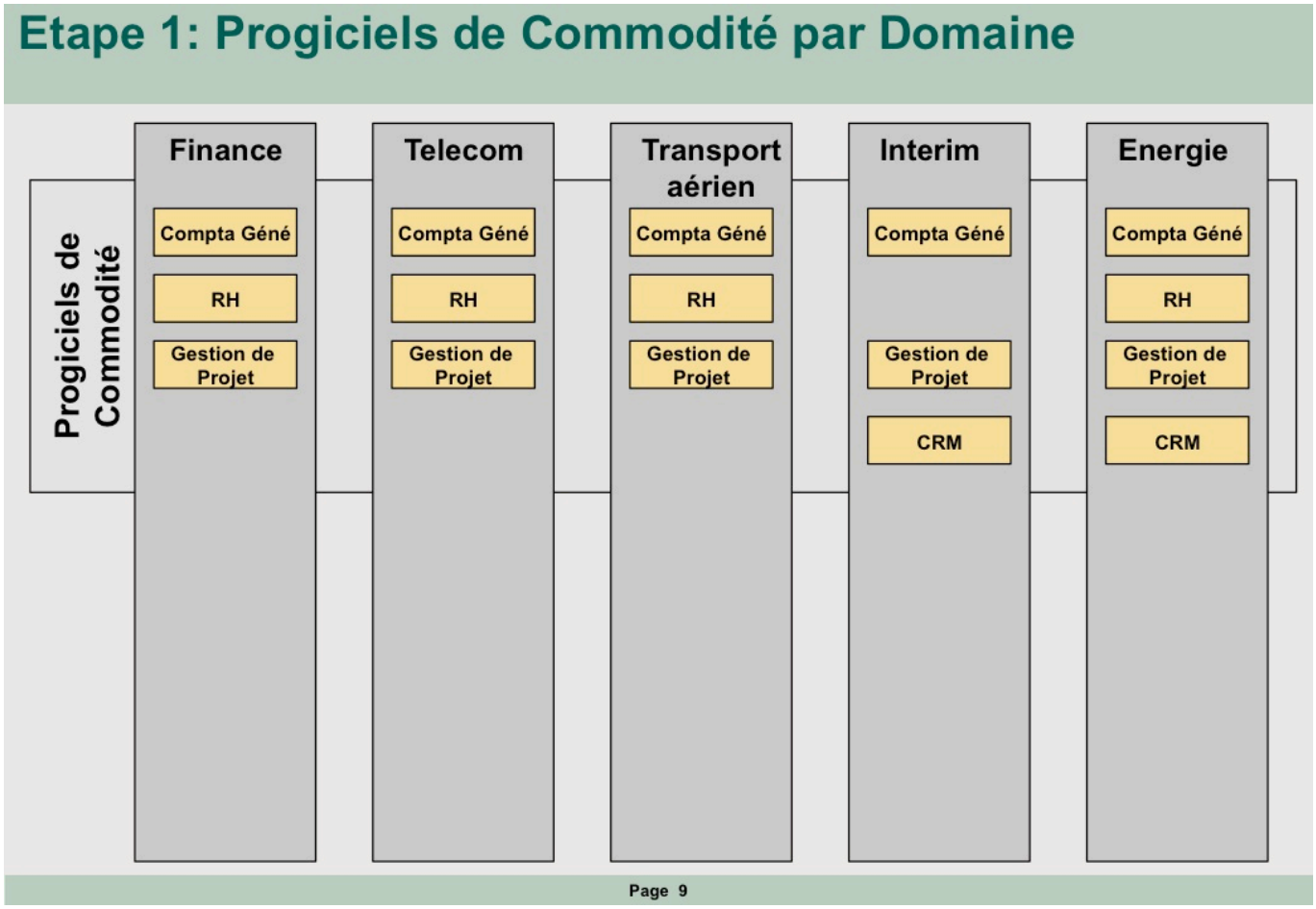
Exemples:

- Axa-France : passer de nombreuses Solutions (Solution de souscription, Solution CRM, Solution d'éditique, Solution de contrôle médical…) à une seule pour les Collectives
- BNP-Assurance : passer de quatorze Solutions de Prévoyance à une seule Solution
- Michelin : Pour chacun des six principaux domaines de l'Entreprise, Michelin a développé une seule Solution, appelée Master Application, ce qui a réduit fortement l'hétérogénéité du parc applicatif.
- Total a procédé à l'harmonisation des Processus RH au niveau groupe au travers d'un outillage unique dont le cœur est la solution SAP HCM

3.6 Après les Progiciels de Commodité, les Entreprises recherchent des Progiciels Verticaux

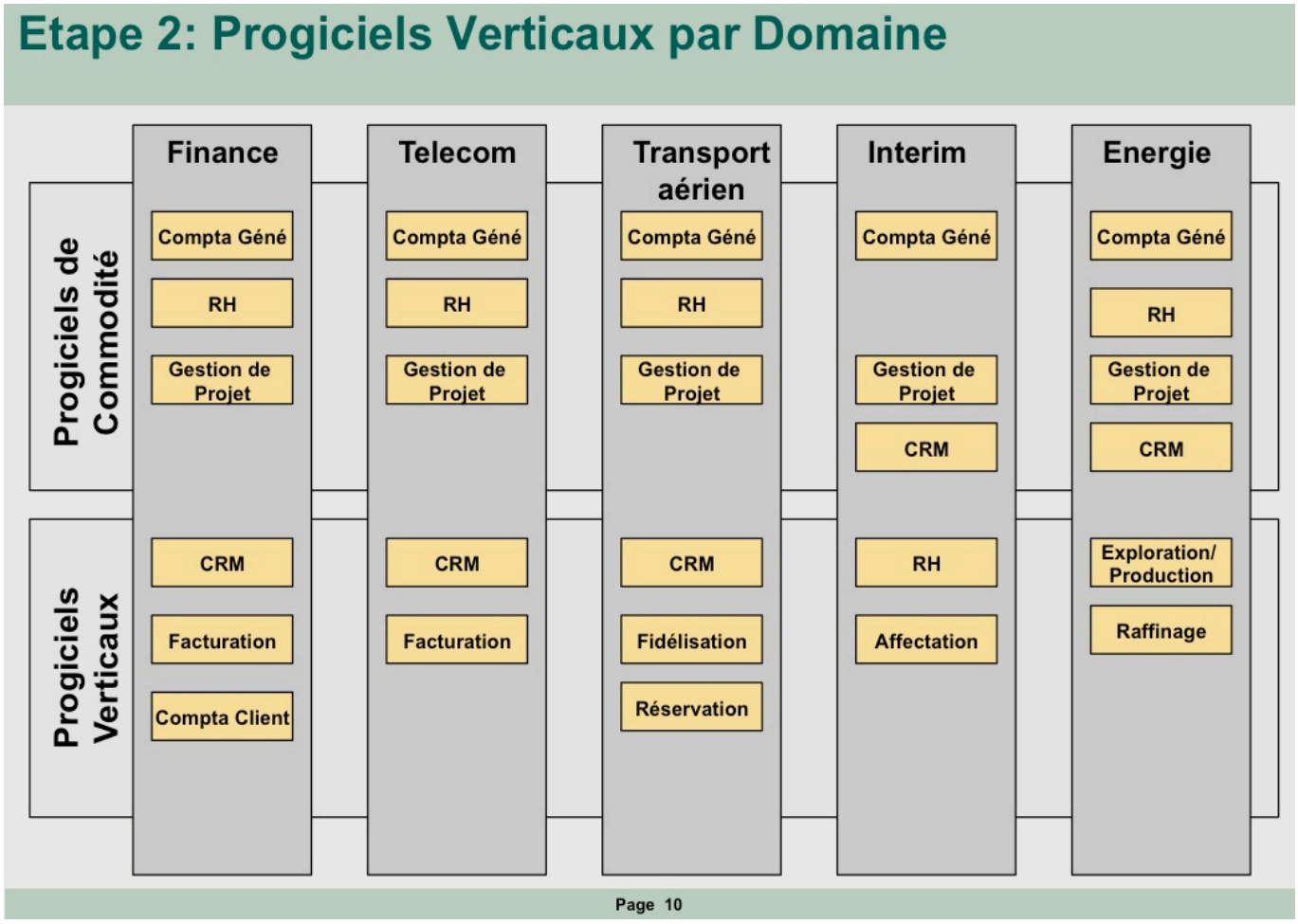
Revenons sur la progressivité de la diffusion des Progiciels au sein des Entreprises.

3.6.1 Etape 1 : Progiciel de Commodité par Domaine



Les Entreprises ont commencé par installer des Progiciels pour des Solutions de Commodité : un Progiciel pour la paye, un Progiciel pour la comptabilité, un Progiciel pour les achats, un Progiciel pour la messagerie…
Ils sont caractérisés par le fait que les fonctions sont **similaires** chez les différents clients. Il est plus économique pour tous de bénéficier d'un seul logiciel. C'est sur ce domaine que s'est développée l'industrie des Progiciels. A titre d'exemple, SAP a 46.000 clients dans le monde.
Ils ont aussi acquis des Progiciels-Composants techniques qui les aidaient à fabriquer leurs propres Solutions spécifiques.

3.6.2 Etape 2 : Progiciel Vertical par Domaine



Mais la nécessité d'évolutions rapides et la croissance de la complexité des métiers exigent de plus en plus de travail de la part des équipes de développement internes qui ont du mal à faire face à la fois à l'évolution des Solutions existantes et à l'installation de nouvelles Solutions. Les Entreprises ont donc recherché des **Progiciels Verticaux** pour informatiser certains Domaines de leur métier**.**

Comme les Entreprises souhaitent se différencier sur le cœur de métier plutôt que sur la gestion de ressources, ces Progiciels Verticaux doivent avoir une capacité de **personnalisation** beaucoup plus évoluée que les Progiciels de Commodité. Les Progiciels Verticaux sont donc **plus complexes** que les Progiciels de Commodité. Les Entreprises demandent aux Editeurs de les aider à gérer le parallélisme entre les ajouts spécifiques à chaque Entreprise et les versions successives du Progiciel Vertical.

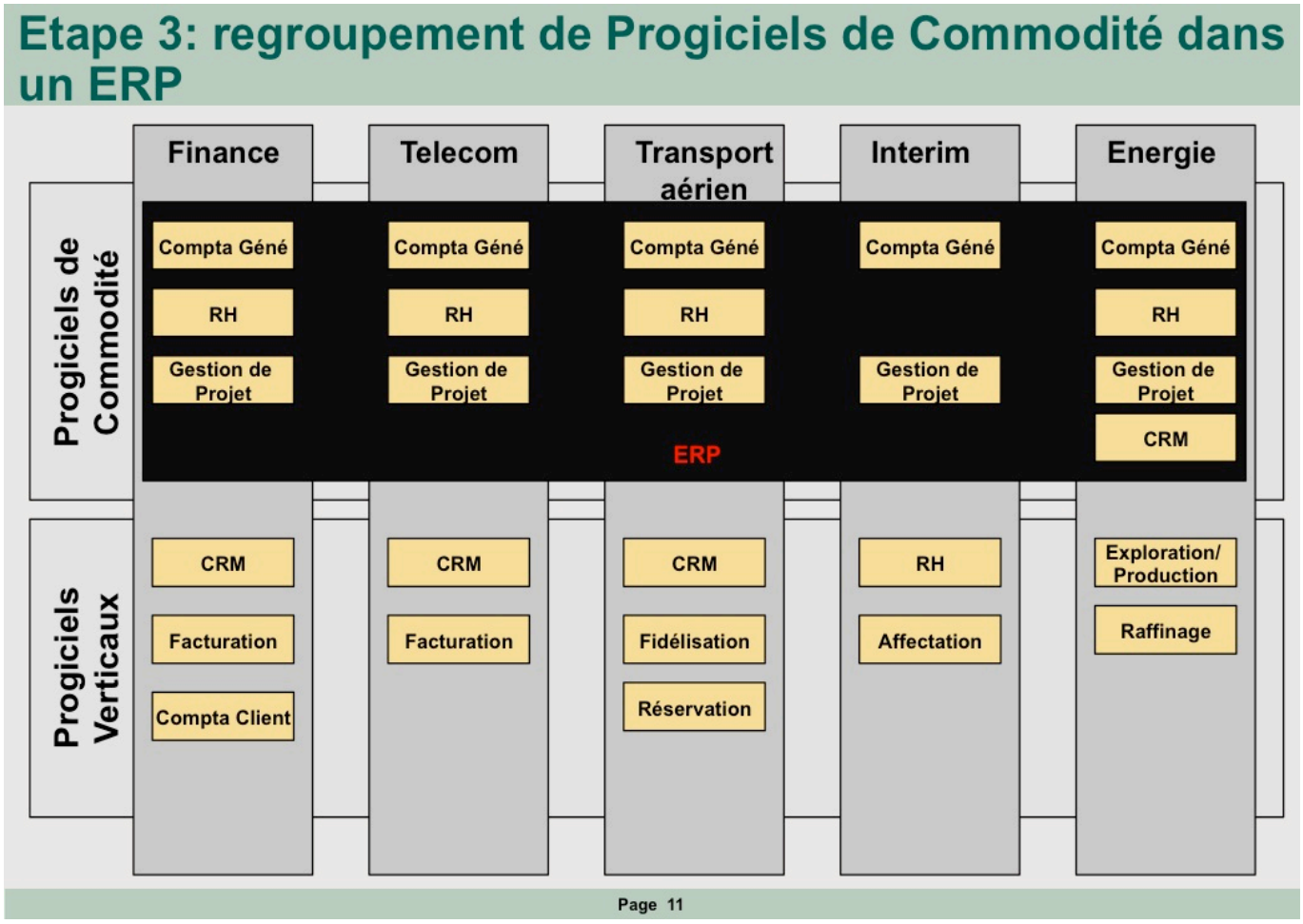
Remarque : le même Domaine fonctionnel peut être un Progiciel de Commodité pour certains et un Progiciel Vertical pour d'autres. Exemples :

- *un Progiciel de paye est un Progiciel Vertical pour l'intérim*
- *un Progiciel de facturation est un Progiciel Vertical pour les télécoms*
- *un Progiciel achat est un Progiciel Vertical pour la grande distribution*

Point de vue du CEISAR

Attention à ne pas choisir un Progiciel de Commodité pour un Domaine Vertical. Le CRM en est un bon exemple : certaines Entreprises ont choisi un CRM de Commodité mais ont dû le modifier profondément pour y introduire les fonctions Verticales nécessaires. Les coûts de personnalisation ont été prohibitifs, et l'Entreprise n'a plus bénéficié des nouvelles versions du Progiciel, les montées de version devenant trop difficiles.

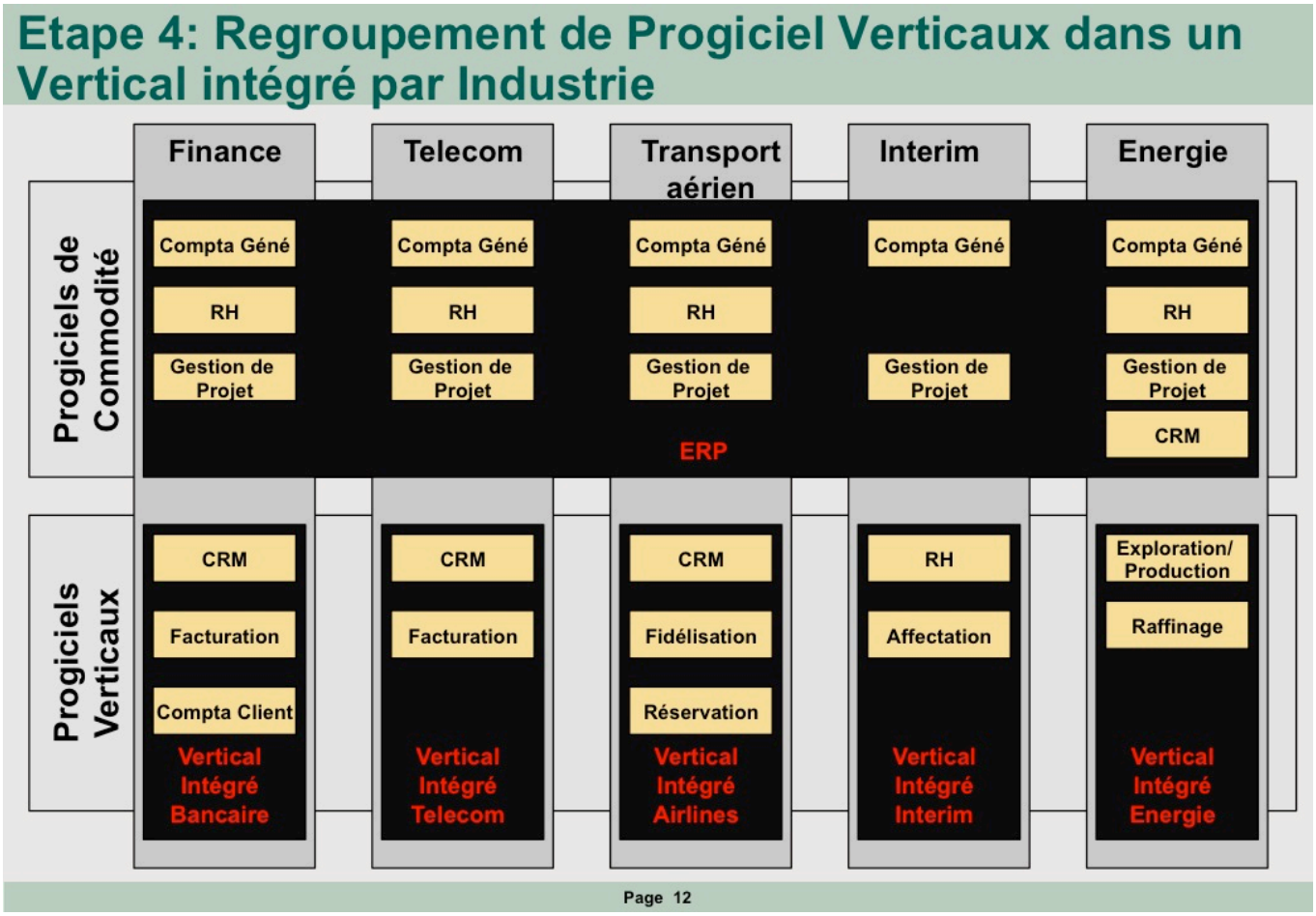
3.6.3 Etape 3 : ERP



Remplacer une Solution interne par un Progiciel est un allègement, mais le problème de l'interopérabilité subsiste. Il s'accentue si les Progiciels choisis proviennent d'Editeurs différents.
Devant la multiplication des interfaces entre Progiciels spécialisés par domaine fonctionnel, les Entreprises font aujourd'hui pression sur les Editeurs pour bénéficier de Progiciels plus intégrés qui combinent des domaines différents.
La première réponse a été fournie avec les ERP « *Enterprise Resource Planning* » qui intègrent différents Progiciels de Commodité.
Les Entreprises souhaitent souvent n'utiliser qu'un sous-ensemble de ces ERP et demandent donc à l'Editeur de fournir un ERP sous forme de modules, la contrainte étant que quelque soient les Modules choisis, l'intégration entre Modules est prérésolue par l'Editeur.

La place de l'ERP n'est pas la même dans le monde industriel ou dans le monde du service.
Dans le monde industriel, l'essentiel des Processus informatisés sont liés à la gestion de ressources ou au pilotage (voir ci-dessus), ce que traite un ERP. Donc le poids de l'ERP peut devenir prépondérant dans le système d'information d'une Entreprise industrielle.
Par contre, dans le monde du service, l'essentiel des Processus informatisés sont des Processus Primaires différents pour chaque Secteur économique, ce que ne sait pas gérer un ERP.
On en arrive donc à la nécessité de Vertical Intégré.

3.6.4 Etape 4 : Vertical Intégré



L'étape ultime est l'intégration des Progiciels Verticaux. L'affaire est plus difficile pour les Editeurs puisque la couverture fonctionnelle de ces Verticaux est plus complexe et que l'on s'adresse à une cible de clientèle restreinte au secteur concerné : la justification économique de l'investissement est plus difficile pour l'Editeur. Mais la pression des Entreprises doit faire émerger progressivement une offre mature de « Verticaux Intégrés ».
Comme l'Entreprise souhaite bien sûr conserver certains Progiciels de Commodité par domaine fonctionnel, tels que Progiciel de Paye ou Progiciel de comptabilité générale, ou Progiciel d'Editique, elle demande à l'Editeur la cerise sur le gâteau : le Progiciel Intégré Vertical doit être livré avec toutes les Interfaces qui lui permettent de coexister avec l'ERP choisi.
Compte tenu du périmètre fonctionnel couvert par un Progiciel Vertical Intégré, la majorité des Entreprises l'installera progressivement. Il faudra donc faire coexister Solutions anciennes et Vertical Intégré, ce qui suppose que le Progiciel Intégré soit **livré avec des Services métier** qui facilitent cette intégration.

3.6.5 Quelle Frontière entre les Progiciels de Commodité et les Progiciels Verticaux ?

Suivant les secteurs d'activité, les Fonctions de Commodité et les Fonctions Verticales ne sont pas les mêmes. Les Entreprises marient des Progiciels de Commodité avec des Progiciels Verticaux.

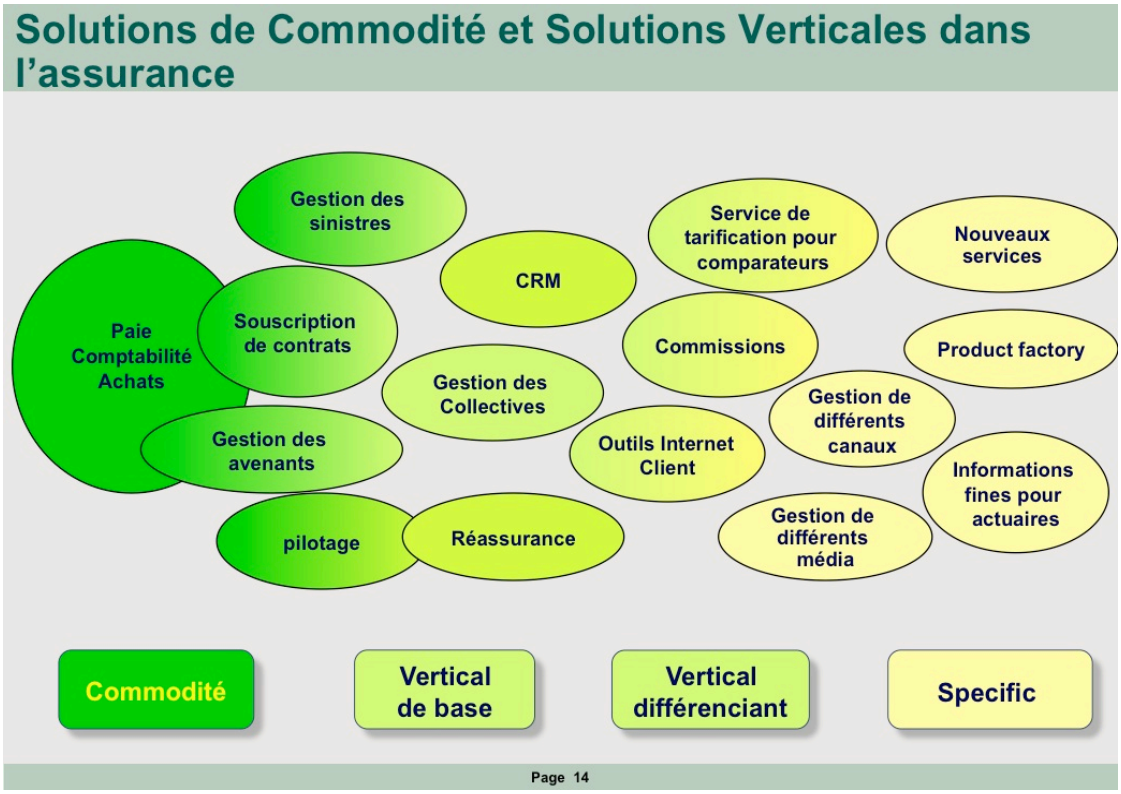
Par ailleurs, chaque fois qu'un Progiciel Vertical réussit à modéliser de nouvelles fonctions du métier, il banalise la fonction puisqu'elle est offerte à tous. Le domaine sur lequel l'Entreprise peut obtenir un avantage concurrentiel se déplace progressivement. L'Entreprise doit en permanence s'interroger sur les Fonctions d'avantage concurrentiel à développer et l'adéquation de ses Solutions à cette typologie (ai-je les bonnes Solutions de Commodité ? Ai-je les bonnes Solutions d'avantage concurrentiel ?).

Le schéma suivant provient d'Air France.
Il classifie les Solutions principales :

1. celles de gauche sont des Solutions de Commodité qui n'offrent pas d'avantage concurrentiel déterminant : les Progiciels du marché font parfaitement l'affaire
2. celles du milieu sont des Solutions bâties sur des Progiciels Verticaux personnalisables
3. celles de droite sont des Solutions spécifiques qui apportent un avantage concurrentiel à la compagnie

L'ensemble se déplace progressivement vers la gauche au fur et à mesure que de nouvelles fonctionnalités de plus en plus sophistiquées sont informatisées (arrivées par la droite) et que les Progiciels Verticaux s'enrichissent (déplacement de la droite vers le centre).
Pour exemple, l'enregistrement classique des passagers était un facteur concurrentiel. Il se banalise. C'est maintenant par des Solutions comme les « Ground and in-flight e-services » et le « Revenue Management » que les compagnies aériennes se différencient.

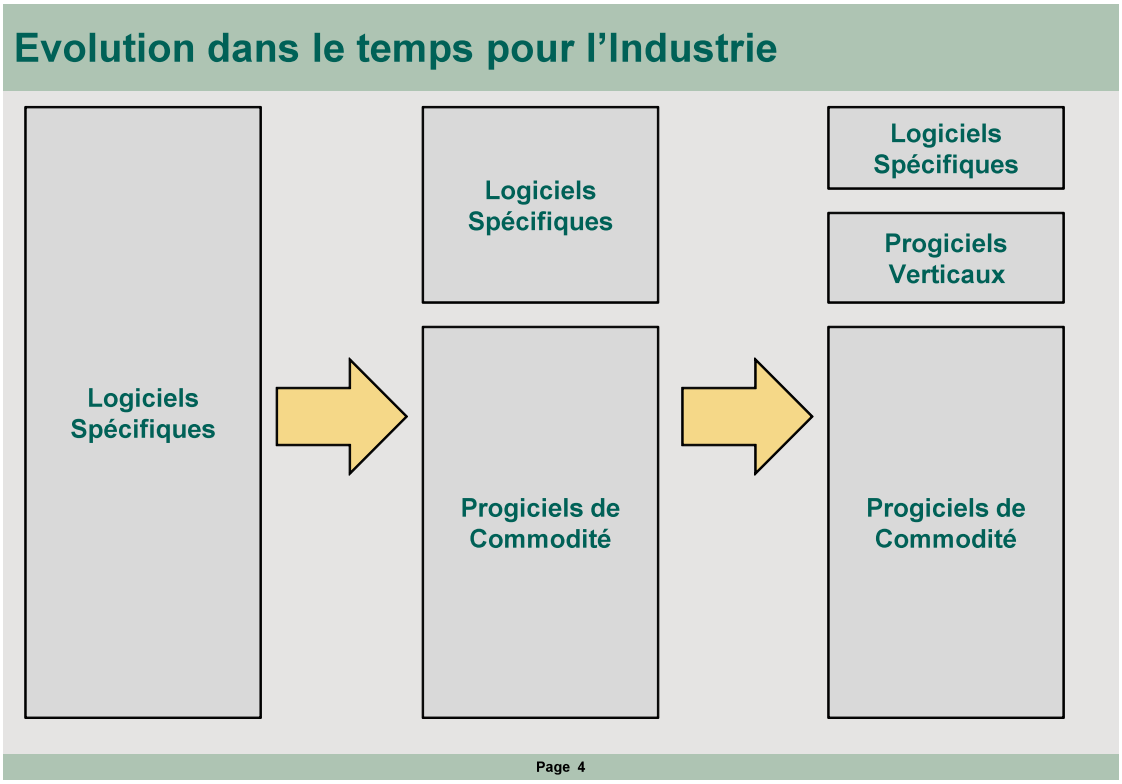
Dans le Secteur de l'assurance, on pourrait imaginer le schéma suivant : les nouveaux domaines sont le « **Product Factory** » pour gagner en time to market, l'offre de **nouveaux services** en prolongement de l'offre des services classiques de l'assurance, la gestion de **nouveaux médias** tels que les tablettes ou les iPhones, l'analyse **d'informations techniques** beaucoup plus fines pour mieux comprendre les différents risques et adapter l'offre.



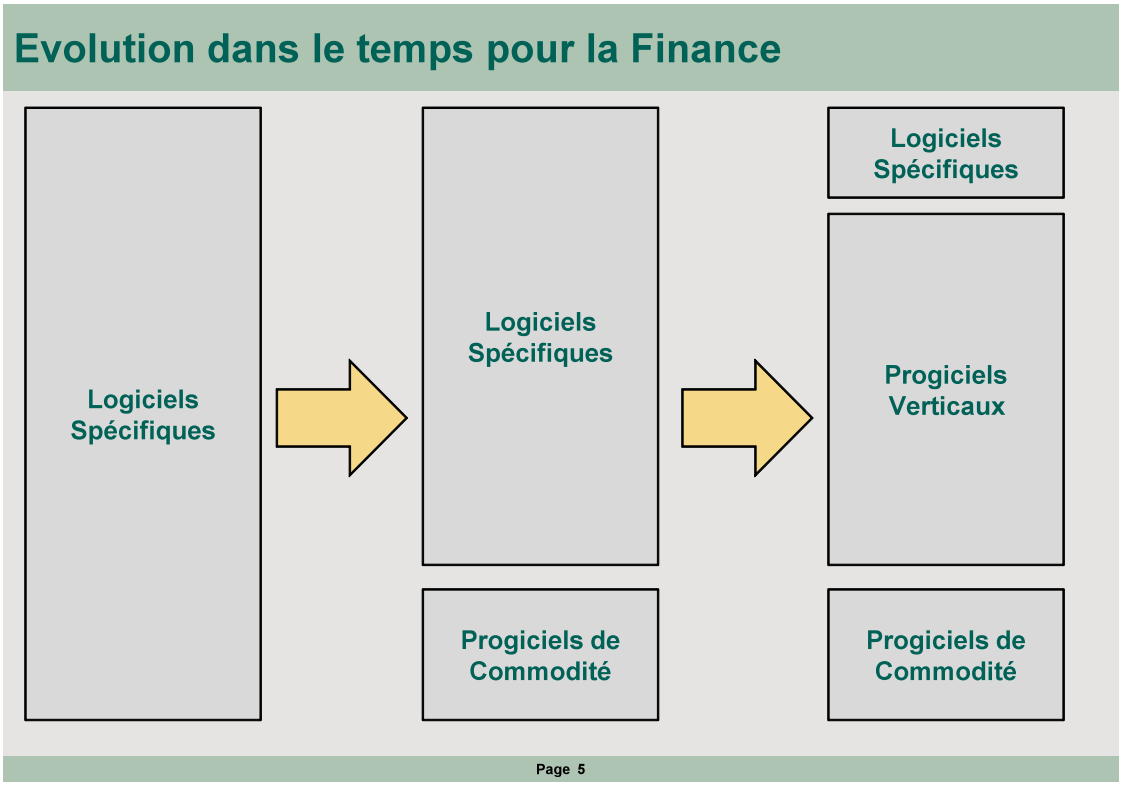
3.6.6 Les Progiciels de Commodité sont majoritaires dans l'industrie, les Progiciels Verticaux sont majoritaires dans la finance

Chaque Entreprise a commencé à développer ses propres Solutions.

Dans l'Industrie, les Progiciels de Commodité ont remplacé la majorité des logiciels spécifiques. Les Progiciels Verticaux ne représentent qu'une faible part du patrimoine applicatif.



Dans la Finance, les Progiciels de Commodité ont aussi été utilisés, mais les logiciels spécifiques qui sont majoritaires n'ont pas encore été remplacés par des progiciels Verticaux : c'est un des grands marchés de demain.



Point de vue du CEISAR

Cette consolidation du marché des Progiciels est classique de tous les secteurs d'activité arrivant à maturité. La domination du marché des Progiciels de Commodité par deux ou trois grands éditeurs est à la fois un risque et une chance. C'est un risque car le choix et donc la force de négociation des clients se réduit. Mais c'est également une chance car les Progiciels de Commodité se banalisent et arrivent à maturité permettant de couvrir correctement et avec un minimum de risques les Fonctions principales de Commodité. L'enjeu pour les Entreprises est alors de savoir limiter leurs investissements sur ces Progiciels de Commodité (pourquoi refaire une nième fois votre comptabilité ?) et de diriger un maximum de leurs investissements sur des Fonctions plus différenciantes, ce qui impliquent de les identifier clairement …
En revanche, le marché des Progiciels Verticaux n'est pas encore arrivé à maturité. Comme pour les Progiciels de Commodité, on peut anticiper le développement de Solutions au périmètre plus large et intégrées : les Progiciels Verticaux Intégrés.

3.7 Personnaliser par configuration et non par développement spécifique

La personnalisation du Progiciel par **développement spécifique** complémentaire est coûteuse : coûts de développement, d'intégration, tests de non régression, montée de version plus difficile…
Le problème est encore plus crucial dans les Progiciels Verticaux. Chaque montée de version est extrêmement difficile pour faire suivre les développements spécifiques.

Les Entreprises recherchent donc des Progiciels qui leur permettent d'évoluer rapidement par **configuration** : c'est-à-dire par paramétrage, moteur de règles, moteur de workflow, données dynamiques… La personnalisation a lieu dans un **cadre prédéfini** où l'exigence de tests est plus faible. Ils permettent aussi de mettre la personnalisation directement **entre les mains du métier**. Les montées de version sont pratiquement automatiques puisque la configuration se traduit uniquement par des données nouvelles. On y gagne donc en agilité, en fiabilité et en autonomie.

Le niveau d'exigence est alors beaucoup plus fort sur le Progiciel : l'Editeur doit avoir **modélisé le métier en profondeur** pour arriver à offrir des outils de configuration puissants et stables qui s'appuient sur cette modélisation métier.

Point de vue du CEISAR
Plus le Progiciel permet des configurations riches, plus il permet à l'Entreprise de coller à son besoin sans avoir à réaliser des développements complémentaires. Cela est essentiel pour garantir des montées de version successives moins coûteuses et sans risque majeur. Ce n'est possible que si l'Editeur a su finement Modéliser le métier.

3.8 Gagner du temps avec des Progiciels préremplis

Les Entreprises souhaitent que l'installation du nouveau Progiciel soit la plus rapide possible.
Un facteur d'accélération est dans la **préconfiguration**.
Comme les possibilités de configuration croissent par rapport aux développements sur-mesure, ce sont souvent les tâches de configuration qui sont sur le chemin critique.
Les Entreprises préfèreraient obtenir des Progiciels préconfigurés pour qu'ils ne travaillent que par différence.
A titre d'exemple : la traduction à la langue locale, les codes postaux et ville, les règles fiscales, les produits les plus courants du marché, les schémas comptables, les standards professionnels, les Processus les plus courants…
Certains éditeurs le proposent comme des options du Progiciel.

3.9 Les Entreprises cherchent des Solutions rapides et scalables

Pour des domaines nouveaux à informatiser rapidement, ou des besoins très ponctuels les Entreprises souhaitent disposer de Progiciels installables très rapidement et permettant une montée en charge progressive, quitte à jeter la Solution pilote au bout d'un temps limité.
Ils s'interrogent sur la capacité du « Cloud » à résoudre ce problème.
Ils vont même plus loin, et devant la rapidité d'utilisation du Cloud, ils s'interrogent sur la possibilité pour cette nouvelle offre de prendre aussi en compte d'autres besoins plus classiques, et d'être ainsi déchargés de l'exploitation informatique.

3.10 Les Entreprises veulent développer des synergies à l'international

Tous les sponsors du CEISAR (Axa, BNP, Michelin, Total…) recherchent des Solutions internationales et non plus des Solutions par pays
Les grands groupes internationaux cherchent à rationaliser leurs Solutions entre pays, pour réduire leurs coûts, pour proposer des Produits internationaux, pour échanger plus aisément les bons **Produits** et les bonnes Pratiques entre pays et rendre au même **client** des services dans tous les pays, pour offrir des plateformes de BPO communes à plusieurs pays.
Le Progiciel a l'avantage de la neutralité, ce qui peut faciliter l'acceptation par les différents pays. Un Progiciel commun aide aussi l'Entreprise à s'installer rapidement dans de nouveaux pays.

Le Progiciel doit alors permettre une personnalisation par pays : non seulement langue, devise, mais aussi

- externalisation des aspects fiscaux et réglementaires qui doivent être traités par configuration et non par développement spécifique complémentaire
- isolation des interfaces avec les systèmes d'échanges interprofessionnels quand ils sont propres à chaque pays

Deux stratégies peuvent être définies :

- Convergence vers une cible « best of breed »
- Convergence vers une cible Intégrée

3.10.1 Convergence vers une cible « best of breed »

Le Groupe définit une bonne Solution pour chaque besoin.
Chaque Solution est construite sur sa propre Fondation.
Chaque fois qu'une filiale du Groupe a besoin de changer une Solution, elle doit choisir la Solution préconisée par le Groupe. Une Compagnie ne peut ignorer l'investissement fait par le Groupe. Le Groupe n'impose pas de rythme de migration aux filiales : chacune doit définir son planning en fonction de ses besoins d'amélioration.
Cette approche a pour avantage **d'harmoniser** les Solutions entre filiales : il sera plus facile par la suite d'échanger des Produits ou des Processus, ou de créer des back-offices communs.
Les filiales n'ont pas besoin d'investir dans l'étude de la cible : on centralise la fonction achat et on réalise des économies d'échelle auprès des Editeurs.
Si le Groupe a les moyens de **préintégrer** les différentes Solutions entre elles à l'aide d'une approche SOA, les filiales n'auront pas besoin de refaire chacune ce travail d'intégration ; elles auront ainsi davantage confiance dans la capacité de ces différentes Solutions à s'exécuter ensemble.
Ainsi, dans le cadre de sa démarche d'Enterprise Architecture, Michelin a développé depuis 2003 un modèle global de Processus Métier dans les six principaux domaines d'activités de l'Entreprise (Recherche & Développement, Marketing & Ventes, Supply Chain & Logistique, Fabrication, Finance, Administration). Pour chaque domaine, ce modèle a ensuite permis de développer une Solution standard (à base de Progiciel et/ou de développements spécifiques) appelée « Master Application » et maintenue par un « Master Application Solution Center », véritable éditeur interne. Chaque Master Application est déployée par continent au rythme des besoins des filiales.

Par contre, cette stratégie **ne simplifie pas considérablement le Modèle de chaque Filiale** qui continue à être constitué de Solutions disparates : elle n'apporte ni l'agilité, ni la souplesse d'organisation au niveau de chacune des filiales.

3.10.2 Convergence vers une cible Intégrée

L'**agilité maximale** est obtenue lorsque les fonctionnalités sont bâties sur la **même Fondation**, au sein de la même Solution : même outils de développement, équipes proches, réutilisation de composants de construction (voir le livre blanc du CEISAR sur l'agilité : on peut atteindre des taux de réutilisation de 80%).

La **souplesse d'organisation** est atteinte lorsque l'utilisation des différentes fonctionnalités est banalisée : même station de travail, même mode d'usage, même ergonomie, même navigation, même présentation, même identification/authentification, même gestion de Processus. On peut alors aisément et progressivement permettre aux utilisateurs d'élargir leur domaine d'action ce qui permet de réorganiser plus facilement l'Entreprise.

C'est un cas encore peu fréquent mais qui devrait se développer dans les prochaines années au fur et à mesure de l'émergence de **Progiciels Verticaux Intégrés**. En effet l'Entreprise n'a généralement pas le temps ni la compétence pour construire et faire mûrir une Fondation : elle recherchera une Fondation disponible sur le marché qui peut être un des Modules offerts par les éditeurs de Progiciels (voir un exemple sur www.ceisar.org et télécharger « quel modèle d'Entreprise pour l'assurance de demain », §5.3)
Dans ce cas, les avantages précédents sont conservés : harmonisation à terme entre les Solutions des différents filiales, centralisation des achats.
Mais en outre chaque filiale va progressivement converger vers une cible qui diminue ses **coûts**, accroît son **agilité** et sa **souplesse d'organisation**. C'est la stratégie la plus efficace pour un Groupe.

Pour réussir ce type d'approche long terme, il faut :

- que le Groupe fasse un **bon choix** en terme de Modèle cible : ce Modèle doit être plus simple pour chaque filiale et il doit respecter les spécificités locales; un choix de Vertical Intégré est nécessaire
- que le Groupe **préconfigure** le Vertical Intégré pour que chaque filiale n'ait pas à le faire
- que le Groupe **intègre** ce Vertical aux Solutions de Commodité en place
- que le Groupe dispose d'une équipe centrale pour **supporter** les filiales dans l'adaptation du Modèle commun ;
- que chaque filiale **joue le jeu** du Groupe ;
- que la **gouvernance** qui accompagne cette politique ne fasse pas porter le poids des investissements aux premières filiales qui vont essuyer les plâtres d'un Modèle nouveau ; cela suppose que le Groupe ait dégagé des moyens pour se doter d'un Modèle cible cohérent et pour le supporter auprès de ses filiales.

En résumé, il faut que le Groupe se comporte vis-à-vis de ses filiales **comme un relais de l'éditeur de logiciel**: une équipe de R/D centralisée, une équipe de support, un Processus de définition des versions futures, une école de formation, une documentation propre…

3.11 Les Entreprises veulent mieux comprendre l'Architecture du Progiciel

Ces dernières années la tendance a été la suivante :

- La majorité des choix de Progiciels sont prioritairement conduits sur la base des exigences fonctionnelles.
- Une majorité d'Entreprises a progressivement réduit les compétences informatiques internes.
- Les développements spécifiques sont généralement exécutés par des sociétés extérieures, qu'elles soient locales ou non.
- L'intégration est confiée soit à l'éditeur soit à une SSII, soit à une association des deux parties.

Le raisonnement courant peut se résumer à

- *« lorsque j'achète une voiture je prête attention aux fonctions offertes, au prix et à la notoriété du fournisseur, je n'ai pas besoin d'ouvrir le capot ; pourquoi ferais-je différemment pour un Progiciel »*
- *« mon métier c'est l'automobile, pas l'informatique : donc j'externalise l'informatique pour me concentrer sur mon cœur de métier » (on pourrait appliquer le même raisonnement à la finance, aux ressources humaines ou au marketing, ce qui conduirait à externaliser ces fonctions)*

Mais les Entreprises se trouvent confrontées à de grandes difficultés :

- les difficultés **d'intégration** des Progiciels
- les difficultés de **personnalisation** des Progiciels
- les difficultés **d'évolution** des Progiciels

qui se traduisent par les échecs de projet, des accroissements de coûts et de délais parfois considérables.

Les Entreprises réalisent qu'elles doivent **comprendre** l'Architecture interne du Progiciel pour mieux en mesurer leurs capacités d'intégration, de personnalisation et d'évolution.

Pour reprendre l'analogie avec l'automobile,

- *une automobile est autonome : elle n'a pas besoin de s'intégrer à d'autres véhicules*
- *les options sont choisies avant livraison : on ne change pas sa couleur ultérieurement*
- *pour bénéficier des nouvelles fonctions des nouveaux modèles, il faut remplacer le véhicule*

Pour résumer, l'automobile choisie fonctionne isolément et n'évolue pas. Un Progiciel a un cycle de vie beaucoup plus complexe.

Les Entreprises expriment une exigence croissante de transparence sur l'Architecture du Progiciel, ce qui suppose qu'elles se dotent d'architectes capables de comprendre : on devrait assister au retour d'excellentes compétences informatiques dans les Entreprises.
Prendre en compte l'Architecture dans le choix d'un Progiciel a des impacts sur la gouvernance qui sont abordés plus loin.

Point de vue du CEISAR

Le manque d'intérêt de nombreuses Entreprises pour l'Architecture au motif que ce n'est pas leur cœur de métier a des conséquences : elles réduisent leur rôle à l'expression de besoin, à la recette et au déploiement et ne sont plus capables de maîtriser l'évolution de leurs Solutions :

- comment évaluer la capacité d'un Progiciel à évoluer, à être personnalisé, à s'interfacer si on n'en comprend pas l'Architecture ?
- comment comprendre les efforts (charge et délais) à fournir pour des évolutions du Progiciel si on ne sait pas comment est structurée la Solution ?

Pour éviter d'être pieds et poings liés à leurs fournisseurs, les Entreprises devraient ne pas se concentrer seulement sur les fonctionnalités offertes, mais aussi comprendre **comment le Progiciel est architecturé** : un besoin de transparence et un retour des informaticiens dans les Entreprises sont nécessaires.

3.12 Les Entreprises recherchent une Fondation pour les Solutions spécifiques complémentaires aux Progiciels

Les Entreprises qui souhaitent continuer à développer des Solutions spécifiques pour des besoins Verticaux sont à la recherche d'une Fondation qui leur facilite la tâche.
Cette Fondation peut être développée en interne par l'Entreprise. Elle peut aussi être acquise à l'extérieur si une offre existe, ce qui permettrait aux Entreprises de se consacrer au développement de Solutions spécifiques plutôt qu'à la construction et à l'entretien d'une Fondation.

Point de vue du CEISAR

Pour donner un ordre de grandeur, une Fondation peut fournir 80% d'une Solution spécifique sous forme de composants réutilisables. L'investissement pour développer une Fondation est d'au moins trois ans avec une équipe qui a déjà de l'expérience dans ce domaine. Les Entreprises préféreraient donc utiliser une Fondation déjà disponible auprès d'un Editeur, ce qui suppose qu'elles aient les moyens de la jauger et qu'elles aient des garanties de pérennité. On peut acquérir un **Progiciel** auprès d'un Editeur, on peut aussi acquérir la **Fondation** qui a permis à l'Editeur de construire le Progiciel : les Solutions spécifiques construites par l'Entreprise à l'aide cette Fondation s'intégreront aisément au Progiciel.

4 Evolution de l'Offre des Editeurs

Les Editeurs sont conscients des besoins des Entreprises : ils font évoluer leur offre pour aller dans le sens des attentes du marché.
Mêmes ceux qui ont une place prépondérante, ont décidé de changer profondément leur offre pour suivre ces tendances.
Cette évolution de l'offre influence aussi certaines Entreprises qui n'ont pas toujours conscience de ces tendances, mais qui bénéficient des innovations des Editeurs.
L'évolution la plus importante est la couverture fonctionnelle croissante des Progiciels. Les Entreprises comprennent progressivement qu'un Modèle d'Entreprise constitué d'une multitude de Solutions différentes choisies indépendamment constitue un ensemble complexe qui est un frein à l'évolution.
Cette prise de conscience a des conséquences profondes sur l'industrie des Progiciels : elle pousse les Editeurs à offrir les Progiciels Intégrés que demandent les Entreprises.

RÉSUMÉ

- L'offre de fonctionnalités progicialisées est en *croissance* : tout particulièrement les Verticaux.
- Les Progiciels sont de plus en plus **ouverts** et intéropérables
- L'offre de Progiciels-**Composants** émerge au sein ou à coté de l'Offre de Progiciels-Solutions
- Pour réduire la complexité des SI de leurs clients, les Editeurs offrent des Solutions à périmètre fonctionnel **plus large**.
- Le marché des Progiciels de Commodité s'est **intégré**: SAP et Oracle possèdent les trois quarts du marché d l'ERP.
- Les capacités de personnalisation par **configuration** montent en puissance, ce qui est indispensable pour les Progiciels Verticaux.
- Les Editeurs proposent des Progiciels **préconfigurés** pour accélérer leur mise en œuvre.
- Le **Cloud** Computing transforme l'offre des fournisseurs mais concerne aujourd'hui une part réduite du marché. Il est principalement adapté aux Solutions avec une faible adhérence avec le reste du SI ou pour gérer intégralement le SI d'une petite ou moyenne Entreprise.
- Les Progiciels **internationaux** se développent.
- La **transparence** progresse.
- Certains Editeurs proposent leur **Fondation** aux Entreprises qui souhaitent développer des Solutions spécifiques.
- La maîtrise d'un Progiciel **Vertical Intégré** va devenir un atout majeur pour tous les métiers qui aident les Entreprises à optimiser leurs Processus (cabinets de conseil, SSII, assistants à maîtrise d'ouvrage) et à les exécuter (BPO).

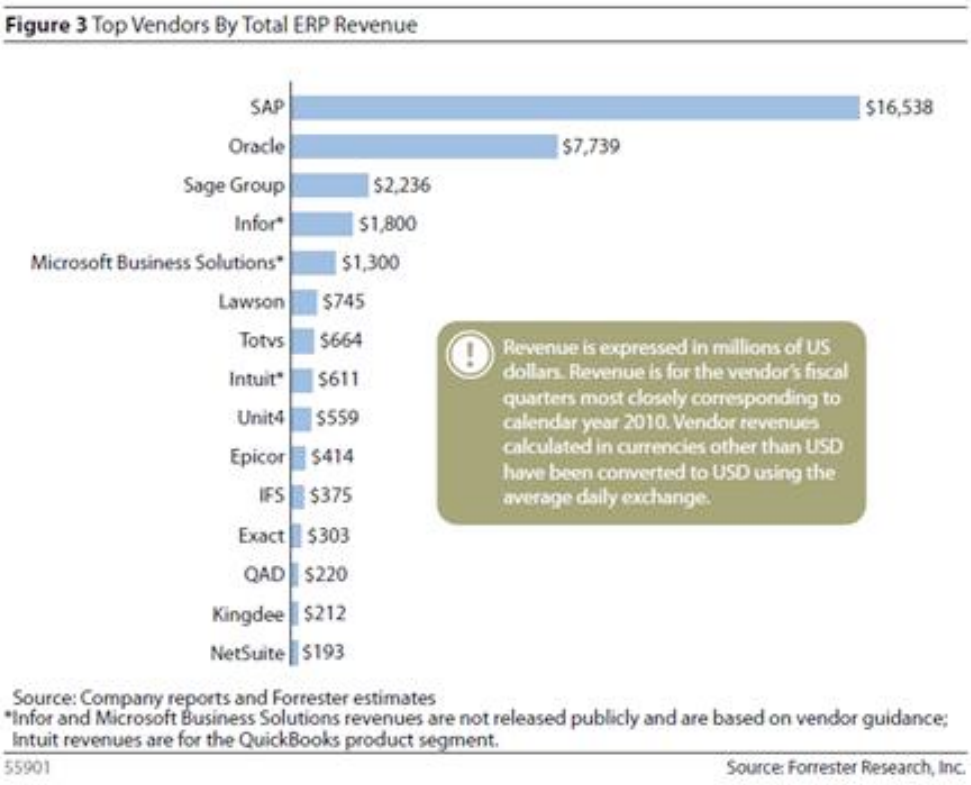
4.1 Les stratégies d'Oracle et SAP

Nous avons interrogé les deux plus grands Editeurs de Progiciels-Solution que sont Oracle et SAP pour mieux comprendre leur stratégie.

Oracle et SAP ont parfaitement compris les attentes du marché :

- attentes de Progiciels **Intégrés** et non de Progiciels indépendants
- attente de Progiciel **Verticaux**
- **modularité** de l'offre pour pouvoir installer progressivement le Progiciel
- attente de **Services-Logiciels** réutilisables pour mieux intégrer les Solutions existantes sur des plateformes d'échange
- personnalisation du Progiciel par **Configuration** plutôt que par Développements spécifiques
- pré-configuration de données
- recherche de Solutions **internationales**
- offre **Cloud** pour répondre au besoin de rapidité et de scalabilité
- besoin de **transparence** et utilisation par leurs clients des outils de développement des Editeurs pour des développements complémentaires

Aujourd'hui, Oracle et SAP représentent 75% du marché des ERP, qui est arrivé à maturité. On peut citer l'étude de Forrester « The state of ERP in 2011 », mentionnée dans le Monde Informatique du 11 mai 2011 :



Mais les démarches d'Oracle et de SAP sont différentes : elles sont liées à l'histoire.

4.1.1 Oracle : du best of breed vers l'homogénéité

L'offre d'Oracle est le résultat de la fusion d'Editeurs acquis successivement comme JD Edwards, People Soft, Siebel, Admin Server … qui se rajoutent à l'offre Oracle Application déjà disponible. Pour homogénéiser son offre, Oracle cherche non seulement à favoriser l'interopérabilité de ses Produits mais aussi à homogénéiser son offre en construisant et maintenant un nombre croissant de Produits sur la même Fondation.

En outre Oracle a bien identifié que le marché de forte croissance était celui des **Progiciels Verticaux** destiné aux secteurs qui consomment beaucoup d'informatique : Banque, Assurance, Retail, Télécoms, Utilities, …
En même temps Oracle a bien compris que les clients cherchaient à remplacer les silos applicatifs par des **Progiciels à périmètre large**.
La stratégie de convergence n'est pas la même pour Progiciels de Commodité et Progiciels Verticaux.

Oracle a fait l'effort de faire converger toutes les bonnes fonctionnalités de Commodité des Progiciels existants vers un ERP « Fusion Application » annoncé au 1er trimestre 2011.

Il s'agissait de fusionner :

CRM	Customer relationship management (socle commun décliné par industrie)
ERP	Enterprise resource planning
HCM	Human capital management
EPM	Enterprise performance management
SCM	Supply chain management

L'effort a été considérable, et a pris plus de temps que prévu (voir ci-dessous). Les nouveaux clients sont orientés vers cette nouvelle offre cohérente. Les anciens clients peuvent conserver les anciens

produits qui continuent à évoluer. A eux de décider de migrer vers le nouvel ERP, s'ils le souhaitent un jour : mais quels que soient les outils de migration mis à disposition par Oracle, on peut s'attendre à ce que la migration soit un projet important puisque l'ERP est construit sur une nouvelle Fondation.

Citation du Monde Informatique du 20 septembre 2010 sur la conférence « OpenWorld » d'Oracle à San Francisco:

Fruit de cinq années de développement et d'intégration, les applications Fusion d'Oracle reprennent le meilleur des offres ERP et CRM de PeopleSoft, JD Edwards, Siebel et Oracle

La première version tant attendue de la prochaine génération des applications Fusion d'Oracle sera disponible au premier trimestre de 2011. C'est ce qu'a déclaré hier son PDG, Larry Ellison, lors d'une allocution faite à la conférence OpenWorld 2010 à San Francisco. "Nous savions que nous devions réaliser une nouvelle génération de produits qui réunisse les meilleures fonctionnalités de PeopleSoft, de la suite E-Business et de Siebel," a-t-il dit. "Il nous a fallu cinq années et un énorme travail pour y parvenir. Cela continue à représenter un immense challenge en matière d'ingénierie." Oracle a du en effet ré-implémenter toutes les fonctionnalités "au sommet d'une infrastructure middleware moderne, et précisément au sein de notre propre middleware Fusion," a déclaré le PDG d'Oracle. "La Suite contient 10 000 processus intégrés uniques et la version finale comprendra 100 produits différents, tous disponibles simultanément," a-t-il ajouté. "Nous n'avions jamais fait cela auparavant, et j'espère que nous n'aurons pas à le refaire."

Pour les Progiciels Verticaux, l'affaire est encore plus difficile : on ne peut réécrire tous les Progiciels Verticaux sous la nouvelle Fondation d'Oracle : l'investissement serait encore plus important que pour l'ERP.

En effet, pour chaque Vertical, l'offre d' Oracle peut être composée de Solutions provenant de sources différentes. Ainsi, pour les opérateurs de télécommunications, Oracle propose une suite "Rapid Offer Design and Order Delivery" qui comprend trois Solutions provenant d'acquisitions différentes:

- CRM Siebel
- Billing Portal Software (spécialisé pour les Telcos)
- Activation OSM (prend la commande et l'éclate pour activer les produits et services)

Oracle a donc décidé une approche progressive : chaque Vertical est décomposé en deux parties :

- Partie 1 : le Progiciel existant qui tire parti progressivement de la Fondation d'Oracle dans ses nouvelles versions.
  - Encapsulation : grâce à un middleware (« Fusion Middleware ») et à un découpage en services, on arrive à encapsuler des fonctionnalités à travers un catalogue de services réutilisables par chacun mais chaque service reste développé avec son outillage d'origine
  - Réécriture progressive avec les nouveaux outils de développement, avec SOA Suite, et application des normes récentes pour s'interfacer avec les portails. C'est un changement important pour les équipes de développement d'Oracle.
- Partie 2 : un produit Vertical complémentaire « AIA » (Oracle Application Integration Architecture) qui inclut le middleware pour interopérabilité, le modèle de données du Vertical et l'orchestration des Processus verticaux « PIP » (Pre Integrated Processes). Cette orchestration permet d'enchaîner des Fonctions provenant soit du Progiciel existant soit d'autres Solutions qui peuvent être des Solutions Oracle ou non.

L'ensemble de ce mouvement est complexe à gérer : 30.000 personnes y travaillent. Mais il est nécessaire pour faire converger progressivement l'offre vers des ensembles plus intégrés et plus ouverts.

4.1.2 SAP : du monolitique vers le modulaire

SAP continue de développer l'essentiel de son offre avec son outil ABAP, complété par l'utilisation de Java pour l'interface utilisateur.

Quelques Produits comme Business Object pour le pilotage, évoluent avec leur environnement de développement d'origine. Mais ces Produits sont situés à la périphérie de l'offre SAP et sont aisément interfaçables. SAP bénéficie donc d'une **Fondation homogène,** de par son histoire essentiellement basée sur la croissance interne.

C'est grâce à cette homogénéité que SAP a pu rapidement proposer un ERP et prendre une part majoritaire de ce marché : sur le marché de l'ERP, SAP possède une avance considérable avec un chiffre d'affaires en 2010 de plus de 16 milliards de dollars égal à celui cumulé de tous ses concurrents.

Par ailleurs, SAP a développé et dispose aujourd'hui de **26 verticalisations** par industrie

- Automotive
- Banque
- Utilities
- Services financiers
- Secteur public
- Oil & Gas
- Live Science (Pharmacie)
- Chimie
- High Tech
- Consumer products (Danone, Nestlé)
- Retail
- Healthcare
- Telcos,
- Services (ISS-International Service Solutions)
- Gestion hôtelière
- Etc.

Il y a aujourd'hui près de 3000 services d'Entreprises disponibles dans SAP qui sont publics et consultables. Ils sont présentés par SAP sous la forme suivante :
« Un service d'Entreprise est un service métier, une fonctionnalité, qui se distingue d'un service Web par le fait qu'elle a une granularité métier, qu'elle est intègre d'elle même, c'est-à-dire qu'elle ne nécessite pas de prérequis ni de postrequis, qu'elle est indépendante d'une fois sur l'autre (pas d'adhérence entre deux sollicitations), elle suit les spécifications propres au métier. SAP propose des solutions axées sur la granularité métier (et non pas des services techniques) et on trouve sur www.sdn.sap.com (SAP Developer Network) le référentiel des services métiers exposés, consommables par qui souhaite, testables en ligne ».

Pour faciliter l'utilisation de ces Services, SAP dispose d'une démarche *Netweaver,* démarche technologique qui a deux axes :

- Un axe technologique qui consiste à mettre en place un **middleware** d'Entreprise ouvert comprenant tous les standards (OASIS[1], W3C[2], standards de migration, standards de composition, etc.) pour motoriser les applications SAP sans pour autant avoir des ruptures technologiques avec les autres applications.
- Une adaptation des modules métiers, de manière à ce qu'ils soient consommables par d'autres applications, en exposant cette logique applicative avec la bonne granularité métier.

Cette démarche date de trois à quatre ans mais nécessite, pour pouvoir être mise en œuvre, d'être au niveau de la dernière génération d'applications, c'est-à-dire dans la version SAP ERP 6.0. Aujourd'hui avec l'usage de cette dernière génération, certains clients ont une vraie démarche globale : ils orchestrent, réutilisent, mettant en œuvre une démarche d'Architecture (Airbus, Publicis, AXA). Pour les aider, SAP propose les outils ARIS d'IDS Scheer qui permettent de modéliser les Processus et d'offrir une vision par Processus, par application, ou par infrastructure.

[1] ***OASIS* (*Organization for the Advancement of Structured Information Standards***

[2] W3C World Wide Web Consortium

Point de vue du CEISAR

Les deux Editeurs ont la même vision : ils ont compris que la multiplicité de Solutions générait de la complexité chez leurs clients. Ils sont confrontés à la même problématique interne : elle est même encore plus complexe compte-tenu du fait que chaque Progiciel doit adresser les besoins de multiples clients.
Comme SAP a construit son offre sur la même Fondation, son problème est d'éclater son offre autrefois monolithique pour la décomposer en Modules que l'on peut installer séparément et d'offrir des Services Logiciels pour que leur Progiciel intéropère avec les autres Solutions.

Oracle a fusionné les Progiciels de Commodité dans un ERP, mais doit réécrire progressivement toutes les fonctionnalités Verticales sur une Fondation unique : c'est un énorme investissement. En attendant que la nouvelle offre soit disponible, Oracle préconise le « best of breed » et facilite l'intégration de ses différents Progiciels en offrant des interfaces entre eux.

Nous estimons que la concentration du métier de l'Edition de Logiciels va conduire une majorité d'Editeurs à suivre cette voie : faire converger une offre disparate vers un ensemble de Modules construit sur la même Fondation. C'est la seule façon de répondre aux attentes fondamentales de leurs clients : simplifier l'Architecture d'Entreprise, offrir un usage standardisé, synchroniser les évolutions…
La clé devient alors la qualité de la Fondation : rapidité de construction, prise en compte des dernières technologies, capacité de personnalisation de la Solution par Configuration.

4.2 L'offre de fonctions « Progicialisées » croît

Pour répondre aux attentes du marché, les Progiciels **Verticaux** sont de plus en plus présents et traduisent une proportion croissante du savoir-faire **métier** en logiciels. Le volume de fonctions informatisées ne fait que croitre.
Cette tendance ne signifie pas que le nombre de Progiciels augmente, compte tenu du périmètre croissant de chaque Progiciel, ni que le nombre d'éditeurs croisse compte tenu de la concentration du secteur.

4.3 L'interopérabilité progresse

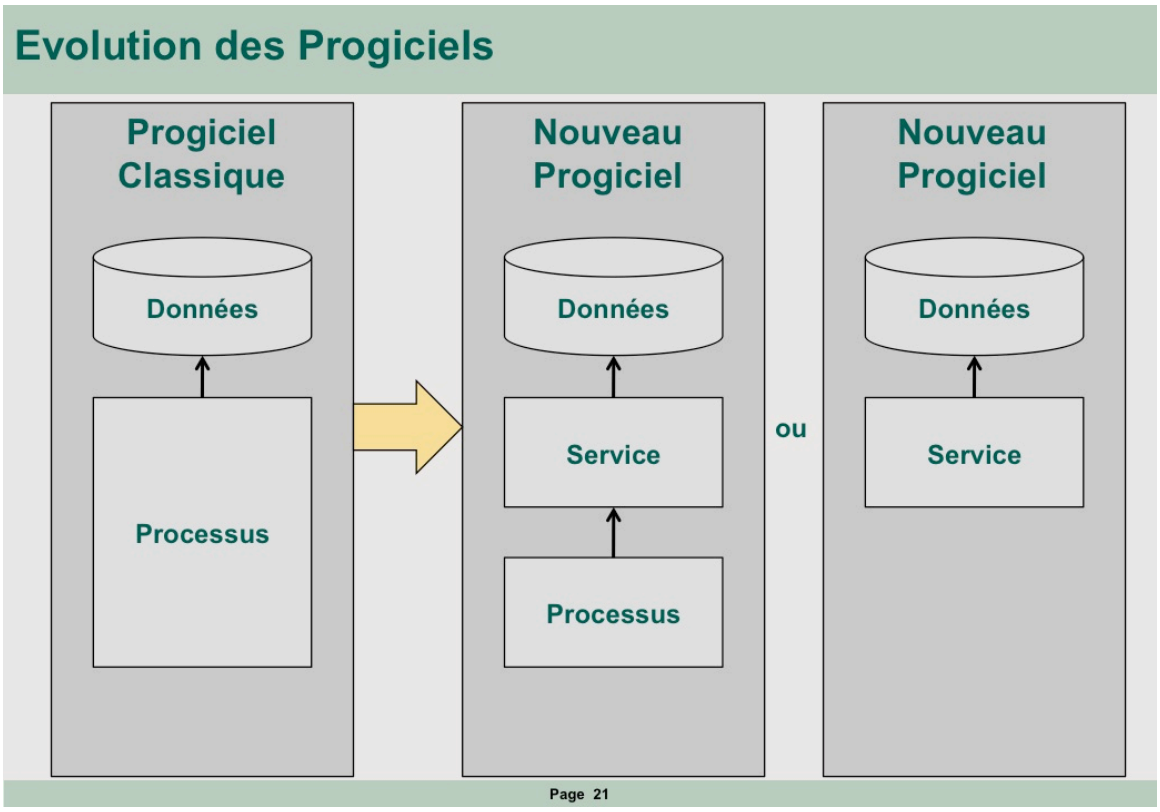
4.3.1 Les Editeurs offrent des outils et méthodes d'interconnexion

Compte tenu des efforts à fournir lors de l'installation d'un Progiciel pour le faire interopérer avec les autres Solutions, les Editeurs offrent des moyens pour les réduire :

- des **interfaces déjà disponibles** : il suffit alors de construire la demi-interface côté Solution externe pour s'interconnecter : on économise la charge de spécification de l'interface et la charge de développement de l'interface coté Progiciel
- des outils pour « **mapper** » les bases de données internes du Progiciel vers l'interface
- des outils pour **parser** des interfaces XML, ou CSV ou autre
- des outils pour **générer** des interfaces XML, ou CSV ou autre
- des outils pour **convertir** les types les plus fréquemment rencontrés tels que des dates, des montants, des devises, des types énumérés (comme « célibataire, marié, divorcé…), des tables, des textes…
- des outils **middleware** pour gérer les échanges
- Pour plus de détails, voir par exemple OTN (Oracle Technology Network)

4.3.2 Les Editeurs offrent des Progiciels-Composants Métier

Les Editeurs de logiciel font preuve de plus de transparence que dans le passé : un nombre croissant d'Editeurs publient leurs interfaces et leurs Modèles de données.
Les Editeurs de logiciel proposent non seulement des Progiciels qui informatisent des **Processus**, mais aussi des Progiciels qui informatisent des **Services-Logiciels.**
Les nouveaux Progiciel-Processus sont aujourd'hui construits sur une Architecture SOA.

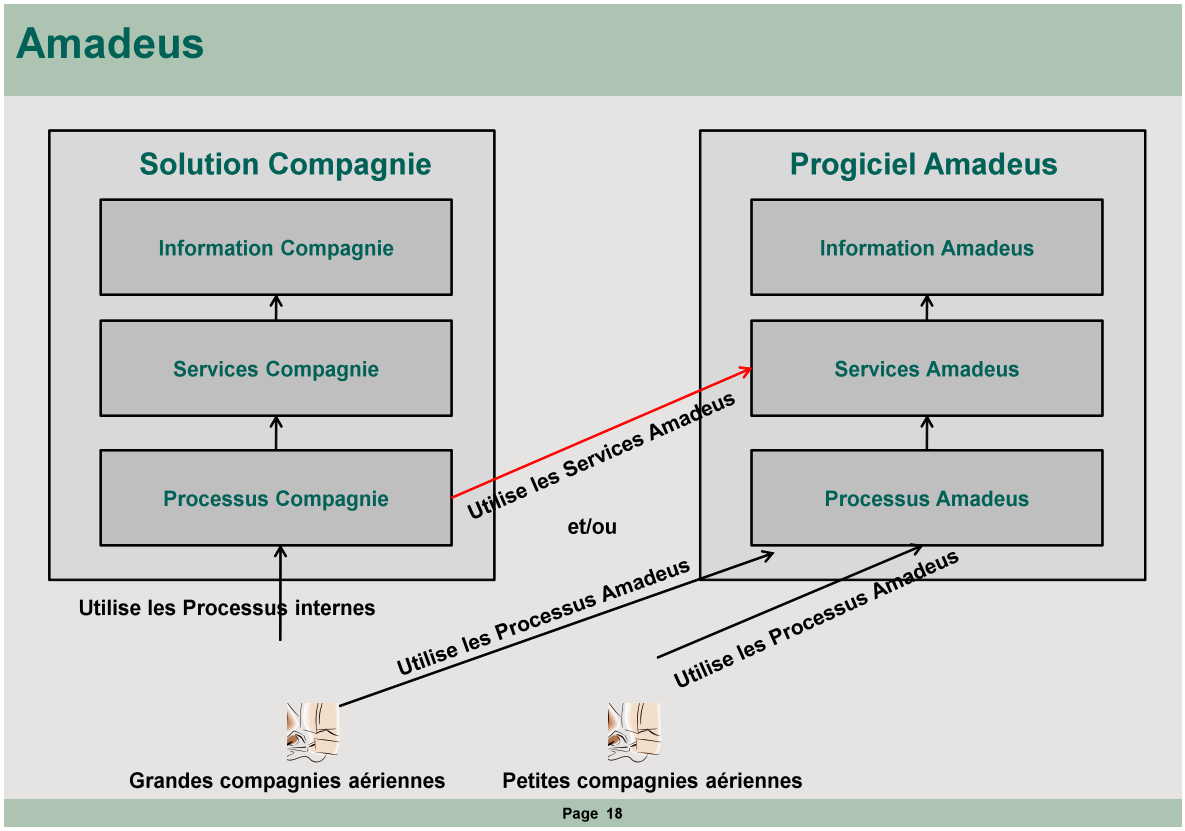


Il est alors possible non seulement d'exécuter directement les Processus de la Solution mais aussi de permettre à des Solutions externes d'appeler directement les Services proposés par le Progiciel.

Certains Progiciels en viennent même à ne proposer que des Services-logiciels (ou Composants) accessibles par d'autres Solutions.
C'est une offre qui fait de plus en plus de sens compte tenu de la nécessité de partager l'exécution des Processus avec des partenaires.

Cette offre devient **publique** : au-delà d'Oracle et SAP, voici quelques autres exemples.

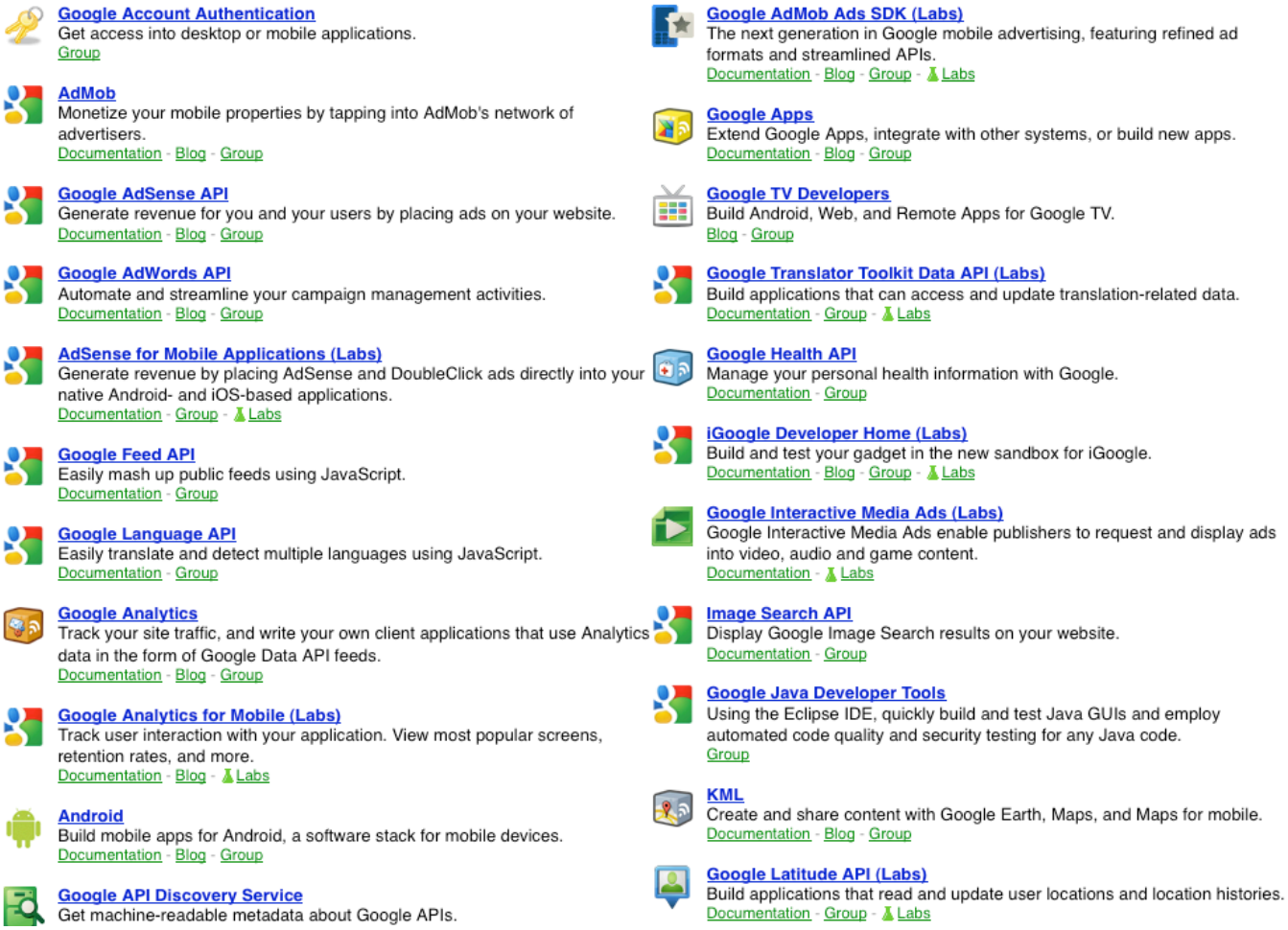
Offre Amadeus
L'offre d'Amadeus pour les compagnies aérienne (Altea CMS) couvre les domaines de l'inventaire, de la réservation et de l'enregistrement des passagers. Les clients peuvent soit utiliser le Progiciel-Processus tel quel, soit créer des Solutions appelant des Services-Logiciels hébergés par Amadeus. Un catalogue d'environ trois mille services a été publié pour les clients d'Amadeus.
Une grande compagnie aérienne préfère parfois utiliser les Services offerts par le Progiciel-Composants d'Amadeus, qui sont appelés par ses propres Solutions, ce qui permet de conserver une homogénéité d'usage.
Par contre, les petites compagnies aériennes utiliseront directement les Processus offerts par le Progiciel Métier d'Amadeus.



Ce scénario ouvert ne peut fonctionner que si les Editeurs sont capables d'isoler les Services dans leur offre. C'est certainement le cas pour les Progiciels conçus récemment. C'est beaucoup plus difficile pour les Progiciels anciens.
Ainsi, le métier du fret aérien dispose d'une offre riche de Progiciels existant depuis plusieurs décennies, bâtis sur des technologies anciennes. Ces produits sont à utiliser comme des boîtes noires car ils n'exposent pas leurs Services-Logiciels internes.

Offre Cegid
CEGID a développé une offre riche de Progiciels Verticaux spécialisés sur certains secteurs d'activité. L'approche mise en avant par l'éditeur est de faire évoluer son offre sur la base d'une Architecture de services ouverte permettant une interopérabilité maximale entre ses Solutions et avec celles de ses Clients. Cegid publie progressivement ses interfaces de Web Services.

Offre Google
Google publie ses interfaces sur le site http://code.google.com :



Google Account Authentication
Get access into desktop or mobile applications.
Group

AdMob
Monetize your mobile properties by tapping into AdMob's network of advertisers.
Documentation - Blog - Group

Google AdSense API
Generate revenue for you and your users by placing ads on your website.
Documentation - Blog - Group

Google AdWords API
Automate and streamline your campaign management activities.
Documentation - Blog - Group

AdSense for Mobile Applications (Labs)
Generate revenue by placing AdSense and DoubleClick ads directly into your native Android- and iOS-based applications.
Documentation - Group - Labs

Google Feed API
Easily mash up public feeds using JavaScript.
Documentation - Group

Google Language API
Easily translate and detect multiple languages using JavaScript.
Documentation - Group

Google Analytics
Track your site traffic, and write your own client applications that use Analytics data in the form of Google Data API feeds.
Documentation - Blog - Group

Google Analytics for Mobile (Labs)
Track user interaction with your application. View most popular screens, retention rates, and more.
Documentation - Blog - Labs

Android
Build mobile apps for Android, a software stack for mobile devices.
Documentation - Blog - Group

Google API Discovery Service
Get machine-readable metadata about Google APIs.

Google AdMob Ads SDK (Labs)
The next generation in Google mobile advertising, featuring refined ad formats and streamlined APIs.
Documentation - Blog - Group - Labs

Google Apps
Extend Google Apps, integrate with other systems, or build new apps.
Documentation - Blog - Group

Google TV Developers
Build Android, Web, and Remote Apps for Google TV.
Blog - Group

Google Translator Toolkit Data API (Labs)
Build applications that can access and update translation-related data.
Documentation - Group - Labs

Google Health API
Manage your personal health information with Google.
Documentation - Group

iGoogle Developer Home (Labs)
Build and test your gadget in the new sandbox for iGoogle.
Documentation - Blog - Group - Labs

Google Interactive Media Ads (Labs)
Google Interactive Media Ads enable publishers to request and display ads into video, audio and game content.
Documentation - Labs

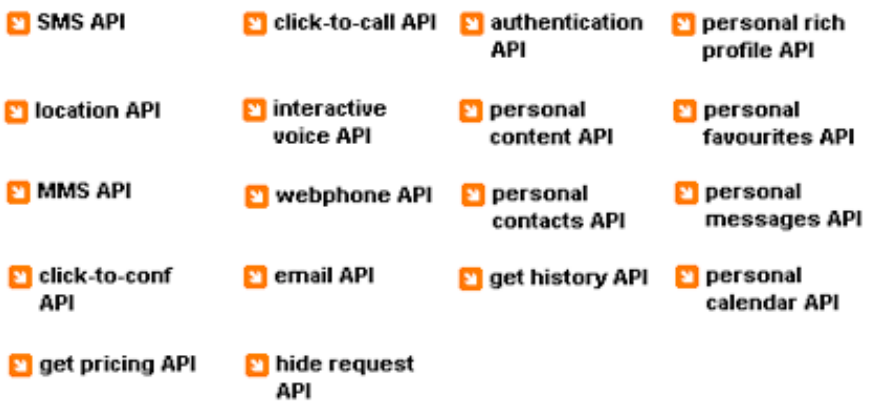
Image Search API
Display Google Image Search results on your website.
Documentation - Group

Google Java Developer Tools
Using the Eclipse IDE, quickly build and test Java GUIs and employ automated code quality and security testing for any Java code.
Group

KML
Create and share content with Google Earth, Maps, and Maps for mobile.
Documentation - Blog - Group

Google Latitude API (Labs)
Build applications that read and update user locations and location histories.
Documentation - Group - Labs

Offre Orange
Même les services de téléphonie peuvent être interrogés par appel de service, comme le montre cet exemple chez Orange (http://api.orange.com) :



4.4 Les Progiciels-Composants Techniques complètent l'offre

En complément, les Progiciels-Composants Techniques jouent un rôle significatif. En effet, il est nécessaire de couvrir certains aspects spécifiques au métier qui ne sont pas proposés par les Progiciels. A cet effet, il est possible de composer des Solutions spécifiques qui s'appuient sur des composants disponibles sans nécessairement avoir recours à de la programmation.

Exemple de l'Offre W4
La solution BUSINESS FIRST de W4 a pour objectif de fournir les outils à la DSI pour la création d'applications sur-mesure à destination du métier, en complément des Solutions en place. Pour ce faire elle propose des fonctions de modélisation des Processus métier, des modèles de données et des interfaces utilisateur pour construire des Solutions spécifiques. En complément, les services d'accès aux nouvelles fonctions proposées sont automatiquement mis à disposition sous forme de web services en vue de leur réutilisation.

4.4.1 Accès multiples au même Progiciel

Les premiers Progiciels n'étaient utilisables que par les employés de l'Entreprise cliente.
Les Editeurs ont progressivement ouvert leur offre pour qu'elle soit aussi utilisable par d'autres Acteurs de l'Entreprise (clients, prospects, partenaires) et d'autres médias : Internet, appareils mobiles (tablettes, smart phones,…).
Cette évolution pousse à la séparation de la logique métier et de l'interface utilisateur dans le Progiciel.

4.5 Les Editeurs offrent des Solutions à périmètre fonctionnel plus large

A vrai dire, les Editeurs sont confrontés au même problème que leurs clients : il vaut mieux peu de Solutions à large périmètre qu'une multitude de Solutions qu'il faut intégrer et faire évoluer.
C'est même encore plus difficile pour eux puisque leurs différentes Solutions sont installées dans différentes Entreprises appartenant à des secteurs différents.

Les ERP remplacent progressivement les Progiciels de Commodité par Domaine.
Les Editeurs préparent aujourd'hui des Progiciels Verticaux Intégrés qui seront dominant sur le marché demain.

Voir les paragraphes ci-dessus 2.4 et 3.6

4.6 L'Offre de Progiciels Verticaux se développe rapidement

Les Editeurs ont compris que le nouveau marché était celui des Verticaux et plus particulièrement des Verticaux Intégrés.
Le baromètre « Cockpit » établi chaque année par l'AFDEL, le CXP et Pierre Audouin Conseil fait apparaître que la demande de Progiciels Verticaux est particulièrement forte pour la Banque, l'Assurance, les Services aux Entreprises et les Utilities, c'est-à-dire très précisément les secteurs qui ne traitent que d'information, alors que la demande est stagnante pour l'Industrie, le Transport ou la Distribution.

Cette évolution s'accompagne d'un effort des Editeurs pour **étoffer leurs équipes avec des professionnels de chaque métier**, capables de spécifier les fonctionnalités attendues par le marché, de vérifier qu'elles sont bien implémentées et d'expliquer leur offre en termes compréhensibles par des non-informaticiens.

4.7 Maturation progressive du marché

Il y avait de nombreux éditeurs de Progiciels de paye dans les années 70 et 80 : ils ne sont plus très nombreux aujourd'hui. Trois phases se sont succédé :

- en face d'une demande Entreprise croissante, apparition d'une offre nombreuse de Solutions
- les meilleurs émergent progressivement
- phase de concentration : les plus grands éditeurs, qui ne sont pas forcément les plus innovants, acquièrent les Editeurs gagnants

Pour les Progiciels Verticaux, on n'en est qu'au début du Processus.

Qui va gagner pour les Progiciels Verticaux Intégrés ? C'est une question extrêmement difficile.
Nous nous contentons de donner quelques pistes :

Ce qui plaide en faveur des grands Editeurs d'ERP actuels :

- Les Editeurs qui ont réussi à intégrer différents Progiciels de Commodité dans un ERP ont gagné de la maturité dans la construction d'une solution intégrée : c'est un plus.
- La taille des grands Editeurs leur donne des moyens financiers pour investir et rassure les Entreprises.

Mais les jeux sont loin d'être faits :

- Avoir réussi à construire un ERP pour lequel les besoins sont similaires d'une Entreprise à l'autre ne signifie pas forcément que l'on est capable de construire un Vertical Intégré hautement personnalisable par Configuration. Les méthodes et techniques pour construire un Vertical Intégré sont plus sophistiquées que celles nécessaires pour un ERP, tout particulièrement si on souhaite personnaliser le Progiciel par Configuration, et bien peu les maîtrisent aujourd'hui. Un prototype de personnalisation est nécessaire au moment du choix de Progiciel pour mesurer cette maturité (voir plus loin, le chapitre sur le choix de Progiciels).
- La taille peut être un atout mais aussi un handicap dans la phase d'émergence de l'offre où on innove ; les procédures en place dans les grandes organisations ne sont généralement pas compatibles avec l'agilité nécessaire dans cette phase ; nombreux sont les produits provenant des plus grands fournisseurs qui ont finalement été abandonnés.
- La connaissance profonde du métier est un plus. Le plus souvent, le Progiciel est construit avec un client pilote, puis généralisé pour être offert à ses confrères. A titre d'exemple, SAP a développé une Solution pour le métier « Câbles » avec Nexans, avant d'en faire un Progiciel Vertical. Il est difficile pour un Editeur généraliste d'être le meilleur dans tous les métiers. Pour prendre un exemple, Amadeus est plus petit que SAP ou Oracle mais a plus de chance de devenir un Acteur prépondérant d'un Vertical Intégré pour les compagnies aériennes.

Point de vue du CEISAR

Pendant la phase d'émergence des Progiciels, les Entreprises doivent être attentives à la qualité du Progiciel plus qu'à la taille du fournisseur. Si une Entreprise taille des croupières à ses concurrents grâce à un bien meilleur « time to market » provenant d'un Progiciel innovant, alors la taille de l'Editeur importera peu, le succès ira vers l'Editeur le plus efficace.
L'information fournie par les analystes ne peut qu'accélérer ce Processus de sélection naturelle.
Une fois les meilleurs Progiciels identifiés, une phase de consolidation sera inéluctable.

4.8 La personnalisation par configuration devient plus puissante

Les possibilités de personnalisation par configuration croissent.
La première forme de Configuration est le paramétrage.
Mais la pression pour réduire les besoins en développements spécifiques, a conduit à accroitre les possibilités de configuration :

- paramétrage :
  - tables à 1 ou 2 ou n dimensions,
  - tables de paramètres et/ou tables de règles
- moteur de règles
  - certains Editeurs de Progiciels utilisent des moteurs de règles externes qui proviennent d'autres Editeurs : il faut alors synchroniser les mises à jour des deux mondes (le Progiciel et l'environnement du moteur de règles), à chaque évolution fonctionnelle, que ce soit pour synchroniser la logique ou les données
  - d'autres ont réussi à **intégrer** le moteur de règles au cœur de leur Progiciel et à offrir pour chaque type de règle une réduction du périmètre de données utiles qui facilite le travail de ceux qui définissent ces règles. Un prototype illustre parfaitement la différence entre un moteur de règles intégré et un moteur de règle dissocié.
- moteur de workflow qui peut couvrir deux domaines

    - externaliser la **logique** d'enchainement des taches (le « quoi »), indépendamment de qui exécute des tâches
    - externaliser la logique **d'affectation** des tâches aux différents Acteurs (le « qui et quand ») pour s'adapter à chaque organisation
- attributs dynamiques
    - un cas souvent délicat est la nécessité pour une Entreprise d'ajouter des Attributs non proposés par le Progiciel, ce qui oblige à revoir la structure de la base de données et à gérer ses conséquences sur les logiciels qui l'utilisent. Une technique proposée par certains éditeurs consiste à insérer dynamiquement des attributs dans des Objets métiers existants par simple configuration, ce qui met à disposition ces attributs comme nouveaux paramètres pour les règles. Cette facilité permet de faire évoluer le Modèle d'Information sans solliciter les informaticiens.

Ces nouveaux mécanismes permettent de personnaliser le Progiciel dans des domaines aussi divers que :

- la langue
- la devise
- les types tels que nom, adresse
- les règles fiscales
- les éditions
- les rapports
- la tarification
- le scoring
- les interfaces
- …

Ainsi Amadeus a choisi de développer un haut niveau de paramétrage dans son offre Altea CMS grâce à un moteur de règles **interne**, qui permet de remplacer une part importante de la logique métier par des règles, et de prioriser les règles entre elles en fonctions du contexte des données. C'est le même cas pour Wyde, Editeur de Solution d'Assurance, qui a intégré un Moteur de Règles natif dans son Progiciel. A l'inverse, Oracle et SAP (Netweaver Business Rules Management qui est le fruit de l'acquisition de la société Yasu) s'appuient sur un moteur de règle **externe**.

4.9 Progiciel prérempli

La pression des Entreprises sur l'agilité, conduit les Editeurs à livrer non seulement des logiciels, mais aussi des **données** : données de configuration (par exemple rentrer des Modèles de Produits, des Modèles de plan comptable, des Modèles de rôles, des Modèles de Processus…). L'Entreprise travaille alors par différence : elle conserve les données qui lui conviennent et modifie celles qui lui sont spécifiques. C'est un Processus beaucoup plus rapide et plus concret, qui permet d'éviter le syndrome de la feuille blanche.
Ainsi, Amadeus propose son offre de gestion de la réservation et de l'embarquement des passagers préparamétrée, ce qui permet un démarrage accéléré pour les compagnies qui l'adoptent.

4.10 Les Editeurs proposent des offres « Cloud »

Le Cloud Computing est le début de concrétisation de ce rêve un peu fou de pouvoir se brancher à l'informatique, comme on le fait pour se brancher au réseau électrique avec une prise. Mais on n'en est pas encore là.

Par « Cloud » on entend :

- **SaaS** Software as a Service (Progiciel Internet fourni et exploité à l'extérieur)
- **IaaS** Infrastructure as a Service (Exploitation externe de Solutions internet)
- **PaaS** Platform as a Service (Plate-forme de développement externalisée)

Les rôles de chaque Acteur sont résumés dans le tableau joint.
Le même Fournisseur peut jouer plusieurs rôles : Editeur d'outils de développement, Editeur de Progiciel Solution ou Opérateur : c'est le cas, par exemple, de Sales Force pour son offre SaaS.

			SaaS	IaaS	PaaS
Transformation	Qui propose l'environnement de développement ?		Editeur d'outil de développement	N/A	Editeur d'outil de développement
	Qui développe?	Spécifique	Editeur de Progiciel-Solution	N/A	Client
		Configuration	Client	N/A	Client
	Qui exploite l'environnement de développement ?		Editeur ou Opérateur	N/A	Opérateur
	Qui définit l'infrastructure d'Exploitation? (OS, Middleware, SGBD…)		Editeur de Progiciel-Solution	Client	Opérateur
Opérations	Qui exploite les Solutions Métier?		Client	Client	N/A
	Qui exploite l'infrastructure d'Exploitation ?		Opérateur	Opérateur	N/A

Comme ce livre blanc est consacré aux Progiciels, on se concentre sur l'offre SaaS : Progiciels Internet exploités à l'extérieur.

La Valeur de l'offre Saas est liée à deux atouts :

- faire face aux **contraintes de charges :** les offres Cloud permettent de banaliser la ressource informatique en permettant son achat au plus juste, selon les besoins, en la rendant disponible rapidement pour un coût proportionnel à l'usage. Elles autorisent également des pics de charge ponctuels et des montées en charge rapides. Il est à noter cependant que si une application tourne 100% du temps et occupe une machine à 80%, il est inutile de la mettre sur le Cloud !
- se **décharger** à la fois du développement et de l'exploitation
- bénéficier en permanence des dernières fonctionnalités des applications disponibles en SaaS sans avoir à effectuer les pénibles montées de versions

Mais elle impose les contraintes suivantes :

- difficile d' **interfacer** la Solution SaaS avec les autres Solutions de l'Entreprise qui ne sont pas dans le même Cloud: on est limité par le débit Internet et par la complexité à gérer en parallèle différents Clouds. A titre d'exemple il est difficile d'imaginer que l'on traite le batch de facturation sur le Cloud, si les bases Contrats » sont sur les mainframes de l'Entreprise.
- Minimum de **personnalisation** du Progiciel puisque le fournisseur a intérêt à standardiser la Solution qui est exploitée pour plusieurs : les Solutions de Commodité sont donc davantage candidates au Cloud que les Solutions Verticales. On note qu'en moyenne les solutions SaaS sont moins riches que les Progiciels classiques en possibilités de personnalisation, ce qui explique en partie la rapidité de leur mise en œuvre.
- Résoudre les problèmes de **sécurité** et de propriété des informations lorsqu'elles sont hébergées à l'extérieur.
- Adapter les Acteurs internes chargés de l'exploitation informatique.

Les Solutions Saas semblent donc bien adaptées à deux mondes :

- à un ERP pour **PME/TPE** : peu de personnalisation, peu d'interfaces, apprécient d'externaliser la fonction informatique
- à des **Solutions de Commodité isolées pour toutes Entreprises**, pourvu que ces Solutions soient provisoires ou autonomes ou que l'Entreprise accepte une approche « silo » dans laquelle les Solutions ne communiquent pas. On peut citer : messagerie, outils collaboratifs d'Entreprise, ressources humaines, comptabilité, achats…

D'après le Gartner, seulement 10% des investissements IT en 2015 seront « *on demand ».*
D'après Forrester (« the state of ERP 2011 »), les ventes d'applications SaaS devraient évoluer au rythme de 20 % par an mais resteront à un niveau relativement modeste représentant environ 5 % de l'ensemble des ventes de licence.

Ce taux augmentera lorsque les standards d'interopérabilité auront été définis, puis traduits par les fournisseurs de Cloud dans leur offre, ce qui devrait prendre un certain temps.

4.10.1 Réutilisation de Services sur le Web

Il existe aujourd'hui des Services-logiciels accessibles sur le web (ex : infos bourse, météo, plan, état des pistes, Amadeus…) que l'on peut intégrer à l'Architecture. A titre d'exemple, TOTAL intègre dans son nouveau portail RH des Services externes pour la météo ou le cours de bourse de l'action TOTAL. AXA intègre de Services Web de DARVA, spécialiste de la gestion des échanges entre les différents intervenants de la gestion des sinistres.

Comment les récupérer, et les orchestrer avec le modèle en place pour en tirer un avantage concurrentiel ?

La recommandation est de se doter d'un module d'accès intermédiaire dont le rôle est

- de **convertir les informations** reçues et transmises aux normes de l'Entreprise pour **simplifier leur accès** par les différentes Solutions
- de **standardiser le mode l'échange**, via un EAI ou un ESB qui sera en charge par exemple de transformer un WebService en requête sur le Web
- de faire du **cache** ou de la bufferisation pour les informations qui sont souvent sollicitées
- de traiter les problèmes de **sécurité**

En résumé, le module s'approprie les informations externes puis les met à disposition des Solutions qui souhaitent les consommer. Cette approche permet aussi de limiter les adhérences avec les services externes et d'en changer plus facilement.

4.11 Les Progiciels internationaux se développent

Les Progiciels sont progressivement devenus internationaux.
Cette tendance est facilitée par la mondialisation qui rapproche les cultures, les produits, les Processus et les modes d'organisation.
C'est plus difficile pour les **Progiciels Verticaux** que pour les ERP, lorsqu'un « Vertical » possède de nombreuses spécificités locales. La banque, l'assurance, la santé, l'enseignement, la défense, la justice, … sont des domaines où les spécificités nationales (réglementaire, fiscalité, échanges interprofessionnels) sont fortes. Il faut donc que les Progiciels associés aient une importante capacité de personnalisation (voir ci-dessus). Pour que l'Editeur puisse faire évoluer rapidement son Progiciel sans être contraint par les multiples versions locales qu'il faut entretenir, on doit encore une fois s'appuyer sur des mécanismes de Configuration puissants : ces mécanismes sont la clé de la réussite des Progiciels Verticaux.

4.12 La Transparence progresse

Les Editeurs ont abandonné l'attitude de protection qui les caractérisait : ne pas donner d'informations sur les internes du Progiciel pour ne pas risquer d'être copié ou pour ne pas donner l'opportunité au client de développer ou d'interfacer des compléments fonctionnels qui diminueraient le périmètre relatif du Progiciel.
Les Editeurs ont compris que les Clients avaient besoin de transparence : ils communiquent beaucoup plus sur leur offre, en particulier sur le Modèle d'information, sur les Services-logiciels offerts, sur les Interfaces disponibles, sur les standards qu'ils respectent.
La documentation disponible augmente beaucoup mais les clients ont parfois du mal à trouver **l'information essentielle**, c'est à dire les éléments structurants de l'Architecture du Progiciel.
A titre d'exemple, les Architectes de TOTAL ont regretté le manque d'informations synthétiques de la part de SAP, et AXA France aurait souhaité une documentation complète sur le Modèle d'Information de Guidewire.

4.13 Les Editeurs fournisseurs de Fondation

Rappelons qu'une Fondation est l'ensemble des outils de développement et des Services-Logiciels réutilisables qui permettent de construire des Solutions Spécifiques. 80% d'une Solution spécifique peut

être prédéveloppé sous forme de composants (voir le livre blanc du CEISAR sur le Fondation www.ceisar.org).
Les Editeurs peuvent non seulement délivrer des Progiciels, ils peuvent aussi délivrer leur Fondation qui sert de base à la construction de leur Progiciel. Les Entreprises clientes peuvent alors développer leurs Solutions spécifiques sur la même Fondation que les Progiciels.
L'intérêt pour les Entreprises est qu'elles n'ont pas à investir dans une Fondation propre, et que l'interopérabilité sera facilitée avec les Progiciels du même fournisseur.
L'intérêt de l'Editeur est un produit supplémentaire à vendre ; il est aussi de favoriser la réussite des développements de ses clients.
La contrepartie est que l'Editeur doit documenter et diffuser le contenu de cette Fondation pour que l'Entreprise puisse juger de son efficacité et de son adéquation : à nouveau un besoin de transparence.

4.14 L'extension du rôle de l'Editeur

La maîtrise des Progiciels Verticaux Intégrés va mettre en valeur les Editeurs qui arrivent à proposer une offre à spectre large : plusieurs lignes produits, plusieurs domaines fonctionnels, plusieurs pays...
Le fait d'avoir réussi à modéliser sous forme de logiciel toutes les finesses d'un métier Vertical signifie qu'ils possèdent la connaissance intime du métier, et ce de manière beaucoup plus rigoureuse que ceux qui ne peuvent s'appuyer que sur une modélisation documentaire, comme les cabinets de conseil.
Ils vont progressivement devenir des interlocuteurs privilégiés pour leurs clients, qui ne peuvent fédérer des filiales internationales sans Solution internationale, réduire le « time to market » sans un Product Factory qui s'appuie sur une modélisation détaillée du métier, accroître la productivité sans automatiser des Processus de bout en bout.
Les interlocuteurs essentiels des Entreprises que sont les Sociétés de Conseil et les SSII devraient se rapprocher des Editeurs sous forme de partenariats étroits voire d'acquisitions, pour bénéficier de la connaissance modélisée dans les Progiciels.
De la même façon, les fournisseurs de « BPO » qui cherchent à offrir des services complets de Middle Office aux Entreprises, vont vouloir acquérir un Progiciel Vertical suffisamment intégré et efficace pour qu'il leur donne un avantage concurrentiel. Rappelons qu'un Progiciel Vertical bien intégré et international signifie :

- homogénéité de l'interface utilisateur : il est alors plus facile de rendre les utilisateurs polyvalents et de répartir la charge en fonction des pointes d'activité
- pas de double saisie, pas de données dupliquées
- possibilité de saisir les informations dans différentes langues
- facilité de gestion de version puisqu'il n'existe qu'un produit

5 L'Entreprise doit maîtriser son Modèle cible

Le rapprochement de la demande des Entreprises et de l'offre des Editeurs nous laisse entrevoir un avenir mieux maitrisé pour les Progiciels Verticaux.
Comment l'Entreprise peut-elle en tirer parti ?
Nous pensons qu'utilisation d'un Progiciel extérieur ne signifie pas abandon de la maîtrise de son système d'information : l'Entreprise doit maitriser son Architecture de Solutions si elle veut faire les bons choix, être consciente des forces et faiblesses des Progiciels choisis, intégrer ces Progiciels et les personnaliser, et conserver une certaine indépendance vis-à-vis des Editeurs. La démarche qui permet de bien intégrer différentes Solutions est la même que les Solutions soient spécifiques ou Progicialisées.

RÉSUMÉ

Ce n'est pas parce qu'on utilise des Progiciels qu'on abandonne pour autant la démarche d'Architecture d'Entreprise. Il faut :
- Définir ses objectifs d'Architecture d'Entreprise
- Concevoir l'Architecture d'Entreprise cible
  - Définir sa **cartographie fonctionnelle** pour connaître ses principales fonctions et leur nature (Commodité ou Vertical)
  - Avoir une **approche Processus**
  - Définir sa méthode et ses outils
  - Disposer d'une **cartographie des Solutions** pour mieux situer le Progiciel par rapport aux autres Solutions avec lesquelles il doit coexister.
  - Disposer d'un **Modèle d'information** qui sert de base pour définir les échanges.
- Détailler les besoins **d'interopérabilité**
  - Quelles Solutions sont maîtres des Informations ?
  - Quelles Solutions doivent conserver l'interface-utilisateur déjà en place ? (ex : distributeurs habitués à un mode d'usage, et qui doivent distribuer un nouveau produit)
  - Quelles Solutions ont besoin de fonctions offertes par d'autres Solutions ?
  - En déduire les interfaces
  - Quel Middleware et quelle sécurité ?

5.1 Définir ses objectifs d'Architecture d'Entreprise

L'Entreprise doit définir quelle cible d'Architecture d'Entreprise elle souhaite atteindre d'ici cinq ans :

- faut-il réduire le nombre de **Solutions** pour diminuer la complexité, ce qui suppose d'utiliser des Solutions plus vastes ?
- comment décider entre acquisitions de Progiciels ou développements spécifiques ?
- faut il favoriser une **Fondation** pour construire une majorité de Solutions ?
- faut-il concentrer les informations dans des **référentiels partagés** (approche « MDM ») ?
- quelle part pour la réutilisation de Services-logiciels ?
- faut il réserver les nouvelles Solutions au **nouveau business** (nouveaux produits, nouveaux territoires, nouvelles cibles de clientèle) et conserver les Solutions qui traitent l'ancien business?
- faut-il réduire le nombre **d'interfaces** entre Solutions ou ajuster chaque Interface à chaque type d'échange ?
- faut-il remplacer des interfaces point à point par un bus de services ?
- comment partager les investissements entre amélioration des Solutions en place et installation de nouvelles Solutions ?
- faut-il utiliser un Progiciel-Composant Technique pour répondre aux fonctions spécifiques ?
- faut-il utiliser les bases de données « in-memory » pour fusionner les bases de données opérationnelles et décisionnelles ?

Les réponses ne sont pas évidentes.

Mais ceux qui ont défini une cible claire compatible avec leur stratégie auront une trajectoire plus directe.

Point de vue du CEISAR

- Chercher des Solutions à périmètre **large**
- Favoriser une **Fondation** qui peut être celle d'un Editeur et les Services associés
- Développer des **référentiels** partagés, même si on souhaite bénéficier de mécanismes de réplication pour assurer une autonomie Opérationnelle
- Chercher des outils qui permettent de **générer automatiquement** des **Interfaces** ajustées

5.2 Définir sa cartographie fonctionnelle

L'Entreprise doit définir sa cartographie fonctionnelle, sous forme d'une classification hiérarchique de ses fonctionnalités.
Elle doit en particulier bien **distinguer** les fonctionnalités de Commodité des fonctionnalités Verticales.
Rappelons qu'utiliser un Progiciel de Commodité pour des Fonctions Verticales conduit à des personnalisations excessives et coûteuses qui ne permettent pas de gérer aisément les versions successives du Progiciel.
Il ne faut pas hésiter à **décomposer** les Processus essentiels en une partie **Commodité** et une partie **Verticale**.
A titre d'exemple, Air France a éclaté son Processus commercial en deux parties : une partie Commodité pour la prospection commerciale développée avec le Progiciel de Commodité CRM de SalesForce.com, et une partie Verticale pour la prise de commande développée en interne.

5.3 Mettre à profit une approche Processus

L'Entreprise doit connaître ses Processus Métier et les cartographier en distinguant :

- Les Processus Primaires
- Les Processus de gestion des ressources (Support)
- Les Processus de pilotage

Elle doit également savoir distinguer les Processus ou parties de Processus sources d'avantage concurrentiel.

Les Entreprises engagent de plus en plus fréquemment des démarches d'excellence opérationnelle, basées sur les Processus. Ces derniers sont identifiés et décrits, puis optimisés. Dans les Entreprises les plus avancées, le management par les Processus peut être adopté : l'Entreprise adapte son organisation autour des Processus en nommant des pilotes de Processus, chargé de la définition et de la bonne exécution des Processus Métier (se référer aux travaux du Club des Pilotes de Processus pour une vision plus approfondie).

Il est clé de bien intégrer les Progiciels dans la démarche Processus et de comprendre leur impact, les bénéfices et les contraintes qu'ils apportent.
Avoir une bonne description de ses Processus Métier permet en outre à l'Entreprise de mieux sélectionner ses Progiciels.

Point de vue du CEISAR

L'Entreprise doit rester ouverte aux apports de l'Editeur de Progiciel, qui a modélisé les Processus de l'Entreprise en échangeant avec de nombreux Clients. Au moment du choix et de la personnalisation, l'Entreprise doit donc s'interroger sur les Processus qu'elle souhaite absolument exécuter comme elle les avait Modélisés et ceux qu'elle est prête à adapter pour coller au Progiciel. Pour guider ce choix, nous invitons les Entreprises à Modéliser à deux niveaux :

- **Le Processus Métier** est décrit à un niveau générique, centré sur le « quoi ? », c'est-à-dire quelles Actions on doit réaliser pour produire la valeur attendue par le Client du Processus. Cette description se concentre sur les grandes étapes invariantes du Processus. On la réalise en général par abstraction de nombreux cas rencontrés sur le terrain. Par exemple, en analysant plusieurs formes différentes de Processus de souscription d'un contrat, on saura identifier les Activités majeures : identifier le Client, Identifier les Produits et options souscrits, Calculer un tarif, Soumettre la proposition au Client puis Enregistrer le contrat.
- **Le Processus Organisé** précise la répartition des tâches sur les différents Acteurs de l'Entreprise. Il définit le « qui ? » et le « comment ? » en détaillant le Processus Métier dans le cadre d'un scénario d'organisation particulier. C'est le Processus tel qu'il s'exécute vraiment sur le terrain à l'instant t. On parle aussi parfois de « procédure ». D'une Entreprise à l'autre, les choix d'organisation peuvent varier et un même Processus Métier peut s'implémenter dans des Processus Organisés différents.

Notre recommandation est donc de s'appuyer sur la description des Processus Métier pour faire le choix du Progiciel. Ce dernier doit être capable de réaliser les Fonctions Métier essentielles du Processus Métier et avoir les capacités de personnalisation suffisantes pour le décliner en différents Processus Organisés.
Par ailleurs, l'Entreprise peut aussi envisager de remettre en cause ses Processus Organisés pour adopter ceux du Progiciel, s'ils ne remettent par en cause le cœur du métier ou si elle estime qu'ils sont plus optimisés que les siens.

5.4 Définir sa méthode et ses outils d'Architecture d'Entreprise

Chaque Entreprise se doit de définir sa méthode et ses outils pour définir son Architecture d'Entreprise globale.
Par contre, l'Entreprise doit s'appuyer sur la méthodologie proposée par l'Editeur pour installer un Progiciel.
En effet les Editeurs connaissent bien leur produit et ont accumulé de l'expérience sur les bonnes pratiques.
A titre d'exemple, SAP propose sa propre méthodologie, « Enterprise Architecture Framework », qui a été créée en partenariat avec Cap et en réutilisant des principes du Togaf.

5.5 Disposer d'une cartographie des Solutions (« Domaining »)

5.5.1 But

Traditionnellement, les Progiciels étaient autonomes et n'avaient pas à coexister avec d'autres Solutions. On n'avait donc pas besoin de cartographie globale.
Aujourd'hui, on cherche à intégrer le Progiciel dans l'Architecture d'Entreprise. Il faut donc un Modèle qui représente ces différentes Solutions et la façon dont elles échangent.

5.5.2 Contenu

- lister les **Solutions**
- déterminer qui est maitre de **l'information**
- déterminer quelles Solutions implémentent quels **Processus** et **Fonctions**

Dans les Modèles qui contiennent de nombreuses Solutions différentes, on cherche à outiller la cartographie des Solutions pour mieux comprendre les conséquences des modifications d'interface sur les Solutions qui l'utilisent.

Lorsque la même **Information** est utilisée par plusieurs Solutions, il faut décider comment les autres Solutions accèdent à l'information :

- duplication d'information
- ou Services d'accès aux référentiels

Lorsque la même **Fonction** est exécutée par plusieurs Solutions, il faut décider

- soit de la dupliquer : mais attention aux coûts de développement et aux synchronisations de mise à jour
- soit de désigner quelle est la Solution maître qui va exposer le Service auprès des autres Solutions

Dans certains secteurs d'activités (télécoms, aérien, …), les Entreprises essaient de normaliser une cartographie fonctionnelle avec des blocs applicatifs au périmètre et aux interfaces clairement définis. Cela leur permet de faire correspondre facilement les offres Progicielles du marché avec leur cartographie de Solutions. Elles espèrent alors pouvoir intégrer dans leur existant des Solutions dans une approche « plug & play ». A titre d'exemple, on pourrait citer la brique « Enregistrement » du secteur des transports, qui correspond à un périmètre précis avec des interfaces identifiées et sur lequel l'Entreprise pourrait positionner des Solutions interchangeables : un développement spécifique, un Progiciel ou une Solution SaaS. C'est aussi le cas dans le secteur des télécoms avec la cartographie fonctionnelle du modèle eTOM.

5.6 Disposer d'un Modèle d'information qui sert de base pour définir les échanges.

5.6.1 But

La majorité des études de cas fait apparaître la nécessité d'un Modèle d'Information préalable à l'acquisition de Progiciels.

Après avoir utilisé des méthodologies exclusivement basées sur l'analyse de Processus, les Entreprises reconnaissent aujourd'hui que le Modèle d'information est une nécessité pour atteindre une bonne cohérence d'ensemble.
En résumé il faut définir :

- un glossaire des termes du métier (compter une cinquantaine d'Objets par métier)
- les relations entre ces Entités Métier
- le contenu de chaque Entité : les attributs
- les types communs : comment représenter un nom, une adresse, un identifiant, un produit…

Les Editeurs qui s'appuyaient sur des Modèles d'Information propriétaires se coordonnent pour proposer un Modèle d'Information commun.
L'initiative « Open data protocol » (www.odata.org) est un projet pour définir des objets partagés par l'ensemble de l'écosystème.
SAP a lancé en mai à Orlando son nouveau « Project Gateway » qui définit les objets du métier susceptibles d'être échangés
(voir :http://www.bluefinsolutions.com/insights/guest_blog/project_gateway_a_call_to_arms_or_at_least_to_data/)
Dans les cinq ans, Oracle disposera d'un MDM unifié et d'une modélisation des données homogène qui fera la synthèse de tous ses modèles de données. Ce produit sera vendu en stand-alone (aujourd'hui il y en a plusieurs : un pour le client, un pour le produit, etc.).

Que le Modèle d'Information choisi soit celui d'un Editeur ou de l'Entreprise, ce Modèle doit servir de base pour les interfaces entre les Solutions.
L'idéal est que chaque Solution s'adapte à ce standard dans sa « demi-interface » et que les flux transportés respectent ce Modèle d'information.
Si ce n'est pas possible, parce que les interfaces sont imposées par la Solution couplée, une traduction est à faire dans l'autre Solution.

5.6.2 Définir quelles Solutions sont maîtres des Informations

Déterminer quelles Solutions sont **maîtres** de quelles **Informations** : faut-il partager ces Informations avec d'autres Solutions ?
Faut-il partager par réplication ou par utilisation de Services d'accès aux référentiels ?

5.6.3 Vers une fusion des Bases de données Opérationnelles et Décisionnelles

Une grande partie des interfaces à développer consiste à alimenter les Solutions de pilotage, ou Solutions décisionnelles.
Une nouvelle tendance risque de faciliter la vie des Entreprises, qui consiste à fusionner les bases de production et les bases décisionnelles.
De grands éditeurs comme SAP (initiative HANA), IBM, Oracle innovent dans des bases de données mixtes qui supporteront à la fois des besoins opérationnelles et des besoins décisionnels. Le principe est de stocker et rechercher directement en mémoire les informations nécessaires.
L'utilisation de ces bases de données nécessitera une refonte progressive des Progiciels.

Commentaire de SAP sur HANA
Les prix des mémoires baissent, elles sont 10 000 fois plus rapide qu'un disque, on peut les adresser en 64 bits sur tous les serveurs, elles sont devenues persistantes.

Les deux modèles, données transactionnelles et données décisionnelles vont donc être stockés totalement « in memory » avec des algorithmes de compression plus performants en mémoire que sur disque.
HANA est une base de données SAP « in memory » qui comprend un moteur de calcul haute performance, un moteur de réplication et trois standards d'accès.
C'est la base de données généralisée, globale, multifonctions, accessible en temps réel ou en batch, qui joint le décisionnel et le transactionnel dans un même stockage avec un accès cent à deux cents fois plus rapide

Depuis le 1er décembre 2010, les clients peuvent rajouter HANA en parallèle avec la base de données traditionnelle pour pouvoir faire de la Business Intelligence temps réel sur l'opérationnel et de la consolidation comme on le ferait avec un entrepôt de données.

Juin 2011 : l'entrepôt de données BW sera sur HANA ; il n'y a plus d'ETL et le lien entre les données se fait sur le même socle.

En 2012, l'ERP sera directement connecté à Hana et on pourra manipuler directement les objets métiers.
- *100 fois plus rapide*
- *Cela écourte le chemin entre le transactionnel et le décisionnel puisqu'il y aura des calculs in memory entre les données.*

En 2013 d'autres applications pourront utiliser SAP HANA

Aujourd'hui, 50 clients en ramp-up !! Cela adresse tous les grands comptes. C'est une rupture

SAP considère qu'il a au moins deux ans d'avance, et que ses concurrents essentiels sont :
- *HP a fait une acquisition…*
- *IBM : SolidB (pas encore mature) et rachat de NETEZZA (in memory)*
- *ORACLE : Hexadata (BD relationnelle optimisée sur du hardware)*
- *Teradata vielle technologie*

L'Open Source a publié « MySQL lite » pour du relationnel in memory

5.7 Interface utilisateur homogène

« Même interface utilisateur » ou « même mode d'usage » signifie :
- même présentation
- même navigation
- identification et authentification uniques
- types de données standardisés

La cohérence d'usage a pour vertu non seulement de diminuer les efforts de déploiement et d'accroître la productivité des Acteurs opérationnels, mais aussi de faciliter l'évolution de leur rôle : il est plus aisé de leur proposer un autre poste, de répartir la charge de travail, de leur donner plus de responsabilité, s'ils retrouvent le même mode d'usage quelque soit l'activité effectuée.

Comment faire :

- il faut identifier **quels Processus** peuvent ou pourront être utilisés par le **même utilisateur**, puis vérifier si l'on peut réduire la diversité d'usage pour cet utilisateur.
- une autre méthode consiste à offrir le **même mode d'usage pour toutes les Solutions**, ce qui permet de faire évoluer l'organisation au fur et à mesure des initiatives, sans avoir à planifier qui fera quoi demain

Dans le premier cas, on peut conserver le mode d'usage existant.
C'est le cas, par exemple, de distributeurs externes habitués à un mode d'usage standardisé, et qui doivent distribuer un nouveau produit : on conserve alors l'interface utilisateur d'origine et on enrichit la Solution existante avec des **appels de nouveaux Services.**

Lorsqu'une Entreprise assemble plusieurs Solutions distinctes en une Solution plus large, elle doit généralement reconstruire l'interface utilisateur.
Un bon exemple est fourni par Air France, qui a remplacé trois Solutions distinctes (gestion des plans de vol + suivi des vols + affichage public) par une Solution cohérente dont l'interface utilisateur a été harmonisée.

5.8 Identifier pour chaque Solution quelles fonctions elle offre aux autres Solutions : l'approche SOA

Certaines Fonctions sont logées dans une Solution sous forme de **Services**-Logiciels, mais peuvent être appelées par une autre Solution.
Par exemple, la Fonction de calcul de tarif peut être logée dans la Solution CRM qui en a besoin pour faire des propositions, mais utilisée aussi par la Solution gestion de contrat ou la Solution facturation.
L'appel de ces Fonctions peut s'effectuer de façon synchrone (comme le calcul de tarif) ou asynchrone (comme la prise en compte d'un évènement comptable).
La mise à disposition de Services réutilisables a des avantages bien identifiés :

- on ne développe le Service qu'une fois : économie de développement
- on ne le met à jour qu'au sein d'une seule Solution : tous les utilisateurs utilisent la même version
- on peut gérer des référentiels partagés en offrant des services de lecture et mises à jour de ces référentiels
- on peut conserver une interface utilisateur stable et enrichir la Solution offerte par appel de Services (voir ci-dessus)
- on peut faire appel à des Services publics

La contrepartie en est que le Service doit être construit pour être utilisé par plusieurs clients, et que la dépendance entre différentes Solutions crée des exigences plus fortes en matière d'exploitation opérationnelle.
Mais la multiplication de ces Services nécessite une approche et des outils.
Dès que l'on atteint quelques centaines de Services on doit les référencer dans un repository de Services qui gère leur « contrats », leurs **versions** successives (fonctionnalité offerte et contrat d'interface), leur **utilisation** et on doit standardiser l'accès à ces Services (**Middleware**).

Approche de SAP
L'Enterprise Service Repository – ESR – et l'Architecture orientée services de Netweaver permettent de produire et de consommer des Services et de gérer la gouvernance.
On peut consommer directement un service à partir de l'ERP, sans mettre aucune infrastructure, mais cela crée une dépendance dangereuse ; il est plus simple d'utiliser le logiciel (avec ou non un ESB), et de consommer par le biais de l'infrastructure Netweaver avec un moteur qui sollicite le service via un proxy.

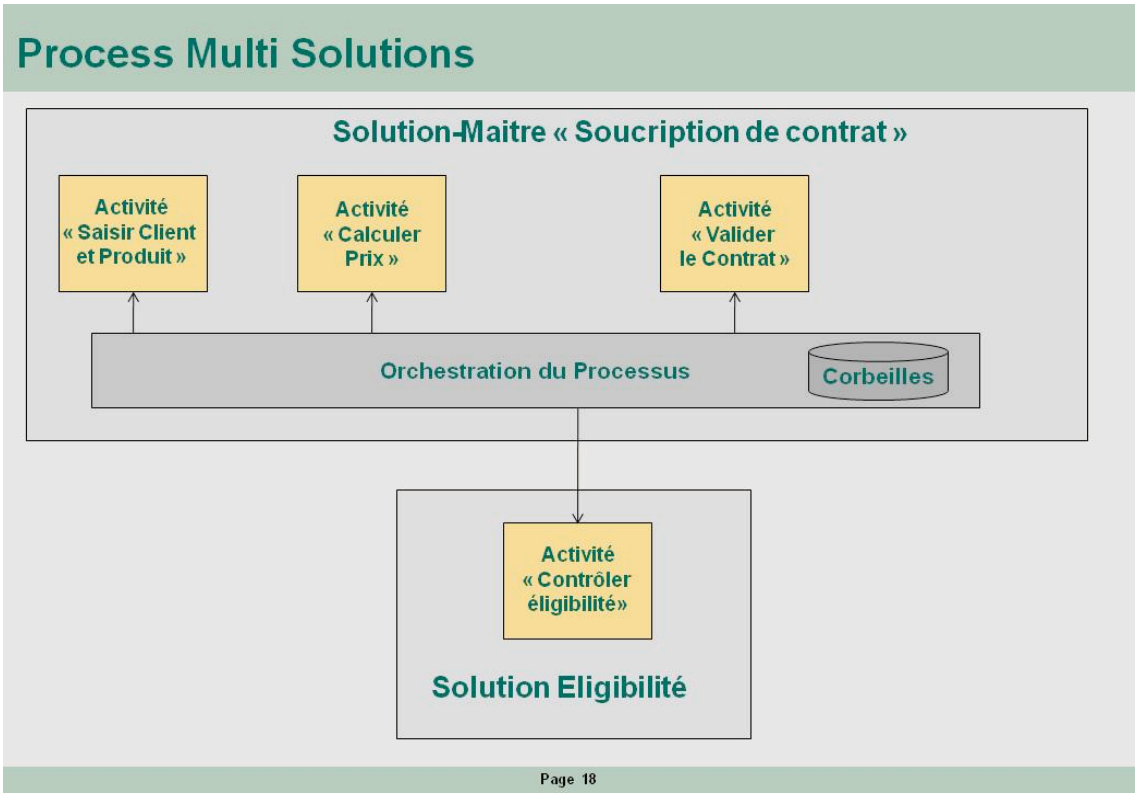
5.9 Moteur de workflow transverse aux Solutions

Afin de favoriser le développement d'une démarche de management par les Processus (au travers de plusieurs silos fonctionnels), les Entreprises cherchent à gérer des Processus de bout en bout qui traversent plusieurs Solutions.
Pour reprendre la terminologie du CEISAR : un **Processus** est décomposé en **Activités** successives ; Une Activité ne peut être exécutée que par un Acteur à un même moment.
Si les choix de Solutions ont été effectués indépendamment les uns des autres, l'enchaînement ne sera pas simple.

Nous n'avons pas trouvé d'exemples opérationnels dans nos études de cas d'un moteur de Processus **externe** à l'ensemble des Solutions qui enchaîne des Activités exécutées par des Solutions différentes.
Par contre, on a trouvé un exemple intéressant : un Processus **interne**, géré par une Solution maître, mais qui coordonne des Activités d'autres Solutions.
Le principe est simple :

- une **corbeille** par Acteur ou groupe d'Acteurs contient les activités à effectuer et leur contexte
- des Services de lecture écriture sont offerts à toutes les Solutions : chacune peut alors poster une activité dans la corbeille ou lister les activités qui s'y trouvent

Dans le schéma ci-dessous, le Processus de souscription de contrat est géré par la Solution de souscription qui est la Solution maître.



Ce Processus est décomposé en Activités successives :

1. « saisir les données du client et le produit souscrit»,
2. « calculer le prix »,
3. « vérifier l'éligibilité du souscripteur »,
4. « valider le contrat »…

Les 2 premières activités sont enchaînées par la Solution de souscription. Si des Acteurs différents doivent intervenir un mécanisme de corbeille permet en fin d'Activité de placer un évènement dans la corbeille de l'Acteur suivant. A chacun d'exécuter les Activités qui lui ont été attribuées ce qui déclenche progressivement des évènements jusqu'à la fin du Processus.
Mais si l'Activité 3 « vérifier l'éligibilité du souscripteur » est gérée par une autre Solution, par exemple la Solution « contrôle médical » qui gère les Activités des médecins, alors il faut que l'écriture dans la

corbeille du médecin, effectuée dans la Solution maitre, déclenche un message sortant vers la Solution « contrôle médical » et qu'à l'issue de l'Activité un message entrant mette à jour la prochaine corbeille.
Pour résumer, on enchaîne bien des **Activités appartenant à différentes Solutions** à partir d'un **chef d'orchestre unique appartenant à la Solution principale**.
A titre d'exemple, l'étude de cas AXA France a montré le souhait de converger vers une corbeille de tâches unique, intégrant des tâches issues de plusieurs Solutions (même si chaque Solution dispose de sa propre gestion de tâches) afin de simplifier l'interface utilisateur.

5.10 En déduire les interfaces d'échange

Une fois connus le Modèle d'Information de l'Entreprise et les besoins d'échanger informations et fonctions, on peut définir les Interfaces.

5.10.1 Définir les interfaces synchrones et asynchrones

Synchrone :

- le Progiciel doit pouvoir **lire** des **informations** gérées par d'autres Solutions (par exemple accéder au fichier client déjà en place), et réciproquement
- le Progiciel doit pouvoir **déclencher des Fonctions** offertes par d'autres Solutions (par exemple « calculer un prix » ou déclencher la fonction « envoi de message »), et réciproquement
- pour **mettre à jour** les **informations** gérées par d'autres Solutions, le Progiciel procède généralement non pas par mise à jour directe des données de l'autre Solution, mais par transfert d'évènements exécutés sur la Solution destinataire.

Asynchrone

- le Progiciel doit pouvoir transmettre des **flots** d'informations à d'autres Solutions (par exemple générer des écritures vers un système comptable), et réciproquement
- les échanges peuvent s'effectuer par **transfert de fichier** périodique ou de façon plus continue, au **fil de l'eau**

Au vu des études de cas, on constate que les **échanges synchrones sont minoritaires** : ils sont limités au strict nécessaire.
Ce n'est pas **par difficulté technique**, mais parce que

- l'appel de Services externes en lieu et place de fonctions offertes au sein de la Solution, signifie une modification importante de la Solution : ce caractère **intrusif** rebute ceux qui ne souhaitent pas modifier des Solutions en place trop fragiles
- par ailleurs, l'utilisation de Services externes suppose que différentes Solutions s'exécutent avec la même fiabilité : cette **interdépendance** fragilise l'ensemble

A titre d'exemple, RCI Banque a choisi de ne développer que des interfaces asynchrones pour limiter les perturbations sur l'ensemble et privilégier des délais rapides dans le projet.

5.10.2 Chaque Interface est définie par le « producteur »

C'est au « producteur » de proposer sa demi-interface, et c'est aux « consommateurs » de s'y conformer.
Par exemple, la Solution CRM gère les informations client : elle offre le Service « Lire Client » et définit donc l'interface de cette Fonction, ce qui entre et ce qui sort. Les Solutions appelantes doivent se conformer à ce format.

Point de vue du CEISAR

Ceci dit, pour définir la demi-interface côté producteur, il faut se mettre à la place du consommateur : définir d'abord le besoin d'un échantillon de consommateurs avant de figer l'interface producteur, puis faire respecter cette Interface par les autres consommateurs.

5.10.3 Limiter le nombre d'interfaces

On peut construire une interface pour chaque échange.
On peut aussi **mutualiser les interfaces** en proposant des formats pivots qui sont réutilisés par plusieurs Solutions, même si chacune d'entre elles n'utilise pas toute les informations contenues dans le format pivot. C'est un moyen de limiter la diversité des interfaces, aux dépens de flux de données plus importants : un compromis doit donc être trouvé entre limitation du nombre d'interfaces pour optimiser les développements et spécialisation des interfaces pour optimiser les flux.

Pour les **Solutions Primaires**, la situation idéale est de se doter d'outils pour limiter la charge de développement et d'évolution ce qui permet de gérer d'avantage d'interfaces optimisées.
Pour les **Solutions de pilotage**, lorsque la liste des rapports croît, le nombre d'interfaces avec le système de reporting s'accroît de la même façon : il vaut mieux alors constituer un repository des informations de base à l'aide d'interfaces standards, puis laisser chacun produire son rapport avec l'outil de reporting

5.10.4 Réutiliser les interfaces existantes ou standards

Certains Editeurs proposent en standard des « demi-interfaces » pour que les Solutions extérieures puissent réutiliser des Services ou des informations du Progiciel.
Dans ces interfaces, les Editeurs essaient de favoriser les standards pour que leurs produits s'intègrent plus aisément avec les autres.
Par « standard », on inclut à la fois la couche **métier** (quelles informations sont échangées, ce qui peut être défini par des organismes interprofessionnels) et la couche **technique** (quel middleware est utilisé pour faire parvenir le message à la Solution destinataire, tels que Web Services pour les échanges synchrones ou MQ pour les échanges asynchrones).
L'utilisation d'interfaces standards entre Entreprises a des avantages évidents :

- pas de charge de spécifications,
- accord aisé entre partenaires,
- diminution de la complexité globale : limite le nombre d'interfaces avec les partenaires
- facilité l'interconnexion de Clouds différents

Une fois ces interfaces mises en place, l'Entreprise peut aussi les utiliser en interne pour faciliter les échanges entre ses propres Solutions. Le secteur aérien est un bon exemple : une association interprofessionnelle (IATA) a normalisé les échanges entre Acteurs. Les Entreprises et les Editeurs (comme Amadeus) ont capitalisé sur ces définitions de données et de messages pour développer des services-Logiciels.

La volonté de couvrir les besoins de toutes les Entreprises conduit parfois à des **interfaces standards qui sont trop lourdes** et non utilisables parce qu'elles représentent le sur-ensemble des besoins des Entreprises.
Voir l'exemple de Renault qui a fini par renoncer aux interfaces standards définies pour la gestion de Sinistres parce qu'elles étaient trop complexes pour leur besoin.

Lorsque les partenaires avec lesquels échanger deviennent trop nombreux, on peut aussi s'appuyer sur des Entreprises spécialisées telles que **Crossgate** (http://en.wikipedia.org/wiki/Crossgate_AG) qui fait de l'EDI « on demand » avec 40.000 clients différents.

5.10.5 La réplication par abonnement

Il semble qu'un mode intermédiaire entre « transfert de fichiers » et « synchrone » qui est la « **réplication par abonnement au fil de l'eau**» soit actuellement de plus en plus apprécié. Il consiste à :

- désigner quelle Solution est maître des données
- répliquer les données dans d'autres Solutions sous forme d'envoi des mises à jour de ces données au fil de l'eau

Ce mécanisme laisse chaque Solution s'exécuter de façon **autonome** puisque les données sont bien locales et d'offrir une bonne **fraîcheur** d'information. Par contre, il nécessite de mettre en place un mécanisme sophistiqué de gestion des mises à jour.
Un bon exemple d'application est la gestion d'un catalogue produit centralisé.
Une étude de cas intéressante est celle d'Oberthur Technology qui a choisi ce mécanisme pour synchroniser en permanence un référentiel produit mis à jour en centralisé, tout en laissant chaque usine autonome dans son fonctionnement quelque soit le pays où elle est implantée.
Cette technique est utilisée aussi pour les échanges entre un site marchand et des catalogues produits de chacun de leurs distributeurs.

5.10.6 Middleware

L'Entreprise doit sélectionner un middleware pour organiser ses échanges.
Si la majorité de ses Solutions provient d'un même Editeur, elle doit choisir le middleware recommandé par l'Editeur, tel que Fusion Middleware chez Oracle ou Netweaver chez SAP.
Sinon, elle doit faire un choix indépendant qui s'appuie sur les standards.

Avis de Conseils-Plus, cabinet de conseil :
Pour s'affranchir de la complexité et pour uniformiser le standard (on ne fait pas forcément communiquer du Dotnet et du Java nativement !), il faut créer une couche intermédiaire entre le WebService fourni par l'éditeur de Progiciel et la Solution qui appelle (et réciproquement). On a besoin de cette couche car les éditeurs s'acharneront toujours à créer des obstacles de communication.

5.10.7 Atelier Produit et génération d'interfaces

La variété des interfaces peut provenir de la diversité des Produits.
Dans l'assurance, chaque Produit est assemblé à partir de garanties et de prestations différentes.
Les objets « Contrats » et « Sinistres » qui s'en déduisent ont donc des formats très variés qui nécessitent autant d'interfaces. Pour limiter cette variété, on définit des interfaces larges qui représentent le sur-ensemble des formats nécessaires ; chacun n'utilise que les données qui lui sont utiles, mais les formats peuvent devenir volumineux et freiner les bonnes performances de la Solution.
Pour éviter ces problèmes, il faut prolonger les Solutions « Atelier-Produits » qui servent à composer de nouveaux produits, pour qu'elles génèrent automatiquement les interfaces spécialisées à chaque type de produit : on obtient alors le beurre et l'argent du beurre, on optimise les interfaces sans avoir à les développer.

6 Processus de choix de Progiciel

RÉSUMÉ

Le Processus de choix du Progiciel implique de :

- Formaliser les **objectifs** du projet, dans un document court et validé par le sponsor du projet, identifiant des indicateurs de mesure de l'atteinte des bénéfices

- Identifier les fonctionnalités attendues, ce qui est une pratique courante, mais également

- Identifier trois caractéristiques majeures du Progiciel
- la capacité de personnalisation
- la capacité d'évolution
- la capacité à s'interfacer

- Procéder par prototypage

6.1 Traditionnellement, le choix se fait uniquement sur les fonctionnalités, le prix, ou la taille de l'éditeur

La sélection des Progiciels se fait généralement domaine par domaine, fondée principalement sur des **exigences fonctionnelles.** La **qualité de l'Architecture globale** passe souvent au second plan. Dans certaines Entreprises, on fait même un choix sans comparer les fonctionnalités offertes par le Progiciel aux besoins de l'Entreprise : pour les Progiciels de Commodité on choisit simplement un grand nom d'Editeur en faisant le pari que cet Editeur a eu du succès parce qu'il savait satisfaire les besoins d'autres clients, et que l'on pouvait donc faire l'économie d'une démarche de choix plus approfondie.

Ce mode de sélection aboutit à un paysage global du SI composé de Solutions

- plus ou moins bien **intégrées** entre elles
- plus ou moins **redondantes**
- laissant des « **trous** » fonctionnels à combler par des développements spécifiques ou des Progiciels spécialisés.

6.2 Les critères de choix évoluent

Pour les Progiciels à couplage fort, analyser les fonctions offertes ne suffit pas. Il faut ajouter **trois critères** :

- capacité à **personnaliser** le Progiciel aux spécificités de l'Entreprise
- capacité du Progiciel à **évoluer**
- capacité du Progiciel à **s'interfacer** dans l'Architecture d'Entreprise

Le cas de Renault (RCI Banque) est un bon exemple : l'Entreprise a insisté sur

- les capacités de personnalisation du Progiciel car cela permet un déploiement rapide de la Solution (sans avoir à faire de spécifique)
- les capacités d'évolution en recherchant un Progiciel riche couvrant des domaines de fonctionnalités non encore utiles à l'Entreprise mais pouvant apparaître nécessaires dans le futur.

Mais autant on peut juger des fonctionnalités disponibles en accédant à la documentation ou en participant à des démonstrations, autant il est difficile de se faire son opinion sur ces trois critères par la même méthode : il faut bâtir des **prototypes** pour mettre réellement à l'épreuve le Progiciel. La

démarche est plus coûteuse, mais le jeu en vaut la chandelle : un Progiciel est en général installé pour au moins une dizaine d'années.

Point de vue du CEISAR

Pour alléger la tâche des Entreprises, il serait fort utile à tous que les analystes qui comparent les différents Progiciels puissent prendre en compte ces trois différents critères (personnalisation, interfaçage, évolutivité): ne faudrait-il pas qu'ils organisent des Prototypes comparatifs permettant de juger les capacités de chaque offre ? Compte tenu de leur influence sur le marché, une majorité d'Editeurs seraient prêts à jouer le jeu, pourvu que les spécifications du prototype soient équitables.

6.3 L'avis des architectes doit être pris en compte

Une **gouvernance** doit être mise en place qui garantit que le choix des Progiciels n'est pas fait que sur la base des fonctionnalités disponibles, le coût, et la pérennité ou taille du vendeur.
Les **Architectes** doivent être associés **en amont** des décisions de Progiciels pour aider à évaluer les trois critères complémentaires cités ci-dessus.
C'est aux Architectes d'exprimer et de faire respecter les contraintes de l'Entreprises en matière de :

- conformité du Modèle Métier proposé
- Architecture SOA
- volumétrie et progressivité
- performances
- fiabilité
- capacité d'interfaçage
- …

Ces contraintes doivent être exprimées sous forme de besoins et non de solutions technologiques.
Par ailleurs, l'Architecte peut vérifier que ses choix préalables d'outils techniques pour les Opérations (OS, SGBD, Middleware) peuvent être respectés en se limitant à l'essentiel: ne pas chercher à prédéfinir des choix technologiques trop précis et nombreux qui restreindraient la liste des Progiciels éligibles.

Point de vue du CEISAR

Pour que l'avis des Architectes soit écouté, ils doivent illustrer les conséquences des différents scenarios en fonction de critères compréhensibles par les métiers qui sont sensibles au time to market, aux coûts de personnalisation, aux coûts d'évolution, à la vision client unique, à la facilité d'utilisation…beaucoup plus qu'aux guerres de religion sur l'Open Source, le respect de tel ou tel standard technologique ou l'efficacité de tel ou tel outil technique.

6.4 L'avis des intégrateurs

L'intérêt de l'intégrateur est de réutiliser les compétences qu'il a déjà formées : il aura donc une certaine tendance à recommander des produits qui ont déjà une grande part de marché aux dépens des nouvelles Solutions plus modernes auxquelles il n'a pas encore formé ses équipes.
Ceci dit, leur intérêt est aussi de se positionner sur les Progiciels émergents qui risquent d'avoir du succès pour monter une pratique nouvelle sur laquelle peu de concurrents existent.
A chaque Entreprise d'identifier si l'intégrateur est conservateur ou novateur…

6.5 La démarche de choix d'un Progiciel

6.5.1 Définir le but du Projet

Il est étonnant de vérifier qu'il existe rarement un document court et clair qui identifie le but d'un Projet d'installation de Progiciel.

On ne peut que recommander de formaliser ce document pour que chaque acteur du projet ait en tête ce but chaque fois qu'il est amené à prendre une décision.
Définir les **objectifs** du projet et les **indicateurs** associés qui permettent de vérifier sa réussite.
Définir le **périmètre :** domaine fonctionnel, territoire, types d'Acteurs.
Préciser les **contraintes sur le projet**, et en particulier les contraintes de coût et délai.
La définition du but n'est pas une spécification détaillée de la Solution recherchée : c'est un document court, centré sur les bénéfices pour le Métier, exprimés sous forme d'indicateurs. Ne pas confondre « Définir les objectifs » et « Spécifier la Solution ».

6.5.2 Prototyper

Sélectionner des candidats.
Analyser et Prototyper : les **démonstrations** sont suffisantes pour juger des fonctionnalités disponibles. Par contre elles sont insuffisantes pour juger des trois autres critères. Un **prototype** est nécessaire.

Point de vue du CEISAR

Le choix d'un Progiciel se fait pour dix ans.
Pour évaluer les trois capacités fondamentales du Progiciel (personnalisation, interfaçage et évolutivité) il est indispensable de réaliser un « **proof of concept** » : ne pas se contenter des déclarations des Editeurs, expérimenter soi-même ces trois capacités quitte à ce que l'Entreprise finance ces prototypes pour le dernier carré d'Editeurs présélectionnés.
Le choix du périmètre du prototype doit été suffisamment large pour être probant, mais suffisamment restreint pour limiter l'effort. Compter une semaine pour les cas simples à trois mois pour les cas complexes.

6.5.3 Comprendre la capacité du Progiciel à être personnalisé

Comprendre les mécanismes de personnalisation : développements spécifiques et configuration

- pour la configuration
  - moteur de règles
  - moteur de workflow
  - paramétrage
  - attributs dynamiques
- pour les développements spécifiques :
  - quels points d'entrée ?
  - comment gérer la compatibilité ascendante ?

6.5.4 Comprendre la capacité du Progiciel à être installé par module

Les Progiciels couvrent un périmètre fonctionnel croissant (voir ci-dessus).
Pour obtenir des résultats rapides et limiter les risques, il faut donc installer le Progiciel par Modules successifs.
L'ordre logique dépend des priorités des besoins clients, il est aussi guidé par l'optimisation des coûts globaux d'intégration : il faut que le scénario choisi minimise les besoins d'interfaces provisoires que l'on abandonne progressivement.
Les interfaces doivent être développées version par version, mais la majorité d'entre elles se retrouvent dans la première version. C'est une des raisons pour lesquelles la première version est plus difficile à installer que les suivantes.

6.5.5 Comprendre la capacité du Progiciel à évoluer

Il existe deux méthodes pour juger de la capacité d'un Progiciel à évoluer.
On peut juger de **l'Architecture d'un Progiciel** pour mieux évaluer sa capacité à évoluer, ce qui nécessite que l'Editeur communique sur ce thème : Modèle d'Informations, structure du logiciel, outils de développement…
On peut juger le résultat d'un **prototype** : quel délai et quelles ressources pour l'ajout de fonctions qui n'existent pas dans le Progiciel?

6.5.6 Comprendre la capacité du Progiciel à s'interfacer

Il faut là encore appliquer les deux méthodes :
Comprendre le Progiciel et ses outils

- Comparer les **modèles d'information**
  - Définir le Modèle d'Information partagé
  - Comprendre les écarts avec le Modèle d'Information du Progiciel
- Comprendre les Interfaces proposées par le Progiciel
- identifier les outils et méthodes fournis par l'Editeur

Prototyper quelques Interfaces pour démontrer la faisabilité.

7 Processus d'installation du Progiciel

Sur quoi faut-il améliorer le Processus d'installation pour faciliter l'intégration avec les autres Solutions ?

RÉSUMÉ

- Exprimer des **besoins** et non des solutions : Utiliser le progiciel au maximum
- Faire de la **configuration** et non du spécifique, en particulier pour les interfaces. Il faut que le Progiciel en ait la capacité.
- Utiliser une approche **agile** (en particulier pour les Progiciels Verticaux)
- Rechercher la **progressivité** dans l'installation : cela permet des résultats rapides. Le Progiciel doit être assez **modulaire** pour permettre cette approche.
- Rechercher une relation **partenariale** entre le Client et l'Editeur. Le « Client est roi » mais doit aussi être conscient des exigences de l'Editeur.

7.1 Gap Analysis : étudier les écarts entre besoins et fonctionnalités fournies par le Progiciel-Produit

Pour utiliser au mieux les capacités du Progiciel, il faut exprimer des **besoins** et non des **solutions.**
Il existe différentes solutions pour résoudre le même besoin ; le Progiciel en a choisi une.
Imposer une solution différente pour ressembler à ce que l'on pratique déjà dans la Solution à remplacer, signifie :

- des développements spécifiques
- une maintenance ultérieure spécifique
- une augmentation des coûts de montée de version
- une difficulté à réutiliser les interfaces proposées par le Progiciel.

Pour ne pas tomber dans ce travers, il est fondamental que ceux qui vont exprimer des besoins s'imprègnent au préalable des fonctionnalités offertes par le Progiciel.
Pour résumer, il ne faut pas exprimer isolément des besoins, mais bien assimiler les fonctionnalités du logiciel **avant** d'exprimer les besoins.

Point de vue du CEISAR

Pour limiter les personnalisations inutiles, certaines Entreprises ont compris qu'elles devaient coller aux Fonctions proposées par le Progiciel. Cela change la façon dont les MOA rédigent les spécifications : ils doivent s'imprégner des fonctionnalités déjà disponibles dans le Progiciel avant d'écrire la moindre ligne de spécification. Il faut bien comprendre qu'il y a une différence entre « exprimer un besoin » et « dicter la réponse au besoin ». Les Acteurs métier franchissent trop souvent cette limite, en spécifiant à outrance des éléments qui ne répondent pas forcément mieux à leur besoin fondamental que le Progiciel. Un besoin est par exemple « Saisir une demande client au call center en moins de 30 secondes » alors que « Réduire le nombre d'écrans d'un facteur 10 » est déjà une Solution. Se contenter de n'exprimer que les besoins fondamentaux a de nombreux avantages : le cahier des charges est plus simple à écrire tout en représentant mieux le vrai problème, le choix du Progiciel est facilité, la mise en œuvre peut tirer pleinement parti du potentiel du Progiciel, les montées de version sont facilitées. Mettre en place un Progiciel n'est pas la même chose que développer une Solution spécifique. La première étape essentielle est de se former au Progiciel pour s'imprégner de ses fonctionnalités avant d'exprimer ses besoins que ce soit pendant le choix du Progiciel ou par la suite. Un certain nombre d'Entreprises ont fixé des taux maximum de personnalisation pour le développement spécifique (moins de 5% de lignes de code pour la plupart). Une fois gommées les personnalisations inutiles, souvent demandées pour reproduire le fonctionnement de l'ancienne Solution qui est remplacée par le Progiciel, il est nécessaire d'adapter le Progiciel à l'Entreprise.

7.2 Personnaliser le Progiciel

7.2.1 Les besoins de personnalisation

Les Entreprises souhaitent personnaliser le Progiciel, tout particulièrement les Progiciels Verticaux. Certaines personnalisations sont légitimes : adaptation à l'organisation de l'Entreprise, prise en compte de ses produits.

D'autres sont moins légitimes. Un cas fréquemment rencontré est celui de **l'interface utilisateur**.
Pour offrir des Processus différents à chaque Entreprise, l'Editeur a modularisé l'Interface utilisateur : les différents modules de présentation sont enchaînés dans un ordre qui peut dépendre de l'organisation de chaque client. Cette modularité peut être mal ressentie par le client : il ne retrouve pas la continuité qui caractérisait son ancienne Solution, il trouve qu'il faut enchaîner trop d'écrans pour aboutir au même résultat, il demande donc à personnaliser l'interface utilisateur.
Il faut être très prudent dans ces demandes de personnalisation : l'Entreprise va devoir maintenir une partie spécifique importante.

Point de vue du CEISAR

Si le Progiciel est manifestement trop lourd à l'utilisation, le même besoin doit être ressenti par les autres clients. Il faut donc demander une évolution du Progiciel plutôt qu'un développement spécifique.
Attention à ne pas reproduire ce à quoi les utilisateurs sont habitués.
Attention à ne pas perdre la modularité de l'interface utilisateur qui va donner de la souplesse d'organisation.
Ne pas hésiter à installer le Progiciel en l'état sur un site pilote pour vérifier si les demandes subsistent après une phase d'apprentissage.
Si la revendication persiste, il faut alors la prendre en compte, soit par une évolution du Produit soit par un développement spécifique qu'il faudra maintenir.

On peut personnaliser un Progiciel par configuration (paramétrage, moteur de règle, moteur de workflow) ou par développement spécifique complémentaire.
Il est cependant formellement déconseillé de modifier le cœur du logiciel.

7.2.2 Personnaliser par configuration

La personnalisation par **configuration** permet de localiser les modifications par paramétrage et moteur de règles, sans déstabiliser le logiciel.
Elle est plus rapide, plus sûre et peut être mise entre les mains de professionnels du métier qui n'ont pas de compétences informatiques. Mais elle est limitée au périmètre prévu par le concepteur du Progiciel.
C'est pourquoi les éditeurs de logiciels cherchent à maximiser l'utilisation de la configuration. Lorsqu'ils parviennent à un périmètre large, le risque est de perdre les « configurateurs » dans le foisonnement de données disponibles : il faut alors **restreindre le champ de données** utilisables pour chaque fonction configurable pour faciliter le travail de chacun.
Une Configuration est rangée sous forme de données. Elle est versionnée comme un logiciel. Elle doit être transportable d'une version du Progiciel à l'autre. Elle est documentée : il faut se rappeler pourquoi certains choix de paramétrage ont été faits.
C'est l'ensemble de ces mécanismes qui va permettre d'assurer la compatibilité ascendante : la montée de version est simple lorsque la personnalisation est faite par Configuration.

7.2.3 Personnalisation par développement spécifique

La personnalisation par **développement spécifique** est plus longue et coûteuse, mais a pour avantage de tout permettre.
La difficulté essentielle est de continuer à bénéficier des nouvelles versions du Progiciel-produit.
Pour limiter la charge de la montée de version l'éditeur de logiciel doit :

- offrir des points d'entrée où peuvent s'effectuer les modifications
- stabiliser ses formats de données pour ne pas contraindre son client à migrer les informations vers le nouveau format, ou, fournir un outil de migration qui aide la migration

Point de vue du CEISAR

Faites tout ce que vous pouvez pour restreindre les développements spécifiques :
- favorisez la nomination d'un responsable fonctionnel qui a le poids suffisant pour freiner les demandes de personnalisation
- refusez les demandes de personnalisation lorsque le Progiciel a déjà résolu le même problème sous une autre forme
- lors du choix de Progiciel prototypez les possibilités de personnalisation par configuration pour en mesurer l'efficacité
- si la personnalisation est indispensable, utilisez la Configuration au maximum

7.2.4 Comment personnaliser pour les différentes filiales d'un groupe ?

Lorsqu'un Progiciel est construit pour **plusieurs pays**, il faut permettre de personnaliser :
- la langue
- les interfaces d'échanges standardisées avec des Solutions externes telles que:
  - Solutions de paiement
  - Solutions d'échanges interprofessionnelles
- il faut parfois ajouter des mécanismes qui n'existaient pas dans le Progiciel mais qui sont nécessaires pour satisfaire une règlementation spécifique
- si le Progiciel a une capacité de configuration puissante, une grande partie de la personnalisation spécifique à un pays peut s'effectuer directement par configuration, sans nécessité de développement complémentaire ; par exemple pour la fiscalité. Cette personnalisation nécessite d'utiliser des **attributs dynamiques** lorsque les critères sont variés d'un pays à l'autre. dans ce dernier cas, la principale difficulté devient alors la fabrication d'interfaces qu'il faut automatiquement générer.

7.3 Construire les interfaces d'échange

7.3.1 Comment faire ?

- Si l'on souhaite échanger des informations en lecture ou écriture
  - préférer les **Services d'accès direct** aux informations :
    - si nécessité d'une **fraîcheur** d'information immédiate (ex : solde bancaire)
    - si on peut garantir un **bon fonctionnement simultané** de toutes les Solutions qui coopèrent
  - préférer les Services de **transfert de fichiers**:
    - si la fraîcheur d'une information décalée peut être acceptable (ex : description de organisation de l'Entreprise)
  - préférer la **réplication par abonnement**:
    - si exigence de fraîcheur d'information
    - si volonté d'indépendance d'exploitation des différentes Solutions (ex : catalogue produit à jour pour toutes les usines)
- Si l'on souhaite exécuter des Services :
  - préférer une exécution immédiate de ces Services si une réponse est nécessaire pour poursuivre (exemple calculer le prix, autoriser une transaction)

Une **approche itérative** est-elle possible ?
Une des Entreprises nous a indiqué qu'elle avait appliqué une méthode agile pour construire ses interfaces avec un nouveau Progiciel complexe nécessitant cent vingt interfaces.

Elle a procédé en deux itérations ce qui l'a contrainte à refaire une partie du travail.
Mais elle considère que globalement, elle est allée plus vite que par la méthode classique.

Connaître la source et la cible
Un facteur souvent cité est la difficulté à connaître la **source** et la **cible** pour concevoir l'interface si l'**Architecture interne** des Progiciels n'est pas communiquée et si la Solution couplée est mal documentée

Versionner les Interfaces : comme tous les éléments d'un Modèle, les Interfaces vont évoluer. Il est souhaitable de prévoir dès le départ qu'elles soient versionnées.

Point de vue du CEISAR :

Certaines Entreprises ne veulent pas modifier leur Solution existante pour qu'elle s'adapte à une nouvelle interface. Elles demandent alors au **Progiciel de s'adapter aux Solutions existantes**.
C'est une option très déstabilisante pour le Progiciel que l'on ne recommande pas : pour ne pas polluer des Solutions qui vont disparaître, on pollue les Solutions futures !

7.3.2 Interfaces provisoires

Dans une de nos études de cas on a installé des Interfaces provisoires avant de développer des interfaces définitives.
Pour installer une première version du Progiciel très rapidement, on a été amenés à fabriquer des **interfaces provisoires** qui n'ont pas le niveau de qualité, de performance et de généricité que doivent avoir des interfaces bien construites.
Dans un deuxième temps on réécrit les **Interfaces définitives** qui remplacent les interfaces provisoires.
C'est une façon de lancer rapidement une version 1 du Progiciel, mais c'est globalement plus coûteux et risqué : il faut être certain que l'on aura bien les moyens de réécrire des interfaces définitives que l'on doit reprendre par la suite.

7.3.3 Métrique

On a obtenu des chiffres extrêmement divers selon que des outils sont ou non disponibles.
S'il n'y a pas d'outils de construction d'interfaces les couts sont de 10 à 30jh selon la complexité de l'interface.
Si des outils sont à disposition, les coûts sont de 2 à 10jh par interface.

A travers nos études de cas nous avons noté que pour les Solutions à fort couplage (hors charge de migration), **80% de la charge est liée aux interfaces et 20% à la personnalisation.**

Il serait intéressant qu'une étude plus approfondie confirme ces premières indications : on n'en a pas eu le temps lors de la fabrication de ce livre blanc.

7.4 Migrer les informations

En général, un nouveau Progiciel remplace une ou plusieurs Solutions existantes.
Les Informations gérées par l'ancienne Solution doivent être migrées vers le Progiciel.
Comme les Modèles d'informations sont différents et que les degrés d'exigence sur la qualité des informations croissent avec le nouveau Progiciel, cette opération est particulièrement délicate et représente souvent un goulet d'étranglement pour les projets.
La migration des Informations va réutiliser les Interfaces d'accès aux Informations pour contrôler et charger les informations dans le Progiciel.

7.5 Progiciel Matrice pour les grands groupes

Le même Progiciel peut être configuré de façon très différente.
Pour harmoniser les Modèles des filiales, les Groupes cherchent à construire une matrice commune réutilisée par chaque filiale. Ceci permet de limiter le nombre d'interfaces à entretenir.

Nous parlerons de Progiciel Matrice (aussi appelé Core Model ou Template). C'est un premier niveau de personnalisation d'un Progiciel-Produit par une équipe Groupe. Le Progiciel-Matrice pourra ensuite être personnalisé pour chaque filiale.
En parallèle de cette recherche d'harmonisation des Processus Métier, on souhaite parfois également partager l'exploitation des différentes instances des filiales, afin de réaliser des économies d'échelle. Plusieurs grands clients de SAP (dont Total) ont d'abord réalisé des implémentations autonomes par filiale du Progiciel, puis on mené des projets de convergence afin de réduire le nombre d'instances et de personnalisations différentes du produit.

7.6 Outils pour faciliter l'intégration d'un Progiciel

Pour résumer chaque éditeur pourrait aider ses clients en fournissant des outils complémentaires à son Progiciel qui devraient être pris en compte dans les choix de Progiciels : par exemple

- outil de fabrication d'interfaces : conversion, mapping, adaptation au middleware choisi
- outil de migration d'information
- outil de gestion de Processus
- référentiel de Services
- outil de gestion de configuration qui gère le cycle de vie du Progiciel et de sa configuration, et les versions successives

7.7 Ne pas négliger le point de vue de l'Editeur

Pour des **Progiciels de Commodité** tels que la paye ou la comptabilité, la relation est assez simple à gérer :

- l'Editeur propose un Progiciel
- le client qui l'a choisi l'utilise tel quel : les demandes d'évolution sont souvent les mêmes que celles des autres clients ; il faut suivre la réglementation et enrichir progressivement des fonctions utilisables par tous

Pour des **Progiciels Verticaux**, le client cherche à se différencier, la relation est plus complexe :

- l'Editeur propose un Progiciel qui est souvent considéré comme **incomplet** par rapport aux besoins métier extensibles du client
- le client cherche à y introduire ses spécificités, ses produits, ses Processus : le niveau de **personnalisation** est plus important
- les **montées de version** sont plus difficiles compte tenu de la masse de personnalisation introduite par chaque client

Mais les Editeurs ont aussi des exigences vis-à-vis de leurs Entreprises clientes. « Le client a toujours raison » : parler des exigences des Editeurs vis-à-vis de leurs clients peut paraitre déplacé. Mais à travers les différentes études de cas, nous avons identifié les exigences suivantes :

- utiliser **l'approche de l'éditeur** et non l'approche client lorsqu'on installe un Progiciel. En particulier pour les Progiciels **Verticaux**, utiliser la méthode **agile** si elle est préconisée par l'Editeur. Il est extrêmement difficile pour l'Entreprise de définir tous ses besoins Verticaux. Elle doit se laisser l'opportunité de les affiner progressivement. En outre, une fois le choix officialisé, l'Entreprise a besoin de résultats rapides pour rassurer décideurs et utilisateurs.
- il est indispensable que les Acteurs de l'Entreprise bénéficient d'une **formation approfondie** à l'utilisation du Progiciel **avant** d'exprimer leurs besoins : les demandes spécifiques seront alors en forte diminution
- l'Entreprise doit désigner un **homme fort** capable de limiter les demandes de spécifique des métiers : « no way » doit être la réponse courante
- choisir un **intégrateur** non pas pour sa taille mais parce qu'il **connait** le Métier Vertical ou le Progiciel ou les deux
- ceux qui connaissent les Solutions avec lesquelles s'interfacer doivent être **disponibles**

8 Processus d'évolution du Progiciel

RÉSUMÉ

- Les Editeurs définissent une politique éditoriale qui inclut une « roadmap » Produit et les règles de maintenance des versions passées. Les avis des clients influencent le choix des priorités d'évolution.
- Les montées de version sont plus simples lorsque
• le Modèle d'Information est mature et n'évolue pas ou peu
• la montée de version est outillée (analyse d'impact, migration de données)
• les développements spécifiques sont limités au maximum
- les Solutions SaaS pratiquent en général des montées de versions fréquentes et transparentes pour les Clients : tous les clients sont dans la même version, ce qui diminue fortement la problématique de montée de version.

8.1 Evolution du Progiciel

L'Editeur qui construit un nouveau Progiciel commence généralement par un développement spécifique pour un client donné. Puis il identifie que des besoins similaires existent pour d'autres clients et il déduit de sa première Solution une première version de Progiciel qu'il fait rapidement murir au contact de ses premiers clients.
S'il s'agit d'un Progiciel Vertical, son évolution n'est pas simple à décider parce que les demandes des différents clients ne convergent pas facilement : chacun souhaite se différencier et les demandes sont beaucoup plus nombreuses que la capacité de l'Editeur à les satisfaire.
Il faut donc un Processus neutre pour définir les priorités. Pour gérer ce Processus, les éditeurs constituent un comité (souvent appelé « Advisory Board » ou « club utilisateurs ») auquel participent les clients principaux.
C'est à travers ce comité que les clients expriment leurs demandes de nouvelles fonctionnalités.
L'Editeur a lui aussi ses propres priorités liées à sa stratégie et détermine le bon compromis.

Chez certains Editeurs, on peut aussi accepter des évolutions produits au-delà de ce qui est défini dans le plan produit, ce qui se passe de la façon suivante :

- le plan produit est défini et figé
- un client dont la demande n'a pu être prise en compte dans les prochaines versions, doit donc développer en **spécifique** le complément de logiciel correspondant à sa demande
- si le client demande néanmoins que sa fonctionnalité soit prise en compte rapidement dans le produit et que sa demande intéresse aussi d'autres clients, il propose d'en **financer la moitié** du développement
- l'équipe Produit analyse si la demande est bien générique, et vérifie si ce complément de budget lui permet d'ajouter la fonctionnalité dans la prochaine version
- au final, si l'équipe Produit accepte la demande, elle devra maintenir la fonctionnalité au sein de son produit

L'avantage pour le client est que ce financement sera généralement inférieur au coût du développement spécifique, et surtout que la maintenance de la fonctionnalité sera à la charge de l'Editeur.
Pour l'Editeur, c'est un moyen d'accroître son offre grâce à un financement complémentaire de ses moyens de R/D.

8.2 Montée de version

Un Progiciel évolue au fur et à mesure des demandes clients ou des initiatives prises par l'Editeur.
Si les modifications apportées sont des enrichissement fonctionnels qui n'impliquent pas de modification du Modèle d'information ou des Interfaces déjà installées avec les autres Solutions, la montée de version est plus simple : il faut tester la nouvelle version de Progiciel, former les utilisateurs du Progiciel, et l'installer en production.
Mais si, pour apporter ces évolutions, l'éditeur a été amené à modifier son Modèle d'Information et/ou

ses interfaces avec les autres Solutions, alors la montée de version est beaucoup plus difficile. L'Entreprise doit alors migrer ses Informations dans le nouveau Modèle ce qui suppose des outils que peut fournir l'Editeur. En outre l'Entreprise doit tester la nouvelle version du Progiciel dans son contexte : avec les personnalisations qui ont déjà été appliquées aux versions précédentes, par configuration ou développement spécifique, avec les interfaces qui lient le Progiciel aux autres Solutions. Les développements spécifiques constituent la plus grande difficulté en cas de montée de version.
Cette opération de montée de version peut être particulièrement lourde, ce qui dissuade les Entreprises de changer de version trop souvent. C'est pour éviter cet effort que les Entreprises conservent plusieurs années la même version du Progiciel, quitte à ne pas pouvoir bénéficier des nouvelles fonctionnalités livrées dans chaque version. Cette attitude contraint les Editeurs à maintenir plusieurs versions successives du même Progiciel, ce qui a des impacts sur les coûts internes.

Pour stabiliser son Modèle d'information, l'Editeur doit avoir fait mûrir suffisamment son Modèle d'information : généralement les montées de version sont plus difficiles au début du cycle de vie du Progiciel au moment où l'Editeur fait mûrir son Produit. Le Modèle d'Information se stabilise progressivement, non seulement parce que le Progiciel est abouti, mais aussi parce que la masse de clients installés constitue un frein au changement de Modèle d'information. Il y a là un cercle vertueux qui garantit un remplacement progressif des Progiciels qui doivent passer par trois étapes :

- une étape de **construction** : nombreuses nouvelles fonctions dans chaque version, les versions successives sont peu compatibles et donc les montées de version sont difficiles
- une étape de **maturité** : les nouvelles fonctions s'appuient sur un Modèle d'information stabilisé, les montées de versions sont bien maîtrisées ; l'Entreprise peut sous-traiter la **maintenance des parties personnalisées** à des centres de services spécialisés qui sont indépendants du centre de maintenance de l'éditeur qui ne **maintient que le Progiciel**.
- une étape de **vieillissement** : de nouveaux entrants apparaissent qui proposent des Progiciels plus modernes, construits sur de nouvelles technologies, avec de nouvelles fonctionnalités innovantes que les Progiciels en place ne peuvent plus offrir

Quelle devrait être la politique d'un Editeur de Progiciel pour allonger sa durée de vie chez ses clients ? On peut déduire des principes décrits ci-dessus les caractéristiques suivantes :

- l'Editeur doit **prévenir ses premiers clients** que le Progiciel est encore en construction, et que le Modèle d'Information va évoluer, ce qui rendra les premières montées de version plus difficiles
- le Modèle d'Information doit s'appuyer sur des mécanismes qui facilitent une extension du Modèle (approche objet, outil de mapping, attributs dynamiques)
- l'Editeur doit fournir un **outil d'analyse d'impact** qui identifie les Solutions utilisant les Interfaces modifiées (un exemple nous en a été donné dans l'étude de cas Axa)
- le Progiciel doit pouvoir être personnalisable par **configuration** et non par développement spécifique
- les **interfaces** doivent être générées automatiquement par des **outils** fournis par l'Editeur
- l'éditeur doit proposer un **outil de migration** des informations d'une version à l'autre
- les **personnalisations** doivent être gérées par un **outil** qui recense toutes les adaptations déjà effectuées

Politique d'Oracle
La règle est: « 5-3-illimité ».
*Une version N est maintenue **5 ans** avec mise à niveau des paliers techniques.*
*Elle peut être maintenue **3 ans** de plus avec le même niveau de support si le client accepte de payer ; sinon le service est un peu réduit.*
*Au **delà**, si l'incident a déjà été recensé et corrigé, Oracle donne le patch ; s'il n'existe pas, Oracle ne gère pas l'incident.*
Le cycle des versions d'un produit est de deux ans. Ainsi sur huit ans on aura au moins N+3 ou N+4.
Il n'y a donc pas réellement de risque vis-à-vis de l'éditeur sur la durée de maintenance d'une version.
Le client doit être très attentif au degré d'obsolescence qu'il admet, car lorsqu'il y a trop de versions de retard, le gap devient très difficile à combler.

Politique de SAP
*Pour simplifier les montées de version, SAP a été amené à couper son **offre** en quatre morceaux indépendants appelés « enhancement packages » (technique, UI, métier, Processus) qui peuvent évoluer indépendamment les uns des autres.*

Cloud

Lorsqu'un Progiciel est proposé sur le Cloud, la problématique est différente. C'est à l'Editeur d'assurer la compatibilité ascendante entre ses différentes versions. Une seule version fonctionne à un instant donné : l'Editeur doit s'assurer que la nouvelle version fonctionne avec les données de chaque client. Si les formats de données ont changé, c'est à l'éditeur de développer un outil de migration des données et d'effectuer cette migration. Pour éviter de faire cet effort pour tous leurs clients, les fournisseurs de « SaaS » (Software as a Service) doivent rapidement stabiliser leur Modèle d'Information.

Le client n'a rien à faire pour bénéficier des nouvelles fonctionnalités : la montée de version est entièrement à la charge de l'Editeur. C'est un excellent argument en faveur du « Cloud ».

9 Conséquences sur l'organisation des Entreprises

RÉSUMÉ

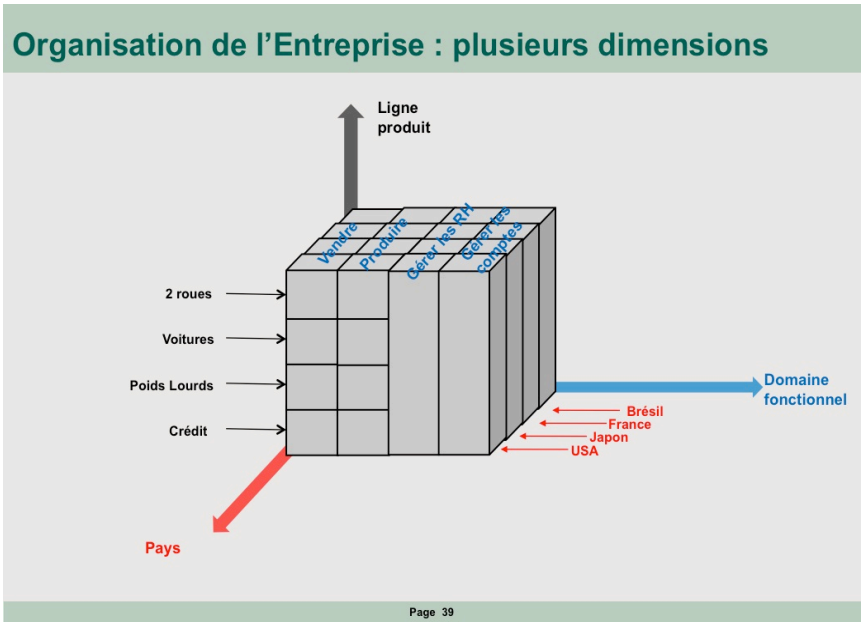
- Pour éviter d'avoir à gérer un ensemble de Solutions disparates, l'Entreprise doit se doter d'une organisation et d'une gouvernance pour développer et faire respecter « **le bien commun** »
- Cela passe par l'existence d'une **Unité « Architecture d'Entreprise »,** qui doit être impliquée dans le choix des Progiciels.
- Le **rôle du Métier** est de plus en plus important, au fur et à mesure du développement des capacités de personnalisation des Progiciels
- Les **compétences** et **formations** associées pour les différents rôles touchent à la fois au savoir faire et au savoir être

9.1 Quelle organisation et rôles sont souhaitables ?

La majorité des Entreprises ne veulent plus d'une Architecture d'Entreprise composée de Solutions choisies indépendamment les unes des autres.
Elles ont compris la difficulté à gérer un ensemble hétéroclite : si les Solutions sont hétérogènes, l'Entreprise est composée d'Unités indépendantes qui coopèrent difficilement et qui ne savent pas échanger. Des Solutions hétérogènes conduisent à des baronnies indépendantes : la transparence est difficile à assurer parce que seuls ceux qui adaptent et utilisent la Solution qu'ils ont choisie en connaissent toutes les forces et faiblesses.
Evoluer vers des Solutions plus homogènes et mieux intégrées signifie moins d'autonomie et plus de contraintes d'évolution. Un responsable de domaine fonctionnel ne verra pas d'un bon œil son indépendance lui échapper.
Il faut donc bâtir une organisation et une gouvernance chargées de faire respecter le « bien commun ».

Une Entreprise peut être structurée en Unités Opérationnelles selon trois dimensions principales :

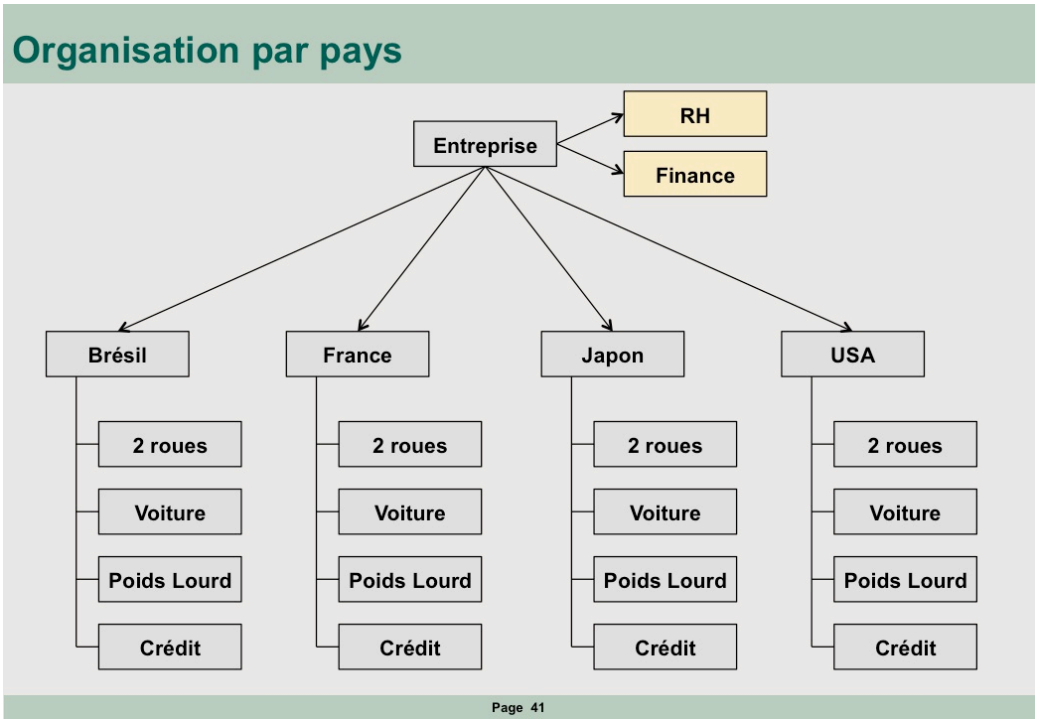
- par **Ligne de Produits** : dans notre exemple on a indiqué quatre lignes de produits qui sont les voitures, les camions, les deux roues et le crédit qui est un produit complémentaire associé à la vente des autres produits
- par **Domaine de Processus** : dans notre exemple on a séparé le commercial, la production, la gestion des ressources humaines et la gestion comptable
- par **Pays** : dans notre exemple on a séparé Brésil, Japon, USA et France



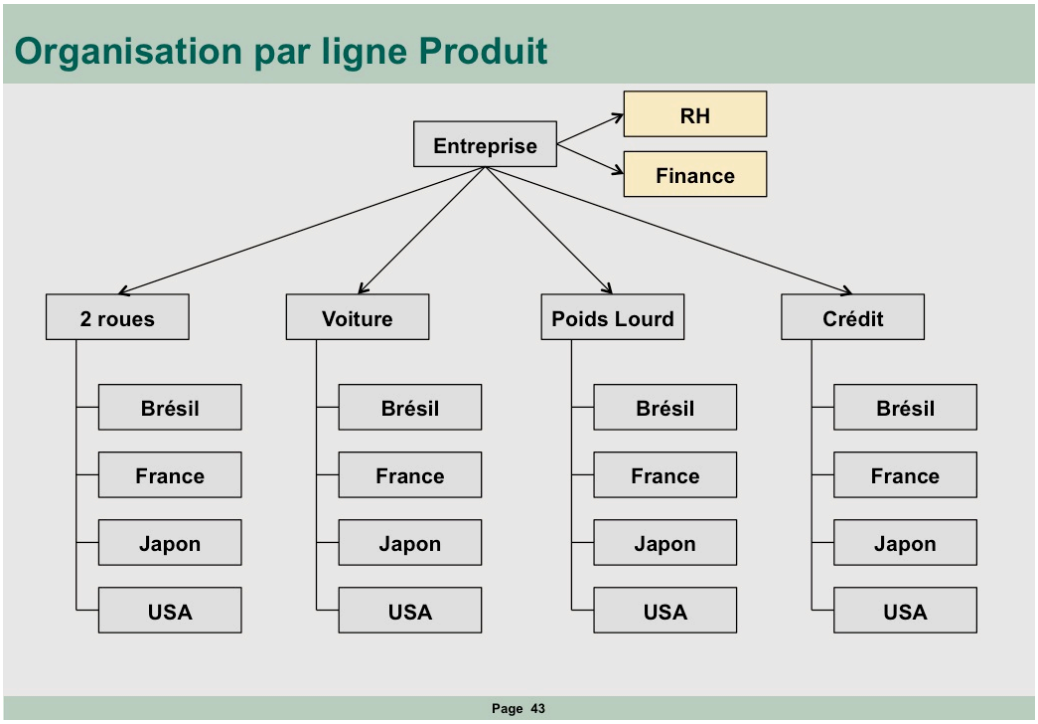
Ces différentes dimensions peuvent se combiner.

Par exemple : une organisation par pays. Chaque pays est autonome pour sa production, ses ventes, et la gestion de ses ressources humaines et de sa comptabilité.
Pour la gestion de personnel, le groupe peut centraliser

- la **construction du Modèle** de gestion de personnel: le groupe établit des règles de gestion de personnel que toutes ses filiales doivent respecter (comment recruter, comment fidéliser, comment muter, comment évaluer…)
- **certaines Opérations** : le groupe gère les cadres supérieurs, les mutations entre filiales, son propre personnel.



Le Groupe peut aussi s'organiser par Ligne Produits, ce qui est plus fréquent aujourd'hui.



On trouve aussi des organisations découpées selon d'autres dimensions telles que

- selon l'axe clientèle (particulier et Entreprises)
- selon l'axe Opérations-Transformations : on dissocie ceux qui Opèrent et ceux qui Transforment, c'est-à-dire qui mènent les projets pour le futur Modèle d'Entreprise.

Qui plus est, ces choix d'organisation évoluent au cours du temps et peuvent être panachés.

9.1.1 Rôle de l'Unité Architecture d'Entreprise

Comment s'assurer que les choix de Progiciels qui sont fait par l'une ou l'autre entité vont contribuer à la construction d'un Modèle d'Entreprise qui permet :

- d'offrir un mode **d'usage homogène** pour aider les utilisateurs à appréhender une plus grande variété de tâches, à s'identifier une seule fois, à gérer des Processus de bout en bout
- de **partager les mêmes informations** sur les produits, les clients, les contrats, les comptes, les ressources…
- de **réutiliser** des Fonctions communes telles que contrôles d'habilitation ou tarification
- de **réduire le nombre d'environnements de développement** avec lesquels on personnalise les Solutions

Il est indispensable qu'une Unité Architecture d'Entreprise soit partie prenante pour gérer le bien commun aux différentes Unités. Un nombre croissant de Groupes ont instauré cette fonction et renforcent progressivement son pouvoir pour réussir à construire progressivement une Architecture cohérente.

Cette Unité doit intervenir en amont des projets pour **définir les critères de choix des Progiciels**. Comme nous l'avons vu ci-dessus il ne suffit pas de choisir un Progiciel uniquement sur les fonctions disponibles, il faut aussi prendre en compte les trois autres critères que sont :

- la facilité à personnaliser le Progiciel
- la capacité d'évolution du Progiciel
- l'ouverture du Progiciel pour qu'il interopère avec les autres Solutions en place.

Elle doit **aider** ceux qui analysent les Progiciels qui n'ont pas toujours la compétence nécessaire pour évaluer les trois critères cités ci-dessus. L'aide des **Architectes des éditeurs** est alors indispensable : ils doivent donner tous les éléments pour que l'Entreprise prenne la bonne décision. Encore une fois la transparence est nécessaire.
L'équipe Architecture doit **valider** la Solution pressentie **avant** que ce choix ne soit proposé aux instances de décision.

Si on souhaite que l'unité d'Architecture joue réellement son rôle, la **direction du Groupe** doit être claire sur le pouvoir qu'elle lui donne vis à vis des Directions de l'Entreprise. Sinon, en cas de désaccord, l'avis des architectes d'Entreprises sera balayé par les opérationnels qui expliqueront que leur choix est le seul possible pour atteindre leurs objectifs, que leurs équipes ont déjà commencé, que le planning et le budget ne peuvent être tenus autrement…

La meilleure solution consiste à faire remonter au niveau de la Direction du groupe tous les désaccords entre Direction Opérationnelle et Unité d'Architecture. Si trop de désaccords remontent, il faut que la DG prenne des mesures soit auprès de l'Unité d'Architecture, soit auprès des Unités Opérationnelles.

9.1.2 Rôle du Métier

Le Métier jour un rôle prépondérant et croissant

- non seulement pour définir le besoin et vérifier le résultat
- mais aussi pour
    - identifier les sources d'avantage concurrentiel et de différenciation
    - personnaliser le Progiciel puisque les instruments de configuration sont à sa portée
    - réaliser les migrations

Cela n'empêche pas qu'il faille toujours des informaticiens pour développer les interfaces, procéder aux réglages techniques, migrer les informations et réaliser les développements spécifiques.

Il faut donc former les Métiers à ces nouvelles activités et également développer leur connaissance sur l'Architecture d'Entreprise pour

- comprendre l'importance d'avoir la vision globale et les enjeux de l'interopérabilité
- comprendre les bénéfices d'une ergonomie standard
- comprendre les bénéfices d'une interface utilisateur modulaire

9.2 Quelles compétences ?

La compétence informatique se raréfie dans les Entreprises.
Sous-traiter les Opérations informatiques, ou la gestion de Processus (le « BPO ») peut faire du sens pourvu que ce soit réversible.
Mais sous-traiter l'évolution du Modèle de l'Entreprise dont le logiciel fait partie est plus dangereux pour l'Entreprise qui risque de ne plus maîtriser ses évolutions et se trouver incapable d'arbitrer entre des souhaits et des coûts d'évolution parce qu'elle ne peut plus faire de lien entre les deux.
L'Entreprise doit comprendre et maîtriser son propre Modèle, et en particulier son Modèle global : elle a besoin d'un minimum d'architectes permanents.
Et même si une Entreprise fait un usage intensif du Cloud, elle doit conserver la responsabilité de son Architecture d'Entreprise.

Pour que la relation entre Architectes et Directions Opérationnelles soit sereine, les Architectes se doivent d'être compétents (connaître le métier de l'Entreprise et les solutions disponibles) et pédagogues (savoir expliquer les conséquences des choix).
Compte tenu de ce qui a été décrit ci-dessus la formation des Architectes doit inclure aussi bien des aspects **métier** que des aspects **informatiques**.

La situation idéale est de trouver des compétences doubles métier + informatique.
Mais il est extrêmement difficile de trouver ces compétences aujourd'hui : il faut donc le plus souvent associer des architectes métier et des architectes informatiques au sein de la même équipe.

Leur rôle est difficile : ils sont des empêcheurs de tourner en rond. Ils ramènent une compétence informatique dont les métiers pensaient s'être débarrassés. Et pourtant chacun comprend bien que leur rôle est absolument nécessaire pour simplifier le Modèle global de l'Entreprise.

9.2.1 Savoir faire

On insiste ici sur les compétences en Architecture à deux niveaux : Architecture Métier et Architecture Technique. Il y a une certaine continuité entre ces deux rôles : il est bon pour un Architecte Métier d'avoir des bases sur l'Architecture technique et vice versa. Par ailleurs suivant les organisations, il existe des Acteurs dédiés sur l'Architecture Métier ou non. Les domaines de compétence sont donc soit distribués sur des Acteurs différents, soit regroupés sur des Acteurs pluridisciplinaires.

Les domaines de compétences essentielles sont :

- savoir définir un **Modèle d'information** : même si le Progiciel impose son propre Modèle d'information, l'Entreprise doit posséder son propre Modèle pour mieux le comparer à celui du Progiciel (fait partie des critères de choix) et s'efforcer de définir des échanges sur la base du Modèle de l'Entreprise.
    - définition des entités essentielles du métier (environ une cinquantaine) dans un glossaire d'Entreprise
    - relations entre entités du métier
    - héritages entre entités
    - définition des identifiants des entités
    - attributs,
    - types
- savoir définir une **carte des Processus**

    - définir les grands domaines de Processus de l'Entreprise, les découper en sous-domaines
    - savoir modéliser un Processus de manière détaillée, en le décomposant en Fonctions
    - savoir distinguer entre les Processus Métier (le Quoi ?) et les Processus Organisés (le Qui et le comment ? c'est-à-dire l'affectation des Actions aux différents Acteurs dans le cadre d'un scénario d'organisation choisi)
    - relier les Processus aux Entités Métier
- Savoir définir une cartographie Fonctionnelle pour favoriser la réutilisation de Fonctions dans différents Processus
    - définir les domaines de Fonctions, les découper en sous-domaines
    - savoir définir les Fonctions Métier et leurs interfaces
    - définir les liens avec les Processus Métier
- savoir **cartographier** l'Architecture des Solutions
    - liste des Solutions
    - rôle fonctionnel de chaque Solution
    - qui est maître des données
    - qui offre des Services
    - nécessité d'interfaces entre Solutions
    - gérer les liens avec la cartographie Fonctionnelle
- savoir définir les **référentiels partagés** : contenu, localisation, périmètre, services d'accès
- savoir définir les **échanges** d'informations entre Solutions
    - synchrone ou asynchrone
    - interfaces standards ou spécifiques
    - outils de construction d'interfaces
- savoir définir les échanges de **Services** entre Solutions
- savoir gérer un **référentiel** évolutif de la **Fondation** de l'Entreprise : tout ce qui est commun aux différents modèles de Solutions (modèles d'information, liste de Services réutilisables…)
- savoir définir un **mode d'usage standardisé** : ergonomie, navigation, portail spécifique, identification et authentification
- savoir **modéliser les Processus métier essentiels** de l'Entreprise : ils sont en général peu nombreux
- savoir comment gérer des **Processus de bout en bout**
- savoir comment utiliser les **Solutions transverses** : reporting, compta,…
- savoir **analyser les caractéristiques d'un Progiciel**, indépendamment des déclarations de l'Editeur

9.2.2 Savoir être

Ils doivent également maîtriser certaines compétences comportementales :

- savoir **expliquer** l'utilité de leurs recommandations
- savoir **convaincre** lorsque leurs recommandations sont difficiles à accepter
- savoir **animer** un réseau de professionnels : les architectes d'Entreprise doivent souvent former et accompagner des architectes applicatifs dans les projets, qui mettront en œuvre leurs recommandations et réutiliseront les briques de la Fondation. Le fonctionnement en cercle d'architectes animé par une équipe centrale se généralise dans les grands groupes.

On distingue l'Architecte d'Entreprise et l'Architecte de Projet : le premier couvre les besoins de coordination globale au niveau Entreprise ou au niveau d'un domaine de l'Entreprise (on parle alors parfois d'Architecte de domaine), alors que le second gère les choix d'Architecture au sein d'un Projet. Le premier est transversal aux Projets et maintient la cohérence globale dans le temps au delà des frontières des Projets. Le second se borne aux choix d'Architecture pour un projet, en s'assurant d'utiliser au mieux la Fondation de l'Entreprise.

9.2.3 Architecte d'Entreprise

Rôles essentiels d'un **Architecte d'Entreprise**

- construction
  - modèle des **Domaines fonctionnels**
  - modèle des **objets** métier essentiels de l'Entreprise: les 50 objets essentiels
  - modèle des **Solutions** de l'Entreprise: la cartographie
  - modèle des **Processus** essentiels du cœur de métier (ils sont en général peu nombreux) : ce sont souvent les Processus cross business
  - gestion du catalogue d'interfaces publiques des **services** qui implémentent les Fonctions communes: recherche des standards
- support
  - aide aux architectes de projet
  - certification des projets

9.2.4 Architecte de projet

Rôles essentiels d'un **Architecte de projet**

- Pré-étude :
  - identifier les principales alternatives possibles
  - réaliser une analyse de la Valeur (analyse bénéfices/coûts)
  - prendre en compte la Fondation
  - contribuer au choix des Progiciels
- Conception
  - définir l'Architecture générale de la Solution
  - résoudre les difficultés majeures d'Architecture
  - le cas échéant, faire réaliser un prototype pour valider ces choix
- Mise en œuvre
  - former les équipes projets
  - fournir des Solutions sur les points bloquants lors de la mise en œuvre des choix d'Architecture
  - contrôler le respect des choix d'Architecture

9.3 Quelles Formations ?

9.3.1 Formation pour l'architecte

Formation Technique

- connaissance des standards ouverts / interopérabilité
- best practices sur la mise en place d'infrastructures / communication
- connaissance des standards internes de l'Entreprise en matière d'interopérabilité
- connaissance de l'Architecture du Progiciel
- connaissance de l'environnement de déploiement du Progiciel
- connaissance de l'environnement de développement du Progiciel

Formation Métier

- quel est le Modèle d'Information : Acteurs, produit, contrat…(Comprendre et parler le langage du Métier)
- quels sont les principaux Processus Métier
- quelles sont les Fonctionnalités essentielles
- quelle est l'organisation qui va utiliser le Progiciel, et ses attentes essentielles
- maitriser les techniques d'analyse de la Valeur

Formation aux compétences comportementales

- etre synthétique et structuré
  - résumer un problème/projet en une page
  - distinguer l'essentiel de l'accessoire (en identifiant les critères discriminants permettant la synthèse)
  - ordonner ses idées de manière logique et priorisée en fonction du contexte

- vendre et convaincre
  - relier ses arguments aux buts
  - écouter et questionner les positions des autres
  - vendre sa Solution aux décideurs, et négocier le tri des modifications demandées en cours de projet
- gérer les conflits
- etre innovant
- travailler en équipe
- gérer les environnements multi-culturels

9.3.2 Formation pour l'équipe Projet

- savoir exprimer ses besoins sans prescrire la Solution
- savoir relier ses objectifs à la Stratégie de l'Entreprise
- savoir analyser les bénéfices/coûts d'une Solution
- savoir définir des priorités
- connaître la Fondation :
  - connaître les méthodes et outils de gestion de projet de l'Entreprise
  - connaître les modèles globaux :
    - modèle d'Information : principaux objets Métier du domaine ou de l'Entreprise
    - carte des Processus
    - cartographie Fonctionnelle
- savoir prendre en compte le cycle de vie complet du Progiciel : Sélection, mise en œuvre, exploitation, maintenance, décommissionnement
- vendre et convaincre
  - vendre sa Solution aux décideurs
  - savoir négocier avec les Fournisseurs
- etre innovant
  - prendre de la distance par rapport à son existant
  - savoir considérer les options du Progiciel en réponse à une exigence et ne pas prescrire systématiquement ses propres habitudes
- savoir décider rapidement
  - arbitrer les choix de conception par rapport aux objectifs
  - se concentrer sur les priorités pour aboutir dans les délais
- conduire le changement
  - expliquer la vision
  - former au fonctionnement la nouvelle Solution
  - contrôler l'appropriation des utilisateurs
  - fournir le support nécessaire au démarrage

10 Annexe 1 - Questionnaire étude de cas Client

10.1 Questions clé

- Quelle est la maturité de la démarche processus dans l'Entreprise (avant et après le projet) ?
- Comment le choix du progiciel a été fait (la démarche processus a-t-elle été un critère) ?
- Mode tiré (je décris mes processus et je les mets dans le progiciel) ou poussé (je prends ceux du progiciel) ?
- Comment l'ERP est venu impacter cette démarche processus et l'EA ?
- Qu'est-ce qui a changé dans la manière de faire le métier ? (opérationnels et IT)

10.2 rappeler le but du projet

10.2.1 présenter Entreprise

- Date création
- son Marché : ses produits/clients/territoire
  - chiffres clés
- Ses Opérations : organisation (nationale vs internationale, centralisée vs décentralisée), Marché BtoB/BtoC, partenaires, effectifs
  - chiffres clés

10.2.2 Démarche processus dans l'Entreprise (globalement, en dehors du projet étudié)

Q. 2 :
Quels sont les processus décrits dans votre Entreprise qui vous vienne spontanément à l'esprit ?

Q. 3 :
Connaissez-vous les fonctions propriétaires de vos processus suivants :
- gestion RH
- amélioration de la qualité
- gestion des ventes
- gestion de la production

Q. 4 :
Etes-vous propriétaire d'un processus ?

Q. 5 :
Votre directeur général vous a-t-il défini des indicateurs de performance transverse (taux de retour des produits défectueux, nombre d'appels au S.A.V., etc)

Q. 6 :
Comment votre directeur général obtient-il les tableaux de bord d'activité / des processus ?

Q. 7 :
Quels processus « transverses » avez-vous identifiés ?

Lesquels sont décrits ?

Lesquels sont outillés selon vous ?

Le sont-ils par un progiciel ?

Pour ceux qui le sont, par quel type de progiciel (présenter notre classification) ?

Q. 8 :
Disposez-vous d'un outil informatique unique pour toutes les activités de l'Entreprise ?

Q. 9 SI:
Quelle est la couverture fonctionnelle de vos progiciels (cf. tableau maîtres d'ouvrages) ?

Q. 9-1 :
Qu'est-ce pour vous que la « cartographie » des processus ?

Q. 10-1 :
Quels sont les outils informatiques utilisés le long de vos processus ? :

Proposer un tableau

Processus ou domaine de processus	Outillé ? O/N	Progiciel ? O/N	Catégories de progiciels Nom du progiciel ? Ou O/N				
RH							
Commercial							
Facturation							
Production							
Finances							
GRC							

Q. 10-2 :
Avez-vous instauré un pilotage transverse de vos processus ?

10.2.3 périmètre du projet

- (fonctionnel et géographique) et relations avec autres Solutions

10.2.4 objectifs et indicateurs

- remplacement ou nouvelle Solution ?

10.2.5 contraintes projet

délais, budget, implication des équipes internes

10.3 définir la Cible

10.3.1 cartographie des Solutions avec lesquelles s'interfacer

- Avez-vous utilisé des **Services** logiciels proposés sur le web (ex : infos bourse, météo, plan, état des pistes, Amadeus…)?

10.3.2 contenu de chaque Solution

Principales fonctionnalités et données gérées

10.3.3 Modèle d'information

- le client a-t-il un Modèle d'information (Définition des principaux objets métier)
- le progiciel publie t il son Modèle d'information
- si oui (pour les deux) : quels sont les écarts ? comment les gérer ?
- qui doit être maître des données : comment transférer (réplication asynchrone ou réplication synchrone ou service synchrone)

10.3.4 interfaces

10.3.4.1 lister les interfaces

- nom

- synchrone ou asynchrone
- sens
- volumes
    - comment est définie l'interface : standard-marché ou standard-progiciel ou standard-client ou spécifique
    - réduit on le nb d'interfaces à un nb limité « d'interface-pivot » ou autant d'interfaces que d'échanges possibles ?
- où est fait la conversion : dans le progiciel, dans les Solutions en interface, dans une Solution Intermédiaire
- en cas d'évolution,
    - comment synchroniser ? versioning des interfaces ?
    - comment mesurer les impacts ? outil ?

10.3.4.2 quel middleware

- synchrone et asynchrone

10.3.4.3 Processus inter-Solutions

Avez-vous des processus orchestrant des appels à différentes solutions ? Si oui, comment ?

10.4 comment le choix du progiciel a-t-il été fait ?

Q. 7 SI:
Sur quoi se basait votre SI avant ce choix ?

- pourquoi un progiciel plutôt qu'un développement spécifique?

Q. 10 SI:
La description de vos processus vous a-t-elle guidé dans le choix du progiciel ?

- quelle a été votre grille de critères ?
- prototype ?
    - fonctionnel
    - technique : capacité à s'interfacer, montée en charge ?
- «l'intégration dans votre Architecture » a-t-il été un critère de choix : si oui, quelle grille d'analyse du progiciel avez-vous utilisée pour comprendre son Architecture interne et l'intégrer au mieux à l'existant de l'Entreprise ?

10.5 comment le projet s'est déroulé ?

10.5.1 Planning

10.5.2 Charge

- effectifs
- quelle part de la charge pour intégration (développement d'interfaces)

10.5.3 Rôles

- Métier : quelle implication du Métier pour
    - la définition des nouveaux processus,
    - l'analyse d'écarts,
    - la configuration : le paramétrage et le moteur de règles ont-ils été directement utilisés par le Métier ?
    - les tests…

Q. 1 SI:
Comment avez-vous envisagé la description des processus pour les métiers ?

Q. 2 SI:
Comment avez-vous décrit vos processus dans le cadre de ces activités ?

Q. 3 SI:

Si non à Q2, comment vous êtes vous assuré de la réelle couverture fonctionnelle sans description des processus ?

Q. 12 SI:
Avez-vous été contraints de reconsidérer vos processus à cette occasion ?

Q. 11 SI:
La description de vos processus a-t-elle été un bénéfice ou une contrainte pour le paramétrage et la mise en place de vos progiciels ?

- Informatique
  - quelle implication des Urbanistes ou des Architectes d'Entreprise?
  - développements spécifiques,
  - configuration
  - Migration
  - Interfaces
- Intégrateur
  - faut-il un intégrateur ? que lui demander ? doit-il être préalablement expérimenté ?
  - quelles **exigences** avez-vous eu vis-à-vis des **intégrateurs** de progiciels ? (interfaces, migration)
- éditeur
  - quelle **formation** de la part de l'éditeur pour vos équipes de
    - gestion de projet et pilotage
    - personnalisation et d'évolution
  - quelles **exigences** avez-vous eu vis-à-vis des éditeurs de progiciels ? (accès à l'Architecture interne, services ouverts, respect des standards du marché et du client, documentation)
  - quels **outils de l'éditeur** de Progiciel avez-vous utilisés pour des développements complémentaires (plate forme de développement, intégration SOA …) ?

10.5.4 Approche classique ou agile ?

Q. 8 SI:
Avez-vous suivi le modèle des processus métiers fourni par votre éditeur de progiciel ?

10.6 exigences vis-à-vis de l'intégrateur

- exigence de compétence métier
- exigence de compétence méthode
- exigence de compétence sur le progiciel

10.7 exigences vis-à-vis de l'Editeur

- Quelle **formation** de la part de l'éditeur pour vos équipes
- Quelles **exigences** avez-vous eu vis-à-vis de l'éditeur de progiciels ?
  - accès à l'Architecture interne : modèle d'informations,
  - services ouverts
  - Demandes d'évolution souhaitées :
  - demandes d'interfaces standards :
- Quels **outils de l'éditeur** de Progiciel avez-vous utilisés pour des développements complémentaires (plate forme de développement, intégration SOA …) :

10.8 bilan du projet d'intégration d'un Progiciel

Q. 11 :
Comment les choses se sont-elles passées : difficultés, obstacles, facilités ?
- les **fonctions** livrées sont elles conformes ?

Q. 6 2:
Le progiciel vous a-t-il facilité la tâche

Dans quels délais y avez-vous répondu de manière continue ?

- **Coût** : quels sont les coûts réels pour intégrer le progiciel
  - coût du progiciel : licence, maintenance, droit d'utilisation, run time…
  - coût d'intégration : développements spécifiques, configuration du progiciel, tests, pour acteurs du métier ou informaticiens
  - coût de migration des données
  - coût de déploiement du progiciel : installation, formation
  - coût opérationnel : exploitation, hot line
- **Délai**
- Capacité à respecter les **spécificités** du client
- Capacité à **s'intégrer** avec le reste du SI : combien d'interfaces : lesquelles sont SOA ?

Q. 13 SI:
Le progiciel vous a-t-il fourni des outils de pilotage des processus (indicateurs de performance transverses) ?

Q. 9-2:
Les outils informatiques sont-ils alignés sur une cartographie ?

Comment vous y êtes-vous pris pour y parvenir ?

Le progiciel vous a-t-il aidé dans ce travail ? Si oui comment ?

- Gestion du changement : acceptation par les utilisateurs, l'exploitation informatique
- Les indicateurs définis dans le But sont-ils atteints ?
- Avez-vous intégré plusieurs Progiciels pour satisfaire votre But ?
- comment le progiciel a-t-il ensuite **évolué** ?
  - Comment intégrer les demandes de l'Entreprise dans la road-map produit de l'éditeur ?
  - Quelle organisation pour la maintenance du spécifique (développement et paramétrage) ?
  - Quelle montée de version ? combien de temps une version est-elle maintenue ?
  - Quel rythme d'évolution du progiciel ?
- **leçons** tirées du projet

Q. 12 :
Quelles leçons avez-vous tiré de ces mises en place ?

- a-t-on pris un marteau pilon pour écraser une mouche ?
- si vous deviez refaire le projet aujourd'hui, quelle solution choisiriez-vous ?

Q. 14 SI:
Comment les choses se sont-elles passées : difficultés, obstacles, facilités ?

Q. 15 SI:
Quelles leçons avez-vous tiré de ces mises en place ?

11 Annexe 2 - Questionnaire éditeur de logiciel

11.1 Questions Générales

- Présentation **d'études de cas** : réussies ou non
- votre **vision** sur les Progiciels de demain :
  - couverture fonctionnelle plus large ou industrialisation des échanges entre Progiciels ?
  - Services ouverts ou Processus complets ?
- **avez-vous connaissance d'études générales** sur les Progiciels qui nous aide à répondre à des questions telles que :
  - part des Progiciels vis-à-vis des développements spécifiques ?

- doit on gérer les Progiciels de Commodité comme les Progiciels cœur de métier ?
- …

11.2 Questions sur vos Produits

11.2.1 Quel est le périmètre couvert ?

périmètre fonctionnel

- Votre logiciel informatise-t-il des lignes Produit ? (Assurance, Santé, Marchés financiers, Télécoms, Utility…)
- Votre logiciel informatise-t-il des Domaine de Processus : CRM, comptabilité, HR, …
- Votre logiciel propose-t-il des Fonctions telles que « scorer », « calculer tarif », « contrôler habilitation », « s'identifier »…

périmètre technique

- votre logiciel propose t il des outils pour construire des Solutions (ex : AGL, outil de modélisation)
- votre logiciel propose t il des outils pour opérer des Solutions (Middleware)

11.2.2 Quelle interface utilisateur offrez vous ?

HTML, Client-serveur ou de type Citrix
si léger : que propose le ProgicielProgiciel

- sa propre interface utilisateur
- l'interface utilisateur du client qui fait appel à des services du ProgicielProgiciel
- les 2

11.2.3 Quels sont les mécanismes généraux offerts aux utilisateurs ?

- desktop ?
- identification et authentification ?
- corbeille ?
- synthèse client ?
- envoi de messages ?
- production de documents immédiats ou différés ?
- quels services d'accès sont disponibles : portail, smartphones… ?

Comment coexiste-t-ils avec les fonctions existantes en Entreprise ?

11.2.4 Pour quelle taille d'Entreprise-cliente ?

- quelle capacité peut il traiter : nb d'objets ou nb de transactions
- la même instance de Progiciel peut-elle traiter simultanément plusieurs Entreprises ?

11.2.5 Comment se situe le Progiciel par rapport à la typologie ?

CRITERES	Votre Progiciel métier
Couplage faible ou fort	
Commodité ou Cœur de métier	
Livré avec configuration métier prêt à l'emploi	
Personnalisation forte ou faible	
Nature: Classique ou cloud	
International	
Techno	
Poids dans le SI (%)	

Si Progiciel technique : commentaire libre.

11.2.6 Architecture du Produit

Quel est le découpage en couches ?

- est-il découpé en quatre couches : les données, les Services, les Processus et les interfaces utilisateurs ?
- peut on modifier le modèle de données sans toucher au logiciel ?
- quelles possibilités de réutilisation ? approche objet ?
- Comment sont gérées les évolutions dans le temps (le versioning) ?
- Comment implémenter les Processus ?

Peut-on n'installer qu'une partie du Progiciel ?

Peut-on utiliser la Fondation du Progiciel pour construire d'autres Solutions ?

Comment sont isolées les spécificités de chaque pays ?
Le Progiciel vise-t-il à offrir une Solution unique pour tous les pays.
Est-il construit pour isoler ce qui propre à chaque pays : devise, langue, réglementations.

Quels Services sont mis à disposition des autres Solutions ?

- Interface utilisateur externe : site web déjà défini
- nouveaux terminaux clients pour mobilité (smartphone, tablette…)

Le Progiciel peut-il appeler des services externes ?

Quelle plateforme de déploiement ?

Quels standards ?

- OS
- DBMS
- Middleware et plateforme d'échange

Quelle capacité à changer de technologie ?

- peut-on changer d'OS ?
- peut-on changer de gestionnaire de base de données ?
- peut-on changer de middleware ?

Offre Cloud

- (public)? comment l'interfacer à des Solutions déjà en place
- Qui porte la responsabilité de **l'intégration** des différentes briques d'une solution qui comporte des services hébergés sur le Cloud et d'autres en interne ?

Quel Modèle Métier ?

- Le Modèle Métier (Modèle d'Informations, Modèle de Processus et Modèle de Fonctions) est-il **formalisé** ?
  - existe-t-il un repository ?
  - existe-t-il un méta model ? et un outillage associé pour y accéder ? quels mécanismes :
    - OO ?
    - Relations sophistiquées ?
    - Transaction métier ?
    - Versionning ?
- Le Modèle Métier est-il **public** ?
- Le Modèle Métier est-il **extensible** par le client ?
- Si oui comment **maintenir** les deux parties ?

Quels Outils ?

- Quels outils pour **Programmation** de logiciels ?

- Quels mécanismes de **Configuration** ?
  - paramétrage ?
  - table à n dimensions ?
  - moteur de règles ?
  - moteur de workflow ?
  - attributs dynamiques ?
- Quels outils pour la **migration** des données?
- Quels outils pour construire les **Interfaces** ?

11.3 Questions sur votre méthodologie

- quels Processus-projets sont recommandés ; quels outils les supportent ?
  - Processus d'intégration du Progiciel
  - Processus d'évolution du Progiciel
  - Processus de montée de version du Progiciel

11.3.1 Existe-t-il un Processus type pour intégrer le Progiciel ?

Modèle = méthode + outils associés

Quel Modèle pour le gap analysis ?

Quel Modèle pour l'évaluation des charges et la planification ?

Quel Modèle pour l'organisation des équipes et la gouvernance ?

Quel Modèle pour l'analyse des gaps et leur traduction en développement spécifique ou configuration ?

Quel Modèle pour les développements spécifiques ?

Quel Modèle pour la configuration ?
Peut-on créer de nouveaux Produits par configuration ?
Peut-on créer de nouveaux Processus par configuration ?
Quelles autres évolutions permet la configuration ?

- configuration de la sécurité ?
- configuration de la structure d'organisation ?
- configuration d'impressions ?

Quel Modèle pour les tests ?
Quel Modèle pour l'intégration ?
Quel Modèle pour les interfaces d'échange ?
Quel Modèle pour les reprises de données ?
Quel Modèle pour le déploiement ?

11.3.2 Processus d'évolution du Progiciel chez l'éditeur

Quel Processus pour rassembler les suggestions d'évolution : clients, sales/marketing,… ?

Quel Processus pour arbitrer ?

Quel Processus pour livrer les versions successives ?

- rythme de livraison de versions

11.3.3 Processus d'évolution du Progiciel chez le client

Quel Processus pour upgrader une version du Progiciel chez un client déjà équipé ?